2024 International Joint Conference on Neural Networks (IJCNN) | 979-8-3503-5931-2/24/$31.00 ©2024 IEEE | DOI: 10.1109/IJCNN60899.2024.10650494

DOS-GNN: Dual-Feature Aggregations with Over-Sampling for Class-Imbalanced Fraud Detection On Graphs

Shixiong jing
Pennsylvania State University
jing@psu.edu

Lingwei Chen
Wright State University
lingwei.chen@wright.edu

Quan Li
Pennsylvania State University
qbl5082@psu.edu

Dinghao Wu
Pennsylvania State University
dinghao@psu.edu

***Abstract*—As fraudulent activities have shot up manifolds, fraud detection has emerged as a pivotal process in different fields (e.g., e-commerce, online reviews, and social networks). Since interactions among entities provide valuable insights into fraudulent activities, such behaviors can be naturally represented as graph structures, where graph neural networks (GNNs) have been developed as prominent models to boost the efficacy of fraud detection. In graph-based fraud detection, handling imbalanced datasets poses a significant challenge, as the minority class often gets overshadowed, diminishing the performance of conventional GNNs. While oversampling has recently been adapted for imbalanced graphs, it contends with issues such as graph heterophily and noisy edge synthesis. To address these limitations, this paper introduces DOS-GNN, incorporating Dual-feature aggregation with Over-Sampling to advance GNNs for class-imbalanced fraud detection on graphs. This model exploits feature separation and dual-feature aggregation to mitigate the impact of heterophily and acquire refined node embeddings that facilitate fraud oversampling to balance class distribution without the need for edge synthesis. Extensive experiments on four large and real-world fraud datasets demonstrate that DOS-GNN can significantly improve fraud detection performance on graphs with different imbalance ratios and homophily ratios, outperforming state-of-the-art GNN models.**

***Index Terms*—Fraud Detection, Graph Neural Networks, Class Imbalance, Heterophily.**

I. Introduction

Increased connectivity of devices and individuals to the Internet has continuously transformed our daily lives in various aspects, such as socializing, online banking, and healthcare. Despite apparent benefits, they also create an ever-expanding surface for fraudsters to exploit these networks for their economic, social, or political intents intentions [1]. As fraudulent activities have shot up manifolds, fraud detection has emerged as a pivotal process in the fields of e-commerce [2], [3], healthcare [4], online reviews [5], [6], and social networks [7]. Fraudulent entities (e.g., accounts, reviews, and transactions) often disguise their malicious nature by incorporating genuine information, making it challenging to directly identify them based on individual attributes, while their interactions with other entities offer valuable clues that can expose themselves [8]. As such, fraudulent activities can be naturally abstracted into graph structures, enabling a more intuitive analysis to reveal the suspicious entities at the graph level [9].

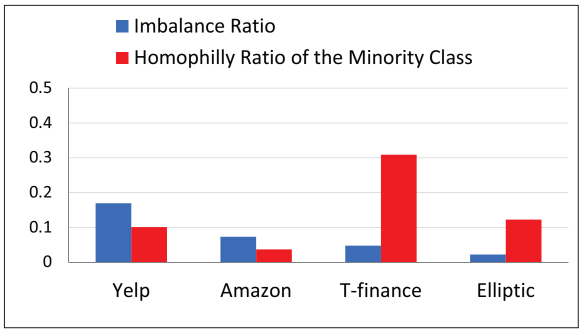


Fig. 1: Imbalance ratio and homophilly ratio of minority class for Yelp, Amazon, T-finance, and Elliptic datasets.

Due to their remarkable learning capabilities [10], [11], graph neural networks (GNNs) [12], [13] have been developed as prominent models to boost the efficacy of fraud detection [5], [6], [14], [15]. In this line of work, fraud detection is cast as a node classification problem, where GNN models are designed to follow the message-passing paradigm, enabling the propagation of information from labeled nodes to unlabeled ones through the graph structure. Nevertheless, graphs used in fraud detection commonly exhibit a natural imbalance among labeled nodes due to the typical rarity of frauds. To put it into perspective, we calculate and present the imbalance ratios (as defined in Sec. II) of four real-world fraud detection datasets (detailed in Sec. IV-A) in Fig. 1, where most of these ratios are below 0.1, indicating that frauds are far less than legitimate instances. Class imbalance is prevalent in fraud detection, which, however, has rarely been considered by conventional GNN models. When the GNN model is trained on imbalanced labeled data, it is difficult to classify the instances in the minority class as fraud, leading to biased predictions [16].

To mitigate the impact of class imbalance on GNN models, many oversampling techniques have been tailored for imbalanced graphs, such as GraphSMOTE [17] and GraphENS [18]. These methods involve synthesizing nodes in the embedding space and subsequently generating edges to connect them with existing nodes [16]. Unfortunately, node embedding is intricately linked with its neighborhood, while fraudsters tend to establish relationships between fraudulent and legitimate entities for camouflage and evasion [14]. This results in a high degree of heterophily [19] for fraud nodes (illustrated in Fig. 1 where most fraud detection datasets exhibit extremely low homophily ratios), and their embeddings are significantly smoothed by neighborhood aggregation using GNNs designed for homophily [20]–[22], rendering the synthesized nodes ineffective. Furthermore, synthesized edges may not accurately represent real relationships between nodes due to constraints. For example, on social networks, an account can only be connected to others when the following request gets approved. The addition of synthesized edges may inadvertently introduce noise, potentially undermining the intended enhancement of neighborhood information and diminishing the efficacy of message passing and the resulting node embeddings.

To address these limitations, this paper introduces DOS-GNN, which implements **D**ual-feature aggregation with **O**ver-**S**ampling to advance **GNN**s for class-imbalanced fraud detection on graphs. DOS-GNN proceeds by acquiring informative and distinguishable node embeddings through homophilic and heterophilic feature aggregation, and then employing an oversampling algorithm on these refined embeddings to exclusively synthesize fraud nodes, achieving a balanced class distribution for fraud detection. The leveraged node embeddings encode both node features and graph structure, which eliminates the necessity for edge synthesis.

Recent methods used to handle graph heterophily primarily focus on adaptive message aggregation [19], which extracts various signals from neighbors by training separate filters to manipulate homophily and heterophily, and mixes the results from these filters into a single vector as node embedding at each neighborhood aggregation step [23]–[25]. This easily leads to dissimilar features being overshadowed by similarities and subsequently becoming too insignificant for further propagation to higher orders. In contrast, DOS-GNN elaborates a simpler yet more effective neighborhood aggregation to avoid over-smoothing on minority node embeddings resulting from heterophily. More specifically, each node preserves its similar and dissimilar features as two separate vectors, and enables dual-feature aggregation to absorb the corresponding signals from neighbors through connection property (i.e., if the edge connecting the target node and its neighboring node is homophilic or heterophilic). By maintaining each node's similarities and dissimilarities locally without excessive smoothing or compromise, these features can be effectively propagated to other nodes with long-range associations. This mitigates the impact of heterophily on graph learning, amplifies node embeddings and the subsequent oversampling, thereby improving class-imbalanced fraud detection performance. In summary, our major contributions are listed as follows:

- A novel oversampling paradigm is proposed for imbalanced graphs in fraud detection, which exclusively synthesizes fraud nodes on node embeddings without the necessity for edge synthesis.
- A simple yet effective dual-feature aggregation is elaborated to mitigate heterophily and enhance node embeddings that facilitate oversampling.
- Extensive experiments are conducted on four real-world datasets, which demonstrate that DOS-GNN can achieve state-of-the-art fraud detection performance on graphs with different imbalance and homophily ratios.

II. Preliminaries

Notations. A given fraud graph is denoted as $G = (V, E, \mathbf{X})$, where $V(n = |V|)$ is the set of entities (e.g., accounts, reviews, and transactions), E is the set of edges indicating reciprocal links between entities, and $\mathbf{X} \in \mathbb{R}^{n\times d}$ is the feature matrix. Edges E can be further encoded as an adjacency matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$. The neighbors for v_i is represented as $\mathcal{N}(v_i) = \{v_j|(v_i, v_j) \in E)\}$. Each labeled node is associated with a ground truth $y \in Y = \{0 : legitimage, 1 : fraudulent\}$.

Graph Neural Networks. In this paper, fraud detection is cast as node classification, which aims to learn a GNN model $f_{\mathbf{W}} : (\mathbf{A}, \mathbf{X}) \to \mathbf{y}$ where $\mathbf{W}$ is the model parameters and $\mathbf{y}$ is the set of labels. Generally, GNN models enforce each node to aggregate information from its neighbors and generate higher-level node embedding [26]. This graph aggregation layer can be defined in a form as follows:

$$\mathbf{H}^{(l)} = \text{aggregate}\left(\mathbf{H}^{(l-1)}, \mathbf{A}, \mathbf{W}^{(l)}\right) \tag{1}$$

where $\mathbf{H}^{(l-1)}$ and $\mathbf{H}^{(l)}$ are the input and output for layer l ($l \geq 1$), $\mathbf{W}^{(l)}$ is a learnable weight matrix, and $\mathbf{H}^{(0)} = \mathbf{X}$. The final output $\mathbf{Z}$ of GNNs with L layers can be computed using a softmax function.

While GNNs can be applied under inductive and transductive settings, we focus on transductive inferences in this paper where all node connections and features are accessible during training. Also, different variants of GNN, such as GCN [13], GAT [27], or others [12], [28], have different aggregation mechanisms. In this respect, we use a GCN as the base model $f_{\mathbf{W}}(\mathbf{A}, \mathbf{X})$ to facilitate the evaluation of DOS-GNN, which aggregates neighborhood information by performing:

$$\mathbf{H}^{(l)} = \sigma\left(\tilde{\mathbf{A}}\mathbf{H}^{(l-1)}\mathbf{W}^{(l)}\right) \tag{2}$$

where σ is an activation function, $\tilde{\mathbf{A}} = \hat{\mathbf{D}}^{-\frac{1}{2}}\hat{\mathbf{A}}\hat{\mathbf{D}}^{-\frac{1}{2}}$, $\hat{\mathbf{A}} = \mathbf{A} + \mathbf{I}$, and $\hat{\mathbf{D}}$ is the diagonal degree matrix defined on $\hat{\mathbf{A}}$.

Class Imbalance on Graphs. Fraud graphs exhibit a nature of class imbalance due to the fact that fraudulent entities are often rare. We define the imbalance ratio to quantify such nature to better understand the data challenge for fraud detection on graphs. Specifically, this imbalance ratio r_i can be specified on the labeled nodes as follows:

Definition 1 (**Imbalance Ratio).** Given a fraud graph G where N_{minor} represents the number of fraudulent entities and N_{major} signifies the number of legitimate entities, the imbalance ratio can be written as

$$r_i = \frac{N_{minor}}{N_{major}} \tag{3}$$

Homophily and Heterophily. When generalizing to graphs, homophily suggests that nodes tend to connect with others

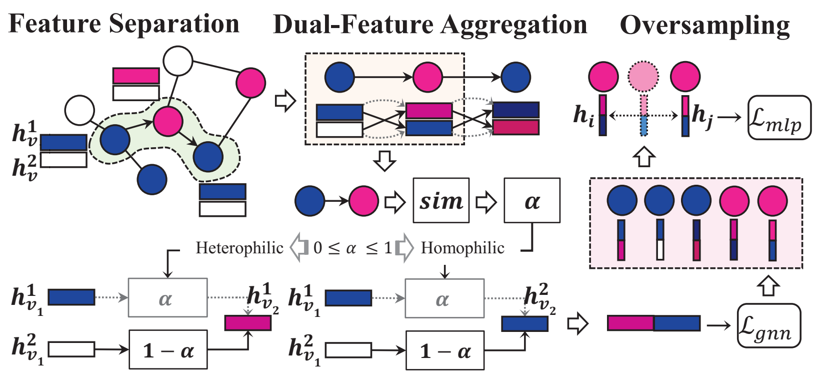


Fig. 2: Overview of our proposed model DOS-GNN for class-imbalanced fraud detection on graphs: the model proceeds with feature separation to maintain each node's similarities and dissimilarities locally without excessive smoothing or compromise, which facilitates dual-feature aggregation to address heterophily and acquire fraud-specific embeddings; these embeddings are used to perform oversampling to generate a balanced class distribution for fraud detection.

sharing similar features [21]. This paper focuses on homophily in class labels [22], where a graph with good homophily indicates that connected nodes share the same label with a high probability. Accordingly, homophily ratio can be defined as follows to quantify the degree of graph homophily:

Definition 2 (**Homophily Ratio**)**.** Given a graph G, the homophily ratio is the proportion of edges in G that connect nodes sharing the same labels, which can be calculated as

$$r_h = \frac{|\{(v_i, v_j) : (v_i, v_j) \in E \wedge y_{v_i} = y_{v_j}\}|}{|E|} \tag{4}$$

In fraud graphs, we are more interested in homophily ratio for minority (fraudulent) class to better understand the impact of heterophily on fraud node oversampling. r_h can be further constrained by

$$r_h = \frac{|\{(v_i, v_j) : (v_i, v_j) \in E \wedge y_{v_i} = y_{v_j} = 1\}|}{|\{(v_i, v_j) : (v_i, v_j) \in E \wedge (y_{v_i} = 1 \vee y_{v_j} = 1)\}|} \tag{5}$$

Heterophily is the opposite of homophily to describe the status of connected nodes belonging to different labels. Graphs with high homophily have $r_h \to 1$, while graphs with high heterophily exhibit low homophily with $r_h \to 0$.

III. Proposed Model: DOS-GNN

In this section, we present our proposed fraud detection model DOS-GNN for technical details. Its overview is illustrated in Fig. 2. DOS-GNN proceeds with dual-feature aggregation to address heterophily and acquire fraud-specific embedding, and oversampling in node embedding space to generate a balanced class distribution for fraud detection.

A. Feature Separation

When a graph has low homophily ratio, conventional GNN models that solely apply low-pass filters [29], [30] to aggregate neighborhood features tend to smooth the outputs from different classes and thus misclassify these nodes [24]. To restore the discrimination capability in such graphs, GNN models need to harness nodes' surrounding dissimilarities, which prompts the utilization of high-pass filters [31] to extract the information of neighborhood differences and address heterophily [25]. Following this idea, different adaptive message aggregation methods [23]–[25], [32] have been deployed to train separate filters to extract similar and dissimilar signals from neighbors to mitigate the impact of heterophily on graph learning. Though these approaches extract dissimilarities from neighbors, they are still mixed with similarities to be compromised as a single feature vector; such straightforward preservation may not be helpful for propagating dissimilarities to higher-order nodes [33].

Since separating the similar and dissimilar signals from node features provides a feasible way to deal with graphs with different homophily ratios [23], we propose to preserve these similarities and dissimilarities separately. Specifically, we elaborate two distinct feature vectors for each node $v \in V$ to facilitate feature aggregation: similar feature vector $\mathbf{h}_v^1$, and dissimilar feature vector $\mathbf{h}_v^2$. $\mathbf{h}_v^1$ is used to aggregate and preserve similar signals from neighborhood features, while $\mathbf{h}_v^2$ is used to preserve dissimilar signals extracted from neighbors. In this respect, the representation for each node $v \in V$ during aggregation can be formalized as

$$\mathbf{h}_v = \{\mathbf{h}_v^1, \mathbf{h}_v^2\} \tag{6}$$

For a given graph G, considering that dissimilarities are not existing prior to feature aggregation, we initialize $\mathbf{h}_v$ as $\mathbf{h}_v = \{\mathbf{X}_v, \mathbf{0}\}$, where $\mathbf{h}_v^1 = \mathbf{X}_v$, $\mathbf{h}_v^2 = \mathbf{0}$, and $\mathbf{X}_v$ denotes the original feature vector of node v. The feature aggregations on $\mathbf{h}_v^1$ and $\mathbf{h}_v^2$ are performed separately as well, where either $\mathbf{h}_v^1$ or $\mathbf{h}_v^2$ aggregates the corresponding features from neighboring nodes based on their connection properties, which is detailed in the subsequent section. Such a bi-vector feature setting can address the aforementioned limitation to collaboratively preserve local neighborhood similarities and dissimilarities, and promote long-range information propagation, which, in turn, boosts node representation expressiveness and model discrimination power.

B. Dual-Feature Aggregation

1) Edge Identification: Dual-feature aggregations rely on edge property specified by the labels of connected nodes, which requires a priori knowledge across the whole graph. However, this knowledge is not always available in real-world fraud graphs, as most of nodes are not annotated. Therefore, we need to introduce an edge identifier to discriminate if an edge is homophilic or heterophilic. A traditional way is to apply multi-layer perceptron (MLP) on the representations of its connected nodes to learn edge identification probability. Considering that node representations vary at each aggregation layer, such a learning process would significantly increase model complexity. Here we offer a simplified implementation to reduce training effort. The edge identifier α_{ij} for the edge $e_{v_j v_j}$ reveals the association between nodes v_i and v_j, which can be interpreted, in a simpler way, as a similarity measure between them. In other words, α_{ij} at aggregation layer l can be calculated using a similarity metric:

$$\alpha_{ij} = \mathrm{sim}(\mathbf{h}_{v_i}^{(l-1)}, \mathbf{h}_{v_j}^{(l-1)}) \tag{7}$$

where α_{ij} $(0 \leq \alpha_{ij} \leq 1)$ is equivalent to a prediction score, indicating how possible an edge $e_{v_i v_j}$ is homophilic.

To exclude large weight matrix updates and multiplications, we borrow the idea of residual framework [34] to assemble all weights into a single linear layer $g_{\mathbf{W}}(\cdot)$ with weight matrix $\mathbf{W}$ that maps the original feature matrix $\mathbf{X}$ to a low-dimensional feature matrix, which can be formulated as follows:

$$\mathbf{X}^{(0)} = g_{\mathbf{W}}(\mathbf{X}) \in \mathrm{R}^{d\times k}, \quad k \ll d \tag{8}$$

such that $\mathbf{h}_v (v \in V)$ is initialized as $\mathbf{h}_v = \{\mathbf{X}_v^{(0)}, \mathbf{0}\}$, where $\mathbf{h}_v^1 = \mathbf{X}_v^{(0)}$, $\mathbf{h}_v^2 = \mathbf{0}$. Afterwards, graph layers attend to edge identification and feature aggregation without adding any weighted operations.

2) Neighborhood Aggregation: The normalized adjacency matrix $\tilde{\mathbf{A}}$ is generally considered a low-lass filter in GNNs to retain the commonality of neighboring features [23], [30], while $\mathbf{I} - \tilde{\mathbf{A}}$ provides diversification operation that is considered as high-pass filter to extract neighborhood differences [25]. This inspires us to directly employ the soft prediction of edge identifier α_{ij} $(0 \leq \alpha_{ij} \leq 1)$ instead of hard prediction to perform dual-feature aggregation, which enables α_{ij} and $1 - \alpha_{ij}$ to act like advanced dual filters that control the number of similarities and dissimilarities from each neighbor to be aggregated and accordingly refine node representations. Formally, for a given node v_i, the dual-feature aggregations are implemented as follows:

$$\begin{aligned}\mathbf{h}_{v_i}^{1,(l)} =& \sigma(\mathbf{h}_{v_i}^{1,(l-1)} + \\ & \frac{1}{|\mathcal{N}(v_i)|} \sum_{v_j \in \mathcal{N}(v_i)} \alpha \mathbf{h}_{v_j}^{1,(l-1)} + (1-\alpha)\mathbf{h}_{v_j}^{2,(l-1)})\end{aligned} \tag{9}$$

$$\begin{aligned}\mathbf{h}_{v_i}^{2,(l)} =& \sigma(\mathbf{h}_{v_i}^{2,(l-1)} + \\ & \frac{1}{|\mathcal{N}(v_i)|} \sum_{v_j \in \mathcal{N}(v_i)} (1-\alpha) \mathbf{h}_{v_j}^{1,(l-1)} + \alpha\mathbf{h}_{v_j}^{2,(l-1)})\end{aligned} \tag{10}$$

For node v_i at layer l, $\mathbf{h}_{v_i}^{1,(l)}$ aggregates neighboring features and produces its new similarity feature vector, while $\mathbf{h}_{v_i}^{2,(l)}$ produces its new dissimilarity feature vector, both of which contribute to the new hidden representation. More specifically, such a dual-feature aggregation absorbs the corresponding signals from neighbors through connection property (i.e., if the edge connecting the target node and its neighboring node is homophilic or heterophilic):

- When $\alpha_{ij} \rightarrow 1$, the edge is homophilic; thus more similarities from $\mathbf{h}_{v_j}^{1,(l-1)}$ and less dissimilarities from $\mathbf{h}_{v_j}^{2,(l-1)}$ would flow into $\mathbf{h}_{v_i}^{1,(l)}$ and less similarities from $\mathbf{h}_{v_j}^{1,(l-1)}$ and more dissimilarities from $\mathbf{h}_{v_j}^{2,(l-1)}$ would flow into $\mathbf{h}_{v_i}^{2,(l)}$.
- When $\alpha_{ij} \rightarrow 0$, the edge is heterophilic; then the dual aggregations would be proceeded in the opposite manner. In other words, less similarities from $\mathbf{h}_{v_j}^{1,(l-1)}$ and more dissimilarities from $\mathbf{h}_{v_j}^{2,(l-1)}$ would flow into $\mathbf{h}_{v_i}^{1,(l)}$ and more similarities from $\mathbf{h}_{v_j}^{1,(l-1)}$ and less dissimilarities from $\mathbf{h}_{v_j}^{2,(l-1)}$ would flow into $\mathbf{h}_{v_i}^{2,(l)}$.

By leveraging dual-feature aggregations, features contributing to the node's ground truth can be effectively preserved in $\mathbf{h}_v^1$, while features correlated to other classes can be explicitly saved in $\mathbf{h}_v^2$ to support higher-order information enhancement, where heterophily can barely impact on the outputs from different classes.

C. Oversampling in Node Embedding Space using SMOTE

After L-layer dual-feature aggregations on the initial graph, $\mathbf{H}^{1,(L)}$ and $\mathbf{H}^{2,(L)}$ are further concatenated to form the refined node embeddings $\mathbf{H}$, which is then utilized to synthesize new fraudulent entities to balance class distribution. As the refined node embeddings $\mathbf{H}$ encode both node features and graph structure, edge synthesis becomes unnecessary. This not only simplifies the oversampling operation but also prevents the introduction of noisy node connection information. Due to its popularity and performance in oversampling, we deploy SMOTE as our oversampling algorithm [35]. Given node embeddings $\mathbf{H}$, for a fraud node v_i and a randomly selected fraud node v_j from v_i's nearest neighbors, their embeddings are specified as $\mathbf{h}_{v_i}$ and $\mathbf{h}_{v_j}$. Accordingly, a new fraudulent entity can be synthesized as follows:

$$\mathbf{s} = \mathbf{h}_{v_i} + \delta(\mathbf{h}_{v_j} - \mathbf{h}_{v_i}) \tag{11}$$

where $\mathbf{s}$ is a synthesized node embedding and δ is a random variable ranging from 0 to 1. Since both $\mathbf{h}_{v_i}$ and $\mathbf{h}_{v_j}$ are fraud nodes, $\mathbf{s}$ equalizes the same class distribution. In this way, we can oversample m fraud nodes $\mathbf{S} \in \mathrm{R}^{m\times k}$ that are integrated with initial labeled nodes $\mathbf{H} \in \mathrm{R}^{n\times k}$ for fraud detector training.

Algorithm 1: DOS-GNN: Dual-feature aggregations with over-sampling for class-imbalanced fraud detection on graphs.

```
Input: $G = (V, E, \mathbf{X})$: a fraud graph with feature
       matrix $\mathbf{X}$ and adjacency matrix $\mathbf{A}$; $L$: number
       of layers in the GNN model; $T_{\text{gnn}}, T_{\text{mlp}}$:
       training epochs for GNN and MLP models.
Output: $f$: class-imbalanced fraud detection model.
// GNN model training:
for each epoch $t \leq T_{gnn}$ do
    Initialize $\mathbf{H}^{1,(0)} = \mathbf{X}$ and $\mathbf{H}^{2,(0)} = \mathbf{0}$;
    for each layer $0 < l \leq L$ do
        Calculate $\mathbf{H}^{1,(l)}$ using Eq. (9);
        Calculate $\mathbf{H}^{2,(l)}$ using Eq. (10);
    end
    Calculate $\mathbf{H} = \mathbf{H}^{1,(L)} || \mathbf{H}^{2,(L)}$;
    Calculate $\mathbf{Z}$ on $\mathbf{H}$ using linear layer;
    Calculate $\mathcal{L}_{\text{gnn}}$ from $\mathbf{Z}$ using Eq. (13);
    Update model parameters by minimizing $\mathcal{L}_{\text{gnn}}$;
end
// MLP model training:
Extract $\mathbf{H}$ from the trained GNN model;
Synthesize fraud samples $\mathbf{S}$ from $\mathbf{H}$ using Eq. (11);
Construct $\mathbf{X}'$ by concatenating $\mathbf{S}$ and $\mathbf{H}$;
for each epoch $t \leq T_{mlp}$ do
    Calculate $\mathcal{L}_{\text{mlp}}$ using Eq. (14);
    Update model parameters by minimizing $\mathcal{L}_{\text{mlp}}$;
end
```

D. Loss Optimization

DOS-GNN first trains a GNN model on the provided fraud graph G for dual-feature aggregation to acquire node embeddings facilitating oversampling, and subsequently trains a MLP model on the integrated embeddings $\mathbf{X}'$ for fraud detection, which can be derived by concatenating the initial labeled nodes $\mathbf{H}$ and the synthesized fraud nodes $\mathbf{S}$:

$$\mathbf{X}' = \mathbf{S} || \mathbf{H}, \quad \mathbf{X}' \in \mathrm{R}^{(m+n)\times k} \tag{12}$$

where $||$ represents the concatenation operation. The former training formulates a cross-entropy loss $\mathcal{L}_{\text{gnn}}$ between predictions and ground truth of initially labeled nodes, which can be formulated as follows:

$$\mathcal{L}_{\text{gnn}} = -\frac{1}{n}\sum_{i=1}^{n} \mathbf{y}_i \log \mathbf{Z}_i \tag{13}$$

where $\mathbf{Z}$ is the final output of the GNN model with L layers that can be calculated as $\mathbf{Z} = \text{softmax}(\mathbf{H}^{(L)})$, and n is the number of initially labeled nodes with the label vector $\mathbf{y}$. The latter training leads to another cross-entropy loss $\mathcal{L}_{\text{mlp}}$ between predictions and labels of both initial annotated nodes and synthesized nodes, which is formulated as:

$$\mathcal{L}_{\text{mlp}} = -\frac{1}{n+m}\sum_{i=1}^{n+m} \mathbf{y}'_i \log \mathbf{Z}'_i \tag{14}$$

where $\mathbf{Z}'$ is the final output of the MLP model and $\mathbf{y}'$ is the label vector for both the initial labeled nodes and the synthesized fraud nodes. The respective model weights can be updated by minimizing $\mathcal{L}_{\text{gnn}}$ and $\mathcal{L}_{\text{mlp}}$ using Adam. Algorithm 1 illustrates the full steps of DOS-GNN for class-imbalanced fraud detection.

TABLE I: Statistics of the datasets (Hom. Ratio reports r_h for minority class)

Dataset	#Nodes	#Edges	#Features	Imb. Ratio	Hom. Ratio
Amazon	11,944	4,398,392	25	0.07	0.04
YelpChi	45,954	3,846,979	32	0.17	0.10
T-finance	39,357	42,445,086	10	0.05	0.30
Elliptic	203,769	234,355	166	0.02	0.12

IV. Experimental Results and Analysis

In this section, we evaluate the performance of DOS-GNN on four real-world fraud detection datasets to answer the following research questions:

- **RQ1:** How does DOS-GNN perform compared with state-of-the-art GNN models in fraud detection tasks?
- **RQ2:** How do different model parameters impact on the performance of DOS-GNN?
- **RQ3:** How do different components contribute to the performance of DOS-GNN?

A. Experiment Setup

Datasets. We evaluate DOS-GNN on four real-world fraud detection datasets with extremely low imbalance ratios and low homophily ratios for the minority class:

- **YelpChi** [36]: This dataset identifies anomalous reviews on Yelp.com that promote or demote products or businesses. It features a graph with three types of edges: R-U-R, R-S-R, and R-T-R.
- **Amazon** [37]: This dataset detects users who are compensated to write counterfeit reviews for musical instruments on Amazon.com. It includes three types of relations: U-P-U, U-S-U, and U-V-U.
- **T-finance** [38]: This dataset identifies anomalous user accounts within transaction networks. Node features include registration duration, login activities, and interaction rates, while edges signify transactional account interactions.
- **Elliptic** [39]: The dataset categorizes Bitcoin transactions into legal entities (e.g., wallet providers and miners) and illegal entities (e.g., scams, malware, and terrorists). It is structured as a graph where nodes are transactions and edges represent the flow of Bitcoin.

We use 40%-30%-30% data-split across all four datasets to train, validate, and test models, respectively. The data statistics are summarized in Table I. In the experiments, all relationships in YelpChi and Amazon datasets are considered the same to align with homogeneous GNNs. The timestamp in the Elliptic dataset is excluded from data splitting and removed before being fed into the model.

TABLE II: Comparison of different Fraud Detection methods (%) on benchmark datasets. Some GNN models are highlighted: **bold** statistics denote the best results. OOM indicates that the machine runs out of memory before the algorithm terminates.

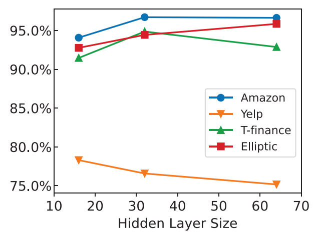
Dataset	**Amazon**		**YelpChi**		**T-Finance**		**Elliptic**	
	AUC	**F1**	**AUC**	**F1**	**AUC**	**F1**	**AUC**	**F1**
GCN	74.34	67.47	52.47	54.31	64.43	70.74	90.47	83.13
GAT	75.16	83.18	56.24	54.64	73.00	53.86	63.87	47.43
GIN	80.56	69.26	74.09	62.85	80.02	65.23	93.71	88.78
GraphSAGE	75.27	74.17	54.00	65.49	67.12	52.71	94.35	89.04
JKGCN	89.63	72.52	80.74	59.75	93.92	85.39	93.60	85.98
GPRGNN	92.79	85.47	81.03	65.46	94.25	87.73	94.49	89.71
GraphENS	80.01	52.81	60.12	47.63	OOM	OOM	83.43	51.35
GraphSMOTE	90.79	88.36	76.74	65.22	OOM	OOM	91.46	86.81
Graph-Consis	87.41	75.12	69.83	58.70	91.42	73.46	93.93	89.28
PC-GNN	95.86	89.56	79.87	63.00	91.23	63.18	94.39	91.02
Care-GNN	89.73	86.39	75.70	63.32	92.16	77.55	94.12	90.55
DOS-GNN	**96.55**	**92.10**	**81.15**	**70.46**	**96.01**	**88.53**	**96.32**	**92.75**

Baselines. We select 11 different models as our baselines, which include four conventional GNNs, two heterophily-wise GNNs, and five advanced models for imbalanced graphs or fraud detection, which are briefly introduced as follows:

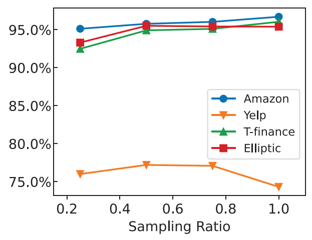
- **GCN** [13]: The vanilla GCN model applies convolution operation to graph data.
- **GAT** [27]: GAT introduces the attention mechanism to GNN for feature aggregation.
- **GIN** [40]: GIN uses a more complicated aggregation method to make GNN more powerful.
- **GraphSAGE** [12]: GraphSAGE samples from node neighbors and aggregates their embeddings.
- **JKGCN** [41]: JKGCN combines learned embeddings at each model layer to generate the final node embeddings.
- **GPRGNN** [42]: GPRGNN combines generalized PageRank techniques with GNNs to overcome heterophily.
- **GraphENS** [18]: GraphENS is an augmentation method that synthesizes ego networks for generated minor nodes.
- **GraphSMOTE** [17]: GraphSMOTE synthesizes more nodes and related edges for minority class nodes.
- **Graph-Consis** [43]: Graph-Consis utilizes context embedding, neighbor sampling, and relation attention for fraud detection tasks.
- **PC-GNN** [44]: PC-GNN uses samplers to construct subgraphs and sample from neighborhood for aggregation.
- **Care-GNN** [14]: Care-GNN uses a special mechanism to select informative neighbors for aggregation.

Some of the reported results in this paper are taken from their original papers.

Implementation Details. The number of aggregation layers L for the feature extraction model is set to 2 for DOS-GNN. All models are trained for 2,000 epochs with a patience of 200 using Adam optimizer with learning rate $lr = 0.01$ and $5e-4$ L2 regularization. sim$(\cdot)$ used in edge identification is cosine similarity and the size of hidden units is explored within $k \in \{16, 32, 64\}$. AUC (Area Under the Receiver Operating Characteristic Curve) and F1-Macro are the primary evaluation metrics to provide insight into a model's effectiveness on class-imbalanced fraud detection. The MLP model consists of 2



(a)



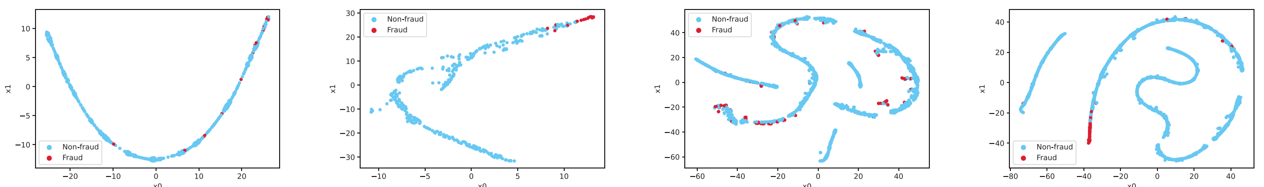
(b)

Fig. 3: Impact of parameters on DOS-GNN: (a) AUC of DOS-GNN using different hidden layer sizes; (b) AUC of DOS-GNN using different oversampling ratios.

layers, and the oversampling ratio is searched within the range of $\{0.25, 0.5, 0.75, 1.0\}$.

B. Comparison with State-of-the-Art Baselines

In this section, we would like to answer **RQ1** to evaluate the effectiveness of DOS-GNN for class-imbalanced fraud detection on graphs by comparing our model with 11 selected baselines over four different public fraud datasets. The comparative results are reported in Table II. From Table II, traditional GNN models exhibit poor performance in class-imbalanced fraud graphs. In contrast, advanced models designed for addressing heterophily, imbalanced data, or fraud detection show significant improvement in detection performance, due to their enhancement in handling heterophilic neighborhood and class imbalance, preventing fraud nodes from being overshadowed by a large number of legitimate nodes. Even so, our proposed DOS-GNN still manages to advance the state-of-the-art performance to a higher level. Compared to the best results of traditional GNN models, DOS-GNN improves the AUC and F1 by 15.99% and 9.08% for Amazon, 4.95% and 3.94% for YelpChi, 15.99% and 17.79% for T-Finance, and 1.97% and 3.71% for Elliptic. Compared to the best results of models for heterophily, imbalanced data, or fraud detection, DOS-GNN further improves the AUC and F1 by 1.37% and 2.83% for Amazon, 0.14% and 7.63% for



(a) (b) (c) (d)

Fig. 4: Visualization of node embeddings: (a) Amazon processed by GCN, (b) Amazon processed by DOS-GNN, (c) T-finance processed by GCN, (d) T-finance processed by DOS-GNN.

YelpChi, 1.86% and 0.91% for T-Finance, and 2.04% and 1.90% for Elliptic.

In summary, DOS-GNN achieves state-of-the-art performance across all four public benchmarks and outperforms the leading GNN models. This comparative study confirms that the combination of local feature separation and dual feature aggregation can extract, preserve, and propagate similar and dissimilar features effectively, which thus boosts node embeddings and model discrimination capability. The refined node embeddings can further facilitate oversampling of fraudulent entities to create a balanced class distribution, which, in turn, enables the training of an unbiased detector to effectively identify frauds across various types of networks.

C. Parameter Evaluation

In this section, we further analyze the impacts of parameters to answer **RQ2**. The performance of DOS-GNN can be potentially affected by the hidden layer size and oversampling ratios. The experimental results are reported in Fig. 3. It can be observed that different parameters contribute to slightly different results. More specifically, DOS-GNN achieves the best performance with hidden layer sizes 32, 16, 32, and 64 on Amazon, YelpChi, T-finance, and Elliptic, respectively. This is not surprising given that the initial feature vector sizes for YelpChi, T-finance and Amazon are 32, 10, and 25 respectively. Providing a hidden layer size larger than the initial feature length is not likely to generate more informative node embeddings, but might have increased the chance of overfitting. The best sampling ratios for DOS-GNN are 1.0, 0.5, 0.75, and 1.0 for Amazon, YelpChi, T-finance, and Elliptic. Overall, these fluctuations are relatively small, which implies that our model exhibits a high degree of stability across various parameter settings.

D. Ablation Study

In response to **RQ3**, we set up the ablation study to investigate how different components contribute to the performance of our model. We investigate the two components in our model design: due-feature aggregation and oversampling. As illustrated in Table III, both components contribute to the performance of DOS-GNN. Among these two components, the dual-feature aggregation contributes the most to DOS-GNN. Even without oversampling on the learned node embeddings, the performance of standalone dual-feature aggregation model achieves a performance boost of over 20% on Amazon, YelpChi, and T-finance from GCN. Utilizing oversampling provides further improvements for both GCN and DFA-based GNN, while the improvement range of $[2\%, 7\%]$ on all datasets for DOS-GNN is more significant. This outcome aligns with our expectations. The core functionality of DOS-GNN relies on the collaboration of dual-feature aggregation and oversampling to alleviate the impact of class imbalance on graphs.

TABLE III: Evaluation on model components in terms of AUC (%): Dual-Feature Aggregation (DFA) and Oversampling (OS)

GCN	DFA	OS	Amazon	YelpChi	T-Finance	Elliptic
✓			74.34	52.47	64.43	90.47
✓		✓	83.28	77.25	87.61	91.56
	✓		91.59	71.89	94.13	93.35
	✓	✓	**96.55**	**81.15**	**96.01**	**95.43**

E. Case Study

To validate our claim that DOS-GNN provides more distinguishable node embeddings in graphs on fraud datasets, we present a brief case study to showcase the difference between embeddings generated by GCN and DOS-GNN. Due to page limit, here we only present results on two datasets, Amazon and T-finance. We map node embeddings into a two-dimensional space using t-SNE, and the resulting embeddings are visualized in Fig. 4. For Amazon, GCN fails to generate clear clusters between fraud and non-fraud points and suffers from fuzzy boundaries (Fig. 4(a)); in contrast, DOS-GNN exhibits better-distinguished boundaries with higher cohesion (Fig. 4(b)). For T-finance, GCN again fails to generate any associations with fraud points being scattered among legitimate points (Fig. 4(c)), while DOS-GNN forms clear boundaries and redistributes the nodes, making them further apart (Fig. 4(d)). These observations reaffirm the effectiveness of DOS-GNN in learning more distinct node embeddings for synthesized node generation and fraud detection.

V. Conclusion

In this paper, we introduce a new model DOS-GNN for fraud detection. DOS-GNN employs feature separation and dual feature aggregation guided by edge identification to better preserve and propagate both similarities and dissimilarities for each node and refine node embeddings, and oversampling on the refined node embeddings to further mitigate the effect of data imbalance. Evaluation through extensive experiments demonstrates that our model achieves state-of-the-art performance, which affirms its effectiveness in class-imbalanced node classification, superiority over baselines, and practical significance in handling fraud detection tasks on graphs.

Acknowledgments

L. Chen's work is partially supported by the NSF under grant CNS-2245968. The authors would also like to thank the reviewers for their valuable feedback.

References

[1] B. Hooi, K. Shin, H. A. Song, A. Beutel, N. Shah, and C. Faloutsos, "Graph-based fraud detection in the face of camouflage," *TKDD*, vol. 11, no. 4, pp. 1–26, 2017.

[2] L. Chen, Y. Fan, and Y. Ye, "Adversarial reprogramming of pretrained neural networks for fraud detection," in *CIKM*, pp. 2935–2939, 2021.

[3] W. Lin, L. Sun, Q. Zhong, C. Liu, J. Feng, X. Ao, and H. Yang, "Online credit payment fraud detection via structure-aware hierarchical recurrent neural network.," in *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 3670–3676, 2021.

[4] R. Bauder, T. M. Khoshgoftaar, and N. Seliya, "A survey on the state of healthcare upcoding fraud analysis and detection," *Health Services and Outcomes Research Methodology*, vol. 17, pp. 31–55, 2017.

[5] J. Wang, R. Wen, C. Wu, Y. Huang, and J. Xiong, "Fdgars: Fraudster detection via graph convolutional networks in online app review system," in *Proceedings of the ACM Web Conference (WWW)*, 2019.

[6] A. Li, Z. Qin, R. Liu, Y. Yang, and D. Li, "Spam review detection with graph convolutional networks," in *CIKM*, pp. 2703–2711, 2019.

[7] S. Feng, Z. Tan, R. Li, and M. Luo, "Heterogeneity-aware twitter bot detection with relational graph transformers," in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 3977–3985, 2022.

[8] T. Pourhabibi, K.-L. Ong, B. H. Kam, and Y. L. Boo, "Fraud detection: A systematic literature review of graph-based anomaly detection approaches," *Decision Support Systems*, vol. 133, p. 113303, 2020.

[9] Z. Qin, Y. Liu, Q. He, and X. Ao, "Explainable graph-based fraud detection via neural meta-graph search," in *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM)*, pp. 4414–4418, 2022.

[10] L. Chen, X. Li, and D. Wu, "Enhancing robustness of graph convolutional networks via dropping graph connections," in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2020, Ghent, Belgium, September 14–18, 2020, Proceedings, Part III*, pp. 412–428, Springer, 2021.

[11] Q. Li, X. Li, L. Chen, and D. Wu, "Distilling knowledge on text graph for social media attribute inference," in *SIGIR*, pp. 2024–2028, 2022.

[12] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.

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

[14] Y. Dou, Z. Liu, L. Sun, Y. Deng, H. Peng, and P. S. Yu, "Enhancing graph neural network-based fraud detectors against camouflaged fraudsters," in *CIKM*, 2020.

[15] F. Shi, Y. Cao, Y. Shang, and et al., "H2-fdetector: A gnn-based fraud detector with homophilic and heterophilic connections," in *Proceedings of the ACM Web Conference (WWW)*, pp. 1486–1494, 2022.

[16] B. Ashmore and L. Chen, "Hover: Homophilic oversampling via edge removal for class-imbalanced bot detection on graphs," in *CIKM*, pp. 3728–3732, 2023.

[17] T. Zhao, X. Zhang, and S. Wang, "Graphsmote: Imbalanced node classification on graphs with graph neural networks," in *WSDM*, pp. 833–841, 2021.

[18] J. Park, J. Song, and E. Yang, "GraphENS: Neighbor-aware ego network synthesis for class-imbalanced node classification," in *ICLR*, 2022.

[19] X. Zheng, Y. Liu, S. Pan, M. Zhang, D. Jin, and P. S. Yu, "Graph neural networks for graphs with heterophily: A survey," *arXiv preprint arXiv:2202.07082*, 2022.

[20] Y. Yan, M. Hashemi, and et al., "Two sides of the same coin: Heterophily and oversmoothing in graph convolutional neural networks," in *IEEE International Conference on Data Mining (ICDM)*, pp. 1287–1292, 2022.

[21] J. Zhu, R. A. Rossi, A. Rao, T. Mai, N. Lipka, N. K. Ahmed, and D. Koutra, "Graph neural networks with heterophily," in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 11168–11176, 2021.

[22] J. Zhu, Y. Yan, L. Zhao, and et al., "Beyond homophily in graph neural networks: Current limitations and effective designs," *Advances in neural information processing systems (NeurIPS)*, vol. 33, pp. 7793–7804, 2020.

[23] D. Bo, X. Wang, C. Shi, and H. Shen, "Beyond low-frequency information in graph convolutional networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 3950–3957, 2021.

[24] L. Du, X. Shi, Q. Fu, X. Ma, H. Liu, S. Han, and D. Zhang, "Gbk-gnn: Gated bi-kernel graph neural networks for modeling both homophily and heterophily," in *Proceedings of the ACM Web Conference 2022*, pp. 1550–1558, 2022.

[25] S. Luan, C. Hua, Q. Lu, and et al., "Is heterophily a real nightmare for graph neural networks to do node classification?," *arXiv preprint arXiv:2109.05641*, 2021.

[26] Q. Li, L. Chen, S. Jing, and D. Wu, "Pseudo-labeling with graph active learning for few-shot node classification," in *ICDM*, pp. 1115–1120, 2023.

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

[28] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 1, pp. 4–24, 2020.

[29] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, "Simplifying graph convolutional networks," in *International conference on machine learning (ICML)*, pp. 6861–6871, 2019.

[30] H. Nt and T. Maehara, "Revisiting graph neural networks: All we have is low-pass filters," *arXiv preprint arXiv:1905.09550*, 2019.

[31] V. N. Ekambaram, *Graph-structured data viewed through a Fourier lens*. University of California, Berkeley, 2014.

[32] H. Pei, B. Wei, K. C.-C. Chang, Y. Lei, and B. Yang, "Geom-gcn: Geometric graph convolutional networks," *arXiv preprint arXiv:2002.05287*, 2020.

[33] U. Alon and E. Yahav, "On the bottleneck of graph neural networks and its practical implications," *International Conference on Learning Representations (ICLR)*, 2021.

[34] X. Liu, J. Ding, W. Jin, H. Xu, Y. Ma, Z. Liu, and J. Tang, "Graph neural networks with adaptive residual," *Advances in neural information processing systems (NeurIPS)*, vol. 34, pp. 9720–9733, 2021.

[35] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *J. Artif. Int. Res.*, vol. 16, no. 1, p. 321–357, 2002.

[36] S. Rayana and L. Akoglu, "Collective opinion spam detection: Bridging review networks and metadata," in *KDD*, pp. 985–994, 2015.

[37] J. J. McAuley and J. Leskovec, "From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews," in *Proceedings of the ACM Web Conference (WWW)*, pp. 897–908, 2013.

[38] J. Tang, J. Li, Z. Gao, and J. Li, "Rethinking graph neural networks for anomaly detection," in *International conference on machine learning (ICML)*, pp. 21076–21089, 2022.

[39] M. Weber, G. Domeniconi, and et al., "Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics," *arXiv preprint arXiv:1908.02591*, 2019.

[40] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?," *arXiv preprint arXiv:1810.00826*, 2018.

[41] K. Xu, C. Li, Y. Tian, T. Sonobe, K.-i. Kawarabayashi, and S. Jegelka, "Representation learning on graphs with jumping knowledge networks," in *International conference on machine learning*, pp. 5453–5462, PMLR, 2018.

[42] E. Chien, J. Peng, P. Li, and O. Milenkovic, "Adaptive universal generalized pagerank graph neural network," in *International Conference on Learning Representations*, 2021.

[43] Z. Liu, Y. Dou, P. S. Yu, Y. Deng, and H. Peng, "Alleviating the inconsistency problem of applying graph neural network to fraud detection," in *SIGIR*, pp. 1569–1572, 2020.

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