DiG-In-GNN: Discriminative Feature Guided GNN-Based Fraud Detector against Inconsistencies in Multi-Relation Fraud Graph

Jinghui Zhang, Zhengjia Xu*, Dingyang Lv*, Zhan Shi, Dian Shen, Jiahui Jin, Fang Dong

Southeast University
{jhzhang, xzhengjia, dylv, shizhan, dshen, jjin, fdong}@seu.edu.cn

Abstract

Fraud detection on multi-relation graphs aims to identify fraudsters in graphs. Graph Neural Network (GNN) models leverage graph structures to pass messages from neighbors to the target nodes, thereby enriching the representations of those target nodes. However, feature and structural inconsistency in the graph, owing to fraudsters' camouflage behaviors, diminish the suspiciousness of fraud nodes which hinders the effectiveness of GNN-based models. In this work, we propose DiG-In-GNN, **Di**scriminative Feature **G**uided **GNN** against **In**consistency, to *dig into* graphs for fraudsters. Specifically, we use multi-scale contrastive learning from the perspective of the neighborhood subgraph where the target node is located to generate guidance nodes to cope with the feature inconsistency. Then, guided by the guidance nodes, we conduct fine-grained neighbor selection through reinforcement learning for each neighbor node to precisely filter nodes that can enhance the message passing and therefore alleviate structural inconsistency. Finally, the two modules are integrated together to obtain discriminable representations of the nodes. Experiments on three fraud detection datasets demonstrate the superiority of the proposed method DiG-In-GNN, which obtains up to 20.73% improvement over previous state-of-the-art methods. Our code can be found at https://github.com/GraphBerry/DiG-In-GNN.

Introduction

The proliferation of the Internet has led to an increase in fraudulent activities in various real-world scenarios (Jiang, Cui, and Faloutsos 2016), such as review fraud, fake accounts, and malicious websites. In recent years, graph-based fraud detection techniques have garnered considerable research interest. By modeling real-world entities as nodes and their interactions as edges, fraudulent entities can be identified through abundant graph information. For instance, the classification of fraud comments in a *Review-User-Review* graph (Dou et al. 2020; Wang et al. 2023), the detection of fake accounts in an *Account* graph (Yuan et al. 2019; Liu et al. 2018), filtering out malicious websites with a *URL* graph (Zhang et al. 2019; Tan et al. 2020).

However, fraudsters often resort to various techniques to camouflage and alleviate suspicion, causing various inconsistencies in the graph, limiting the effectiveness of fraud detectors (Dou et al. 2020; Zhang et al. 2021). In Figure 1, a multi-relation graph depicts a realistic e-commerce scenario where users give rating stars and reviews on e-commerce websites. In this scenario, fraudsters typically use two sophisticated camouflage strategies to avoid detection. Feature camouflage is the first, where fraudsters adopt the linguistic style of benign reviews, and may even leverage deep language generation models to make their fraudulent reviews appear more authentic. This leads to a mismatch between node feature patterns and their labels, termed **feature inconsistency**. The second strategy, relation camouflage, involves fraudsters employing various identities to post reviews on various items. They cleverly connect themselves within the graph to benign users, creating various camouflage connections. In this way, fraudsters hide among benign nodes, resulting in **structural inconsistency**.

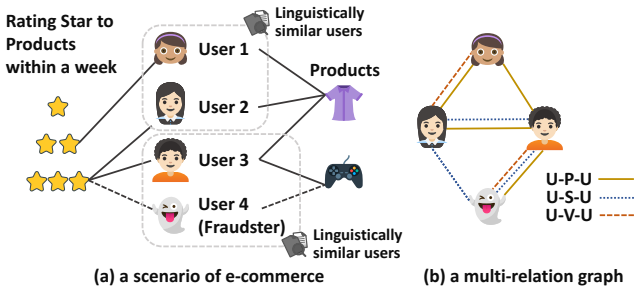

Figure 1: (a) Users rate and review products, it can be modeled by a multi-relation graph. (b) In the multi-relation graph, there are three edge types: U-P-U connects users reviewing at least one product; U-S-U connects users having at least one identical star rating within a week; U-V-U connects linguistically similar users.

To identify suspicious entities in such multi-relation graphs, graph-based algorithms are typically used. In recent years, Graph Neural Networks (GNNs) have garnered significant attention as an effective approach to perform deep learning on graph data, and have become the de facto solution for graph-based fraud detection (Ma et al. 2021). GNNs employ a message passing mechanism to recursively aggregate neighbor information into each node's representation.

*Corresponding authors

Challenges. While promising, the GNN-based fraud detectors face two challenges due to feature and structural inconsistency, which pose obstacles to GNN-based methods relying on message passing.

Node representations become confusing due to feature inconsistency. The message passing mechanism enhances the node representations by aggregating neighbor nodes. This relies on the premise that the raw features effectively capture the class semantic information. However, the lack of distinguishability of raw features undermines the premise. Consider a fraud target node with a distinctive feature; after message passing, it may lose discriminability because its fraud neighbors have inconsistent features. In this situation, GNNs produce negative results. Moreover, ambiguous node features can seriously affect the decision of neighbor selection. Therefore, nodes with different labels need more distinctive features.

Structural inconsistency necessitates meticulous neighbor selection. Aggregating a large amount of benign node information can obscure the suspiciousness of fraud nodes, highlighting the importance of more nuanced neighbor selection. Existing neighbor selection approaches rely on weighting neighbor nodes based on similarity (Peng et al. 2021; Liu et al. 2021), assuming that the raw features provide sufficient semantic class information. However, this assumption is compromised in the presence of feature inconsistency. In our work, we emphasize the need to consider the impact of feature inconsistency when addressing the challenge of structural inconsistency through neighbor selection.

Current Attempts. We categorize the existing methods as similarity-based neighbor selection and heterophily-aware. Similarity-based methods are widely studied to mitigate structure inconsistency. CARE-GNN (Dou et al. 2020) and PC-GNN (Liu et al. 2021) offer a similarity-based neighbor selection strategy that establishes a threshold for filtering. RioGNN (Peng et al. 2021) applies a reinforced relation-aware neighbor selection to choose the most similar neighbors of a targeting node. C-FATH (Wang et al. 2021) proposes a community-based framework for large-scale heterogeneous graphs and adopting similarity-based sampling for 3-hop neighbors. While these efforts have made excellent progress in fraud detection, they ignore feature inconsistency. On the other hand, H^2-FDetector (Shi et al. 2022) and GHRN (Gao et al. 2023) explore from the perspective of heterophily. H^2-FDetector aggregates all neighbors but employs different aggregation weights for homophilic and heterophilic edges. GHRN utilizes a high-pass filter to prune inter-class edges, reducing the heterophily degree of the graph, thus mitigating the impact of structural inconsistency. Nevertheless, these methods still lack consideration about the feature inconsistency. GTAN (Xiang et al. 2023) employs the risk embeddings to enhance node feature. However, constructing risk embedding requires the use of raw features to obtain pseudo labels, so the performance is still limited by the feature inconsistency of the raw features.

Our Contributions. To address the challenges mentioned above, we propose DiG-In-GNN (DiG-In), **Di**scriminative Feature **G**uided **GNN** against **In**consistency, to *dig in*to graphs for fraudsters. Firstly, we utilize a multi-scale contrastive learning model to generate guidance nodes which have discriminative features, enabling better distinguishability among different classes. Secondly, we design a learnable neighbor selection strategy that carefully selects neighbors guided by the guidance nodes to enhance message passing. The following are the main contributions of our work:

- We train a relation-wise guidance node generation model with multi-scale contrastive learning including context- and local-level, to tackle feature inconsistency, which also serves as a crucial foundation for enhancing neighbor selection strategies. This generates guidance nodes with discriminative feature distributions for different classes, thus addressing the feature inconsistency.
- We propose a node-wise neighbor selection strategy based on Reinforcement Learning (RL). Structural inconsistency can be effectively overcome by incorporating feature-consistent guidance nodes and employing a RL-based neighbor selector. This enables message passing to aggregate more beneficial neighbors to enhance the final representations for the detection task.
- The proposed DiG-In-GNN is evaluated on three public datasets. Our method achieves an improvement of up to 20.73% compared to previous state-of-the-art graph-based fraud detectors.

Problem Formulation

In this section, we define multi-relation graph and neighborhood subgraph, and introduce graph-based fraud detection.

Multi-Relation Graph. A graph can be formulated as $\mathcal{G} = \{\mathcal{V}, \mathcal{X}, \mathcal{E}, \mathcal{R}, \mathcal{Y}\}$, where $\mathcal{V} = \{v_1, v_2, ..., v_{|\mathcal{V}|}\}$ is a set of nodes. For each node $v_i \in \mathcal{V}$, there is a label $y_i \in \mathcal{Y}$. $|\mathcal{V}|$ is the number of nodes, $\mathcal{X} = \{x_1, x_2, ..., x_{|\mathcal{V}|}\} \in \mathbb{R}^{|\mathcal{V}| \times d}$ is a set of node features. $\mathcal{R} = \{1, 2, ..., R\}$ represents different relation types, $R = |\mathcal{R}|$. $\mathcal{E}$ is the edge set. Nodes can be connected by edges of various relation types.

Neighborhood Subgraph. Given a graph $g_v^r = \{v, \mathcal{N}_v^r, \mathcal{E}_v^r, t_v^r, r, y\}$, $v \in \mathcal{V}$, $r \in \mathcal{R}$, and $\mathcal{N}_v^r$ is a set composed of one-hop neighbors of v under relation r, $\mathcal{E}_v^r$ is the edge set of g_v^r. In our work, t_v^r, named guidance node, connects v and $u \in \mathcal{N}_v^r$ in g_v^r, and y is the label of v. We call g_v^r a neighborhood subgraph of v. Each neighborhood subgraph can be uniquely determined by v and r.

Graph-Based Fraud Detection. The objective of the task is to determine whether a target node in a graph is a fraudster. Each node is labeled as either a fraud node or a benign node. Therefore, our graph-based fraud detection is a binary node classification problem.

Methodology

This section details our proposed DiG-In-GNN thoroughly. The overall structure of the framework is shown in Figure 2. As depicted, it consists of three parts: guidance node generation (❶), neighbor selection (❷), and neighbor aggregation (❸). For guidance node generation, we utilize multi-scale contrastive learning, combining both context- and local-level, to generate a guidance node for each neighborhood subgraph. Then, we conduct RL-based neighbor selection to determine the aggregation of neighbor nodes.

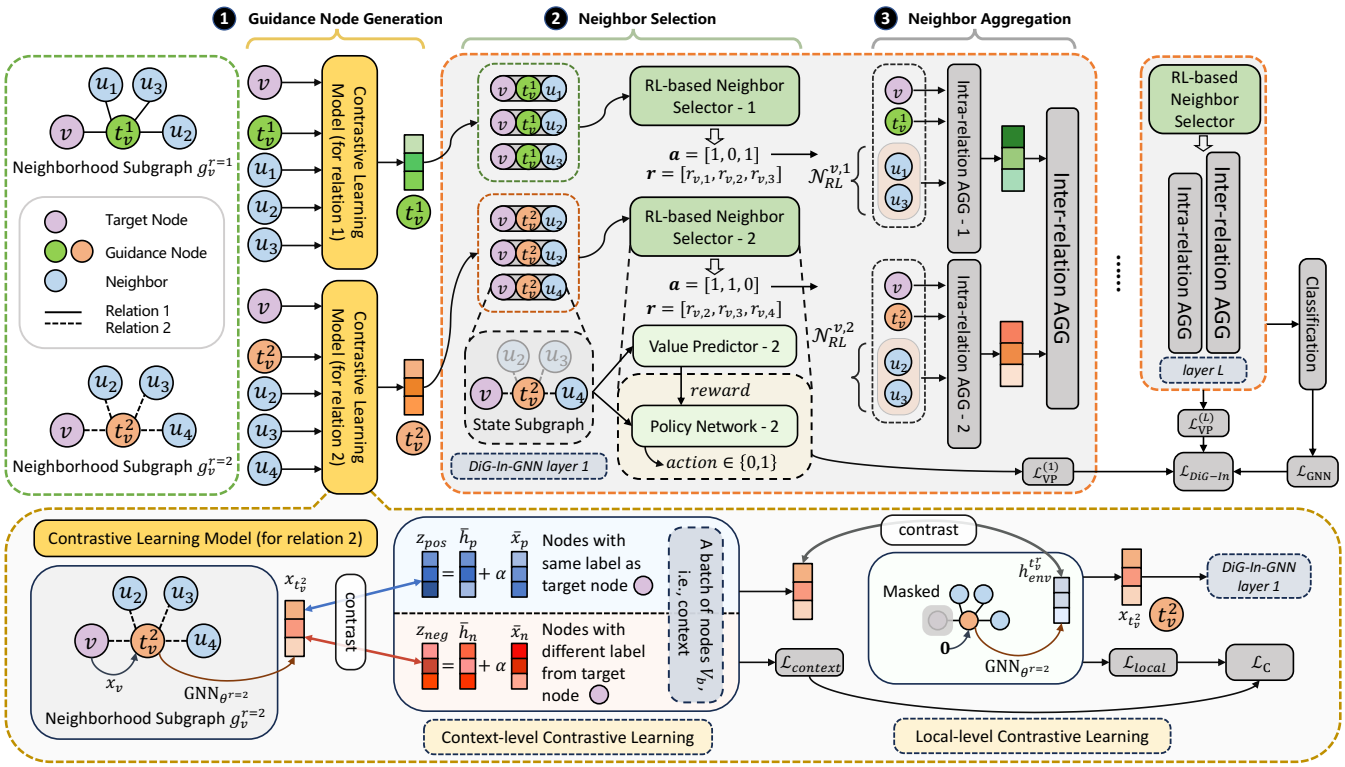

Figure 2: The overall architecture of DiG-In-GNN. The number of relations is set to 2 for simplicity.

Guidance Node Generation

To address feature inconsistency, we introduce guidance nodes in this work. The guidance node features are generated by multi-scale contrastive learning, which better captures the semantic information of the node classes, compared to the raw node features. The context- (node vs. node) and local-level (node vs. environment) contrastive learning tasks (Figure 2 ❶) are designed to capture the distribution pattern of nodes in each neighborhood subgraph at different scales, thereby generating the guidance nodes. Before the details, simply speaking, at the context level, we compare fraud nodes and benign nodes within a batch, and at the local level, we compare fraud nodes and their neighborhood environment within the neighborhood subgraphs. We treat each relation type separately by applying three sets of neural network parameters.

Context-level Contrastive Learning Task. This module is specifically designed to learn the distribution pattern in different neighborhood subgraphs at a global scale. Given a batch of nodes $\mathcal{V}_b$, for a target node $v \in \mathcal{V}_b$ and relation r, a guidance node t_v^r is created, initialized with the corresponding target node v's original feature x_v. L-layer GNN_{θ^r} is then utilized to generate feature $x_{t_v^r}$ for guidance node t_v^r from the neighborhood subgraph $g_v^r = \{v, \mathcal{N}_v^r, \mathcal{E}_v^r, t_v^r, r, y\}$. This process can be defined as Equation 1. $h_{v,r}^{(l)}$ is the representation of v at layer l, with $h_{v,r}^{(1)}$ initialized by the raw features. AGG signifies node aggregation, and θ^r denotes parameters for relation type $r \in \mathcal{R}$.

$$
\begin{aligned}
z_{v,r}^{(l)} &= h_{v,r}^{(l-1)} \parallel \mathrm{AGG}\left(\{h_{u,r}^{(l-1)}, u \in \mathcal{N}_v^r \cup \{v\}\}\right) \\
h_{v,r}^{(l)} &= \mathrm{MLP}_{\theta^r}\left(z_{v,r}^{(l)}\right) \\
x_{t_v^r} &= h_{v,r}^{(L)}
\end{aligned}
\tag{1}
$$

To create positive context-level contrastive samples, the generated feature and the raw feature are combined as follows. $\overline{x}_p^{v,r}$ is the average of the original features of all nodes in $\mathcal{V}_b$ with the same label as v, and $\overline{h}_p^{v,r}$ is the average of the corresponding guidance nodes' generated features. Then, the positive sample $z_{pos}^{t_v^r}$ for $x_{t_v^r}$ is defined as:

$$
\begin{aligned}
\overline{x}_p^{v,r} &= \mathrm{MEAN}\left(\{x_u \mid u \in V_b, y_u = y_v\}\right) \\
\overline{h}_p^{v,r} &= \mathrm{MEAN}\left(\{x_{t_u^r} \mid u \in V_b, y_u = y_v\}\right) \\
z_{pos}^{t_v^r} &= \overline{h}_p^{v,r} + \alpha \overline{x}_p^{v,r}
\end{aligned}
\tag{2}
$$

where α is a hyper-parameter. Similarly, The negative contrastive sample for $x_{t_v^r}$ can be defined as:

$$
z_{neg}^{t_v^r} = \overline{h}_n^{v,r} + \alpha \overline{x}_n^{v,r} \tag{3}
$$

In this context-level task, the objective is to amplify the dissimilarity between the generated guidance node feature $h_{t_v^r}$ and its negative contrastive sample, while enhancing the similarity with the positive sample. To quantify the similarity between $h_{t_v^r}$ and the contrastive samples, we employ a

bilinear layer to calculate the similarity score.

$$p_v^r = \text{Bilinear}\left(z_{pos}^{t_v^r}, x_{t_v^r}\right) = \sigma\left((z_{pos}^{t_v^r})^T W_{ctx}^r x_{t_v^r}\right) \quad (4)$$

$$n_v^r = \text{Bilinear}\left(z_{neg}^{t_v^r}, x_{t_v^r}\right) = \sigma\left((z_{neg}^{t_v^r})^T W_{ctx}^r x_{t_v^r}\right) \quad (5)$$

where W_{ctx}^r is a learnable weight matrix, r denotes the relation type since each relation is processed separately in our method. The loss function for the context-level contrastive learning task is as:

$$s_{ctx}^{(i)} = \begin{cases} 1, i \le |\mathcal{V}_b| \\ 0, i \ge |\mathcal{V}_b| + 1 \end{cases}, \hat{s}_{ctx}^{r(i)} = \begin{cases} p_i^r, i \le |\mathcal{V}_b| \\ n_{(i-|\mathcal{V}_b|)}^r, i \ge |\mathcal{V}_b| + 1 \end{cases}$$

$$\mathcal{L}_{ctx}^r = \text{BCE}(S_{ctx}, \widehat{S}_{ctx}^r) = \frac{1}{2|\mathcal{V}_b|} \sum_{i=1}^{2|\mathcal{V}_b|} \text{BCE}(s_{ctx}^{(i)}, \hat{s}_{ctx}^{r(i)}) \quad (6)$$

where the binary cross-entropy (BCE) loss function is widely used for binary classification. Here it can be defined as $\text{BCE}(s_1, s_2) = -\left[s_1 \log s_2 + (1 - s_1) \log (1 - s_2)\right]$.

Local-level Contrastive Learning Task. In reality, compared to benign behavior, fraud behavior is rare, resulting in benign nodes being the majority in graph data (the average probability of benign nodes appearing in the neighbors of fraud nodes is 75.73% according to our statistics). Therefore, it can be assumed that benign nodes primarily provide environmental information during message passing. Thus, the more distinct guidance nodes are from their local surroundings, the more discernible information benign neighbors can convey to fraud nodes, resulting in generated features of the guidance nodes capable of expressing fraudulent semantic information. Inspired by (Jin et al. 2021), we design a local-level contrastive learning task to make guidance nodes aware of neighbor nodes' distribution pattern.

The GNN employed for acquiring environmental information utilizes the same parameter θ^r as the context-level. Specifically, we mask the target node v in the neighborhood subgraph g_v^r and temporarily replace the guidance node with a zero vector: $x_{t_v^r}^{tmp} \leftarrow \mathbf{0}$. This process yields a subgraph $g_v^{r,env} = \{\mathcal{N}_v^r, \mathcal{E}_v^r, \mathbf{0}, r, y\}$, which is then fed into the model GNN_{θ^r} to generate contrastive sample $h_{t_v^r}^{env}$ for t_v^r, providing local environmental information:

$$\begin{aligned} z_{t_v^r}^{tmp(l)} &= h_{t_v^r}^{tmp^{(l-1)}} \parallel \text{AGG}\left(\{h_{u,r}^{tmp^{(l-1)}}, u \in \mathcal{N}_v^r\}\right) \\ h_{t_v^r}^{tmp^{(l)}} &= \text{MLP}_{\theta^r}\left(z_{t_v^r}^{tmp(l)}\right) \\ h_{t_v^r}^{env} &= h_{t_v^r}^{tmp(L)} \end{aligned} \quad (7)$$

Then, we calculate the similarity between $x_{t_v^r}$ and its environmental representation $h_{t_v^r}^{env}$ in a trainable manner:

$$s_v^r = \text{Bilinear}\left(h_{t_v^r}^{env}, x_{t_v^r}\right) = \sigma\left((h_{t_v^r}^{env})^T W_{local}^r x_{t_v^r}\right) \quad (8)$$

The optimization objective of local-level contrastive learning is to encourage higher semantic similarity between the generated guidance node feature $x_{t_v^r}$ and its environmental contrastive sample $h_{t_v^r}^{env}$ when $v \in \mathcal{V}_b$ is a benign node ($y_v = 0$). Conversely, when v is a fraudster ($y_v = 1$), it is desired that $x_{t_v^r}$ exhibits greater dissimilarity with $h_{t_v^r}^{env}$. Based on this, the local-level loss $\mathcal{L}_{local}^r$ can be defined as:

$$s_{local}^{(i)} = 1 - y_i = \begin{cases} 1, y_i = 0 \\ 0, y_i = 1 \end{cases}, \hat{s}_{local}^{r\,(i)} = s_i^r$$

$$\mathcal{L}_{local}^r = \text{BCE}\left(S_{local}, \widehat{S}_{local}^r\right) = \frac{1}{|\mathcal{V}_b|} \sum_{i=1}^{|\mathcal{V}_b|} \text{BCE}\left(s_i^r, \hat{s}_i^r\right) \quad (9)$$

Joint Training. Finally, to optimize the guidance node generation, we employ a joint optimization approach that combines context- and local-level losses. With a hyperparameter $\beta \in [0, 1]$, the joint loss function is defined as:

$$\mathcal{L}_C = \sum_{r=1}^{R} \mathcal{L}_C^r = \sum_{r=1}^{R} \left(\beta \mathcal{L}_{ctx}^r + (1 - \beta)\, \mathcal{L}_{local}^r\right) \quad (10)$$

Neighbor Selection

After generating guidance nodes for each neighborhood subgraph, we select appropriate nodes for message passing to address structural inconsistency, by a fine-grained neighbor selection mechanism that enables evaluating each neighbor's value and selecting more beneficial nodes through a learnable approach. Intuitively, the value of a neighbor node can simply be based on the deviation of the prediction result from the correct label after aggregation. Therefore, the value of a neighbor node is defined by the deviation of the prediction result from the label when it is selected. If the prediction result tends towards the correct label, the selected neighbor node can be considered beneficial for the target node; otherwise, it is harmful or useless. In addition, to address feature inconsistency, we utilize the generated guidance node feature, which has better discriminability, rather than the raw target node feature.

For each neighbor node $u \in \mathcal{N}_v^r$ of the target node v under the relation r, all nodes except the selected neighbor node u are masked, so that a state subgraph $g_{v,u}^r = \{v, u, t_v^r, e_{v,t_v^r}, e_{u,t_v^r}, y_v\}$ is obtained as input to GNN (Figure 2 ❷) with L layers. Here e_{v,t_v^r} is the edge connecting v and t_v^r. This process can intuitively evaluate the effect of neighbor nodes on the target node. After aggregating the information from the selected neighbor node, we introduce a value predictor VP, which is an MLP model. It takes $h_{t_v^r}^{(l)}$ as input, which is the lth-layer activation obtained by feeding $g_{v,u}^r$ into the GNN model. It estimates whether u can enhance the expression of the representation for the semantic information of the target node v. The predictor VP produces the predicted result p_{u,t_v^r}. We train the predictor VP using the following loss function:

$$\mathcal{L}_{\text{VP}}^{(l)} = \sum_{v \in \mathcal{V}} \sum_{u \in \mathcal{N}_v^r} \text{BCE}\left(y_v, p_{u,t_v^r}^{(l)}\right) \quad (11)$$

We adopt an RL model based on policy gradient, which outputs probabilities for neighbor selection, enabling to explore more options and consider a wider range of choices.

The state space of $\mathcal{S}$ is infinite while the action space $\mathcal{A}$ is finite, we construct a policy network, denoted as $\pi_{\theta_{RL}}$, with parameters θ_{RL}. In each layer of the L-layer GNN, $\pi_{\theta_{RL}}$ takes $\mathcal{S}$ as input and outputs the probability of each action for each agent of $u \in \mathcal{N}_v^r$. The detailed exposition on Markov Decision Processes (MDPs) is as follows:

- **State Space** $\mathcal{S}$: As our objective is to decide the retention or removal of each node, so the state subgraph is defined as $g_{v,u}^r = \{v, u, t_v^r, e_{v,t_v^r}, e_{u,t_v^r}, y_v\}$.
- **Action Space** $\mathcal{A}$: Defined as $\mathcal{A} = \{0, 1\}$, $a = 1$ means selecting and retaining neighbor node u, and $a = 0$ indicates discarding this neighbor.
- **Reward Function** $\mathrm{R}(s)$: As mentioned earlier, the value of neighbor node u can be expressed as $w_u^{t_v^r} = \mathrm{VP}^r(\mathrm{D}^l(g_u^r))$, where D^l represents the partial GNN model up to the lth layer. To calculate the reward more fairly and efficiently, we define the reward function:

$$\mathrm{R}(s) = \begin{cases} w_u^{t_v^r} - (\overline{w_{pre}^r} - \epsilon), a = 1 \\ (\overline{w_{pre}^r} - \epsilon) - w_u^{t_v^r}, a = 0 \end{cases} \tag{12}$$

where $\overline{w_{pre}^r}$ represents the average value of all neighbor nodes under the relation r during previous training rounds. This value generally increases as the training progresses. When the reward exceeds 0, it indicates that we encourage the execution of the corresponding action, and the policy network parameters will be adjusted to increase the likelihood of this action. Conversely, if the reward is less than 0, the action will be discouraged. The magnitude of the reward determines the extent of the optimization in the policy network. After several epochs, the value of $\overline{w_{pre}^r}$ may become relatively larger than $w_u^{t_v^r}$. This could result in mistakenly discarding a neighbor node that should not be disregarded. To address this, we set a tolerance parameter ϵ to provide a positive reward in such situations, encouraging the retention of the node, and vice versa. Finally, if we decide to keep a node u connected with v by relation r, it will be added to the selected set neighbor $\mathcal{N}_{RL}^{v,r}$.

Neighbor Aggregation

Message passing enhances the target node by aggregating information from neighbor nodes (Figure 2 ❸). $h_{v,r}^{(l)}$ is the representation of node v at layer l under relation r, where $v \in \mathcal{V}$, $r \in \mathcal{R}$, $l \in \{1, ..., L\}$, L is the number of layers, we define the intra-relation neighbor aggregation as follows:

$$\begin{aligned} z_{v,r}^{(l)} &= h_{v,r}^{(l-1)} || \mathrm{AGG}_r^{(l)} \left(\{h_{u,r}^{(l-1)}, u \in \mathcal{N}_{RL}^{v,r} \cup \{v\}\} \right) \\ h_{v,r}^{(l)} &= \mathrm{ReLU}\left(h_{u,r}^{(l-1)} W_{intra} \right) \end{aligned} \tag{13}$$

where $\mathrm{AGG}_r^{(l)}$ is a weighted aggregator. $h_{v,r}^{l-1}$ is initialized with $x_{t_v^r}$ and $h_{u,r}^{(l-1)}$ is initialized with x_u. Then the target node's representation can be obtained by aggregating the representations $h_{v,r}^{(L)}$ under all relations. It can be defined as:

$$\begin{aligned} z_v &= x_v || \left(\Big\|_{r=1}^{R} h_{v,r}^{(L)} \right) \\ h_v &= \mathrm{ReLU}\left(z_v W_{inter} \right) \end{aligned} \tag{14}$$

Dataset	#Node	IR	Relation	#Edges
YelpChi	45,954	5.9	R-U-R	49,315
			R-S-R	3,402,743
			R-T-R	573,616
Amazon	11,944	13.5	U-P-U	175,608
			U-S-U	3,566,479
			U-V-U	1,036,737
T-Finance	39,357	20.8	A-T-A	21,222,543

Table 1: Dataset and graph statistics. Imbalance Ratio (IR) is the ratio of benign nodes to fraud nodes.

where $\|_{r=1}^{R}$ denotes concatenation aggregation operation, W_{intra} and W_{inter} are learnable aggregation matrices.

Training

After the inter-relation aggregation, the final representation of node $v \in \mathcal{V}$ is h_v. Then we use the cross-entropy loss function to optimize the model.

$$\begin{aligned} p_v &= \mathrm{Softmax}\left(\mathrm{MLP}(h_v)\right) \\ \mathcal{L}_{\mathrm{GNN}} &= \mathrm{BCE}\left(y_v, p_v\right) \end{aligned} \tag{15}$$

Combined with the loss in Equation 11, we define the final loss as Equation 16, where λ is the weight parameter.

$$\mathcal{L}_{DiG\text{-}In} = \mathcal{L}_{\mathrm{GNN}} + \lambda \sum_{l=1}^{L} \mathcal{L}_{\mathrm{VP}}^{(l)} \tag{16}$$

Additionally, it is worth mentioning that our DiG-In-GNN is trained in a mini-batch manner. Considering that fraudster samples are in a minority within a dataset, we set a class sampling probability in our implementation to ensure that the batches are more balanced.

Experiments

In this section, we will investigate the following three research questions through our experiments:

- **RQ1** Does DiG-In-GNN outperform state-of-the-art GNN-based fraud detection methods?
- **RQ2** Does the guidance node generation model contribute to the downstream detection task?
- **RQ3** Can the RL-based neighbor selection using guidance nodes lead to more beneficial aggregation?

Experimental Setup

Datasets. Two widely-used multi-relation graph fraud datasets, YelpChi (Rayana and Akoglu 2015) and Amazon (McAuley and Leskovec 2013) are used. We also adopt a single-relation dataset, T-Finance (Tang et al. 2022). Table 1 shows the statistics of the datasets. YelpChi includes reviews connected by three types of edges: (a) R-U-R connects the reviews made by the same user. (b) R-S-R connects two reviews with the same star rating for the same product. (c) R-T-R connects two reviews on the same product in the same month. Amazon includes product reviews under the musical instruments. In T-Finance, accounts that have transactions are connected.

Method	Dataset	YelpChi			Amazon			T-Finance		
	Metric	AUC	F1-macro	GMean	AUC	F1-macro	GMean	AUC	F1-macro	GMean
GCN$_{(\text{ICLR}'17)}$		0.5983	0.5620	0.4365	0.8369	0.6408	0.5718	0.9176	0.8828	0.8402
GAT$_{(\text{ICLR}'18)}$		0.5715	0.4879	0.1659	0.8102	0.6464	0.6675	0.9553	0.9059	0.8636
GraphSAGE$_{(\text{NeurIPS}'17)}$		0.5439	0.4405	0.2589	0.7589	0.6416	0.5949	0.9116	0.8749	0.8042
Cluster-GCN$_{(\text{KDD}'19)}$		0.5921	0.5523	0.4038	0.8295	0.6488	0.7963	0.9482	0.8973	0.8477
GraphSAINT$_{(\text{ICLR}'20)}$		0.6983	0.5870	0.5857	0.8741	0.7512	0.7677	0.9527	0.8963	0.8620
CARE-GNN$_{(\text{CIKM}'20)}$		0.7619	0.6332	0.6791	0.9067	0.8990	0.8962	0.9014	0.8778	0.8038
PC-GNN$_{(\text{WWW}'21)}$		0.7987	0.6300	0.7160	0.9586	0.8956	0.9030	0.9272	0.8734	0.7967
RioGNN$_{(\text{TOIS}'21)}$		0.8238	0.6526	0.7508	0.9558	0.8848	0.9012	0.9456	0.7945	0.8614
AO-GNN$_{(\text{WWW}'22)}$		0.8805	0.7042	0.8134	0.9640	0.8921	0.9096	-	-	-
H^2-FDetector$_{(\text{WWW}'22)}$		0.8877	0.6944	0.8160	0.9689	0.8392	0.9203	0.9539	0.6062	0.8682
GTAN$_{(\text{AAAI}'23)}$		0.9108	0.8037	0.7768	0.9736	0.9230	0.8879	0.9641	0.9086	0.8720
GHRN$_{(\text{WWW}'23)}$		0.8957	0.7622	0.7703	0.9649	0.9111	0.9002	0.9601	0.8942	0.8552
Ablation	DiG-In-GNN$_{/\text{RL}}$	0.9253	0.8063	0.8145	0.9736	0.9106	0.9126	0.9632	0.9070	0.8731
	DiG-In-GNN$_{/\text{Gen}}$	0.9259	0.8078	0.8245	0.9749	0.9107	0.9054	0.9596	0.8989	0.8456
	DiG-In-GNN$_{/\text{AGG}}$	0.9345	**0.8181**	0.8394	0.9702	0.9205	0.9120	0.9621	0.9061	0.8356
Ours	DiG-In-GNN	**0.9357**	0.8154	**0.8413**	**0.9778**	**0.9251**	**0.9214**	**0.9658**	**0.9112**	**0.8916**

Table 2: Performance Comparison on two multi-relation graphs, YelpChi and Amazon, and a single-relation graph, T-Finance.

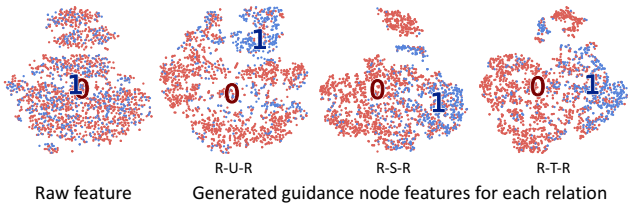

Figure 3: The t-SNE visualization of target nodes' raw features and the guidance node features in YelpChi.

Compared Methods. We compare our method with classic GNNs and state-of-the-art fraud detection models: **GCN** (Kipf and Welling 2017), **GAT** (Velikovi et al. 2018), **GraphSAGE** (Hamilton, Ying, and Leskovec 2017), **Cluster-GCN** (Chiang et al. 2019), **GraphSAINT** (Zeng et al. 2020), **CARE-GNN** (Dou et al. 2020), **PC-GNN** (Liu et al. 2021), **RioGNN** (Peng et al. 2021), **AO-GNN** (Huang et al. 2022), **H^2-FDetector** (Shi et al. 2022), **GTAN** (Xiang et al. 2023) and **GHRN** (Gao et al. 2023). **DiG-In-GNN**$_{/\text{Gen}}$ removes the guidance node generation and **DiG-In-GNN**$_{/\text{RL}}$ removes neighbor selection. **DiG-In-GNN**$_{/\text{AGG}}$ only utilizes the guidance nodes without aggregating neighbors.

Evaluation Metrics. Due to class imbalance in the fraud detection datasets (Brennan 2012), three evaluation metrics, **AUC**, **F1-macro**, and **GMean** are utilized in this paper.

Parameter Setting. We employ Adam with Learning Rate (LR) of 0.003. Specially, for YelpChi and Amazon, the LR for the guidance node generation and GNN is 0.005. The dataset is split into training (40%), validation (20%), and test (40%) sets. The embedding dimension is 128. For YelpChi (Amazon, T-Finance), the training consists of 160 epochs, of which 70 (40, 60) epochs are dedicated to generate guidance nodes. In RL-based neighbor selection, for YelpChi (Amazon), due to various relations, the tolerance values of each relation are 0.1 (0.06), 0.09 (0.06), and 0.08 (0.06), respectively. Also, we sample different numbers of neighbors to train the RL model. The sampling rates for each relation are set as 0.7 (0.3), 0.06 (0.02), 0.008 (0.05). The two RL parameters for T-Finance are set as 0.09 and 0.1 respectively.

Performance Evaluation (RQ1)

Traditional GNNs (e.g., GCN, GAT) exhibit limited performance due to the class imbalance issue and the sophisticated camouflages. The models designed for fraud detection (e.g., PC-GNN, AO-GNN) perform better, with their focus on neighbor sampling. This demonstrates the importance of developing a neighbor sampling approach for fraud detection. However, the previous approaches primarily relied on similarity-based sampling, overlooking intricate camouflages in fraud detection. In contrast, DiG-In-GNN addresses both feature and structural inconsistency by the generated guidance nodes and the RL-based neighbor selection, leading to its outstanding performance, surpassing the prior state-of-the-art methods (e.g., H^2-FDetector, GTAN, GHRN). Table 2 presents a comparison of DiG-In-GNN's performance with the baselines. The results on the three metrics show significant improvements across three datasets, achieving a significant 20.73% improvement on YelpChi. It is interesting GCN works better than some newer methods on T-Finance. The reason may be that T-Finance is more homophilous, so it is easier for the vanilla GCN.

Generated Guidance Node Analysis (RQ2)

We employ t-SNE (Van der Maaten and Hinton 2008) to visualize in Figure 3. Here YelpChi is three-relation, so three corresponding guidance nodes are generated for each node. The raw features appear to be confusing and difficult to distinguish due to feature camouflage, whereas the generated

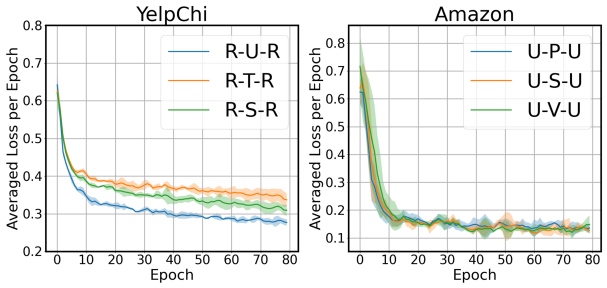

Figure 4: Contrastive learning losses in various relations.

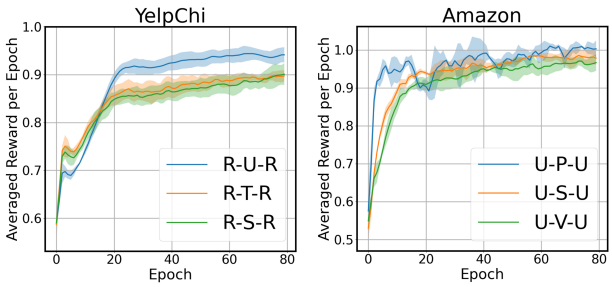

Figure 5: The rewards in neighbor selection. We display the value of $w_u^{t_v^r} + \overline{w_{pre}^r}$ for clearer observation.

guidance nodes are more distinguishable. After the guidance node generation stage, the generated guidance node features have a higher concentration of nodes within the same class and increase the distance between the two classes. Figure 4 illustrates the contrastive learning loss for the two multi-relation datasets. It is evident that in each relation, it exhibits a similar strong trend toward convergence.

We design two ablation studies for the guidance node generation module. DiG-In-GNN$_{/\mathrm{Gen}}$ in Table 2 shows a decrease in performance without guidance nodes. In addition, by removing one of the two scales from multi-scale contrastive learning in DiG-In-GNN, Table 3 demonstrates the benefits of utilizing either context- or local-level guidance nodes. Both of the two scales achieve strong performance when used individually and combining the two scales together leads to better results. Furthermore, we mask all neighbors in DiG-In-GNN$_{/\mathrm{AGG}}$, and with only the raw feature and the guidance node feature when aggregation, it still achieves a comparable result due to the high distinguishability of the guidance nodes.

Neighbor Selection Analysis (RQ3)

As shown in Table 2, DiG-In-GNN$_{/\mathrm{RL}}$ performs worse without fine-grained neighbor selection, which indicates that our RL-based neighbor selection helps GNN transmit more beneficial information. Table 4 presents the statistics of DiG-In-GNN's fine-grained neighbor selection results on the validation set. The "Selected" column indicates the average proportion of neighbor nodes selected, while the "Fraud" and "Benign" columns represent the proportions of selected nodes among fraud and benign nodes, respectively. In YelpChi, the R-U-R edges, connecting reviews from the same user, are homophilic edges, thus enhancing node representation in aggregation. On the other hand, R-T-R represents reviews of the same product within a month, and R-S-R represents reviews with the same rating star under a product. These two relations, R-T-R and R-S-R are weaker semantic connections than R-U-R, resulting in more noise when aggregating. As shown in Table 4, the proportion of discarded nodes is indeed higher in R-S-R and R-T-R than R-U-R. This observation shows that the decisions by our neighbor selection are reasonable. For Amazon, the U-V-U relation, which connects two users with similar reviews, exhibits the lowest neighbor selection rate. This might be due to feature camouflage. Furthermore, it is observed that the overall probability of a fraud node being selected is lower, consistent with the class imbalance in fraud detection. This result further validates our RL-based approach. In T-Finance, most nodes are selected, but the fraud node selection rate is still slightly lower than the benign node selection rate. Figure 5 illustrates the process of increasing rewards, showing an overall upward trend and a gradual flattening as training progresses.

Dataset	Method	AUC	F1-macro	GMean
YelpChi	w/Context	0.9280	0.8016	0.8281
	w/Local	0.9306	0.8108	0.8209
Amazon	w/Context	0.9728	0.9167	0.9183
	w/Local	0.9725	0.9103	0.9198
T-Finance	w/Context	0.9553	0.9024	0.8694
	w/Local	0.9615	0.8988	0.8683

Table 3: Ablation study to demonstrate the effectiveness of the two scales of contrastive learning.

Dataset	Relation	Selected	Fraud	Benign
YelpChi	R-U-R	90.32%	50.07%	92.46%
	R-S-R	71.21%	36.05%	77.20%
	R-T-R	67.01%	35.03%	72.62%
Amazon	U-P-U	92.25%	17.05%	99.95%
	U-S-U	91.99%	20.32%	99.34%
	U-V-U	83.50%	15.97%	90.34%
T-Finance	A-T-A	99.89%	99.46%	99.92%

Table 4: Statistics of the neighbor selection decisions.

Conclusion

In this paper, we present a multi-relation graph-based fraud detector. Enhanced by the guidance node and RL-based fine-grained neighbor selection, DiG-In-GNN solves two inconsistency problems in the fraud graphs. Experiments on three public datasets demonstrate the effectiveness of our method. For future work, generalizing DiG-In-GNN to multi-class node classification could be a promising direction.

Acknowledgments

This work is supported by the National Key R&D Program for the 14th-Five-Year Plan of China (2023YFC3804104 in 2023YFC3804100), National Natural Science Foundation of China under Grants No. 61972085, 62072099, 62232004, 62272101, Jiangsu Provincial Key Laboratory of Network and Information Security under Grant BM2003201, Jiangsu Provincial Key Research and Development Program under Grant BE2022065-4, Natural Science Foundation of Jiangsu Province under Grant BK20230083, Key Laboratory of Computer Network and Information Integration of Ministry of Education of China under Grant No. 93K-9, and the Fundamental Research Funds for the Central Universities. We also thank the Big Data Computing Center of Southeast University for providing the experimental environment and computing facility.

References

Brennan, P. 2012. A comprehensive survey of methods for overcoming the class imbalance problem in fraud detection. *Institute of technology Blanchardstown Dublin, Ireland*.

Chiang, W.-L.; Liu, X.; Si, S.; Li, Y.; Bengio, S.; and Hsieh, C.-J. 2019. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 257–266.

Dou, Y.; Liu, Z.; Sun, L.; Deng, Y.; Peng, H.; and Yu, P. S. 2020. Enhancing graph neural network-based fraud detectors against camouflaged fraudsters. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 315–324.

Gao, Y.; Wang, X.; He, X.; Liu, Z.; Feng, H.; and Zhang, Y. 2023. Addressing heterophily in graph anomaly detection: A perspective of graph spectrum. In *Proceedings of the ACM Web Conference 2023*, 1528–1538.

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

Huang, M.; Liu, Y.; Ao, X.; Li, K.; Chi, J.; Feng, J.; Yang, H.; and He, Q. 2022. AUC-oriented Graph Neural Network for Fraud Detection. In *Proceedings of the ACM Web Conference 2022*, 1311–1321.

Jiang, M.; Cui, P.; and Faloutsos, C. 2016. Suspicious Behavior Detection: Current Trends and Future Directions. *IEEE Intelligent Systems*, 31(1): 31–39.

Jin, M.; Liu, Y.; Zheng, Y.; Chi, L.; Li, Y.; and Pan, S. 2021. ANEMONE: Graph Anomaly Detection with Multi-Scale Contrastive Learning. In Demartini, G.; Zuccon, G.; Culpepper, J. S.; Huang, Z.; and Tong, H., eds., *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*, 3122–3126. ACM.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations*.

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

Liu, Z.; Chen, C.; Yang, X.; Zhou, J.; Li, X.; and Song, L. 2018. Heterogeneous graph neural networks for malicious account detection. In *Proceedings of the 27th ACM international conference on information and knowledge management*, 2077–2085.

Ma, X.; Wu, J.; Xue, S.; Yang, J.; Zhou, C.; Sheng, Q. Z.; Xiong, H.; and Akoglu, L. 2021. A comprehensive survey on graph anomaly detection with deep learning. *IEEE Transactions on Knowledge and Data Engineering*.

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

Peng, H.; Zhang, R.; Dou, Y.; Yang, R.; Zhang, J.; and Yu, P. S. 2021. Reinforced neighborhood selection guided multi-relational graph neural networks. *ACM Transactions on Information Systems (TOIS)*, 40(4): 1–46.

Rayana, S.; and Akoglu, L. 2015. Collective opinion spam detection: Bridging review networks and metadata. In *Proceedings of the 21th acm sigkdd international conference on knowledge discovery and data mining*, 985–994.

Shi, F.; Cao, Y.; Shang, Y.; Zhou, Y.; Zhou, C.; and Wu, J. 2022. H2-FDetector: A GNN-based Fraud Detector with Homophilic and Heterophilic Connections. In *Proceedings of the ACM Web Conference 2022*, 1486–1494.

Tan, C. L.; Chiew, K. L.; Yong, K. S.; Abdullah, J.; Sebastian, Y.; et al. 2020. A graph-theoretic approach for the detection of phishing webpages. *Computers & Security*, 95: 101793.

Tang, J.; Li, J.; Gao, Z.-C.; and Li, J. 2022. Rethinking Graph Neural Networks for Anomaly Detection. In *International Conference on Machine Learning*.

Van der Maaten, L.; and Hinton, G. 2008. Visualizing data using t-SNE. *Journal of machine learning research*, 9(11).

Velikovi, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. In *International Conference on Learning Representations*.

Wang, L.; Li, P.; Xiong, K.; Zhao, J.; and Lin, R. 2021. Modeling Heterogeneous Graph Network on Fraud Detection: A Community-based Framework with Attention Mechanism. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 1959–1968.

Wang, Y.; Zhang, J.; Huang, Z.; Li, W.; Feng, S.; Ma, Z.; Sun, Y.; Yu, D.; Dong, F.; Jin, J.; et al. 2023. Label Information Enhanced Fraud Detection against Low Homophily in Graphs. In *Proceedings of the ACM Web Conference 2023*, 406–416.

Xiang, S.; Zhu, M.; Cheng, D.; Li, E.; Zhao, R.; Ouyang, Y.; Chen, L.; and Zheng, Y. 2023. Semi-supervised Credit Card Fraud Detection via Attribute-driven Graph Representation. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Yuan, D.; Miao, Y.; Gong, N. Z.; Yang, Z.; Li, Q.; Song, D.; Wang, Q.; and Liang, X. 2019. Detecting fake accounts in online social networks at the time of registrations. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, 1423–1438.

Zeng, H.; Zhou, H.; Srivastava, A.; Kannan, R.; and Prasanna, V. 2020. GraphSAINT: Graph sampling based inductive learning method. In *International Conference on Learning Representations*.

Zhang, G.; Wu, J.; Yang, J.; Beheshti, A.; Xue, S.; Zhou, C.; and Sheng, Q. Z. 2021. FRAUDRE: fraud detection dual-resistant to graph inconsistency and imbalance. In *2021 IEEE International Conference on Data Mining (ICDM)*, 867–876. IEEE.

Zhang, M.; Meng, W.; Lee, S.; Lee, B.; and Xing, X. 2019. All your clicks belong to me: investigating click interception on the web. In *28th USENIX Security Symposium (USENIX Security 19)*, 941–957.