Enhancing Fraud Detection via GNNs with Synthetic Fraud Node Generation and Integrated Structural Features[*]

Georgia Kapetadimitri[**], Dimitrios Hristu-Varsakelis

Department of Applied Informatics, University of Macedonia,
Egnatia 156, Thessaloniki, 54636, Greece
{aid23007,dcv}@uom.edu.gr

Abstract. Graph Neural Networks are widely employed for node classification in attributed networks. When it comes to fraud detection, however, GNNs can perform poorly, because a node's features are typically computed based on its local neighborhood, and this allows fraudsters to "blend in" among legitimate users. In this paper, GNNs and supervised contrastive learning are proposed for fraud detection on datasets where fraudsters may use intricate strategies to camouflage themselves within the network. We train our GNNs using novel structural features in addition to those typically used in similar studies. The proposed features are based on the empirical probability distributions of various graph structural attributes which are extracted from a given dataset. We also apply supervised contrastive learning, enhanced with synthetic samples for the minority class (i.e., the fraudsters). Under our approach, the classifying capability of the GNN (measured via F1-macro, AUC, Recall) is improved by boosting the representation power of the calculated embeddings that maximize the similarity between legitimate users while minimizing that between fraudsters and legitimate users. Numerical experiments on two real-world multi-relation graph datasets (Amazon and YelpChi) demonstrate the effectiveness of the proposed method, whose improvements over the state-of the-art were especially significant in the larger YelpChi dataset.

Keywords: Graph neural network · Fraud detection · Contrastive learning.

1 Introduction

This paper is concerned with the problem of fraud detection in attributed networks via graph neural networks (GNNs). In the recent years, GNNs have re-

[*] This paper is a result of research conducted within the "MSc in Artificial Intelligence and Data Analytics" of the Department of Applied Informatics of University of Macedonia. The presentation of the paper is funded by the University of Macedonia Research Committee.

[**] corresponding author

ceived significant attention [19], [17], [13]. Yet, "simple" GNNs are not by themselves sufficient to perform effective fraud detection in multi-attribute graphs because of various challenges inherent in this task. GNNs are based on the homophily assumption, meaning that all nodes in the graph are of the same class and have similar features. However, in a fraud detection setting there are at least two dissimilar classes of nodes, namely "legitimate" users vs. "fraudsters", the latter usually being a small fraction of the total nodes. One may expect that these two classes exhibit different features and characteristics. Moreover, fraudsters may deliberately alter their characteristics in order to resemble those of the legitimate users. This behavior is called “camouflage" [2] and consists of two separate devious practices illustrated in Figure 1: i) feature camouflage; fraudsters may attempt to modify their features in order to “look like" regular users, and ii) structural inconsistency; fraudulent users tend to connect less with other fraudsters and more with legitimate ones in order to blend in. As a result, the neighborhood aggregation process that occurs during GNN training will tend to blend together the features of legitimate users with those of the fraudsters, resulting in a "fading away" of the dissimilarity of the fraudsters. This is why directly using GNNs for fraud detection is not a very effective approach.

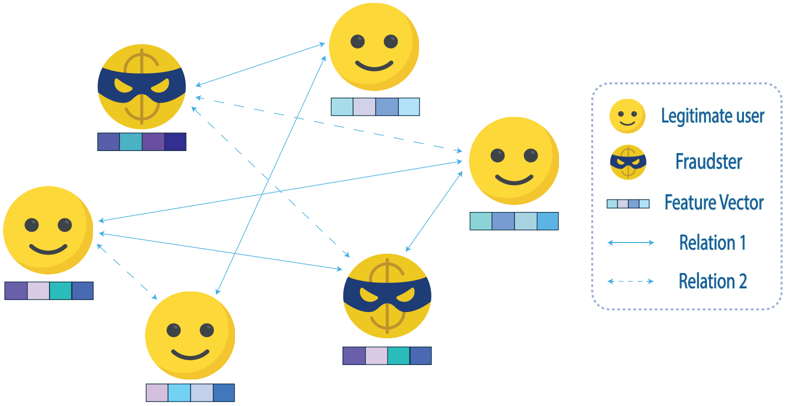

Fig. 1. Toy example of a multi-relation graph that illustrates 2 techniques fraudsters use to disguise themselves. Relation 1 may correspond to having reviewed at least one product in common while relation 2 may correspond to having purchased the same product in a given time period. 1) Feature camouflage: fraudsters adjust their attributes to blend in with legitimate entities. An attribute could be the percentage of special characters appearing in the user’s review. The fraudster situated at the bottom of the image tries to imitate the features (seen by the resemblance of the colors in their feature vector) of the leftmost legitimate user with whom they are connected by relation 1. 2) Structural inconsistency: fraudsters connect more often with benign entities than with other fraudsters.

While GNN-based models have provided some solutions for effectively addressing the imbalanced nature of the problem, and have demonstrated very good results in fraud detection tasks, there are still challenges associated with their use. One of those challenges is to improve their ability to distinguish between fraudster and legitimate users. Because of camouflage, atomic features of a node can be unreliable when used in the GNN process. Furthermore, although previous methods have addressed the general imbalance in the dataset by employing methods such as a neighborhood sampler [7], [1], [15] within the supervised contrastive learning process, the imbalance between the 2 classes remains, resulting in fewer contrastive pairs for the model to learn from and in contrastive embeddings that are not sufficiently representative of the nodes.

The contribution of this paper is to propose a novel approach that combines GNNs with i) supervised contrastive learning, ii) the introduction of additional graph-structural features, and iii) synthetic fraud nodes, to improve fraud detection in networks. To tackle the aforementioned difficulties, and provide more representative node features and more positive-negative samples in the supervised contrastive learning process, a new model, termed **B**oosted **R**epresentation with **I**ntegrated Structural Features and Synthetic Fraud G**E**neration (BRIE) is proposed in this paper. The new structural features which we propose are used alongside the basic pre-existing features which are part of similar studies, and are selected specifically for mitigating the camouflage of fraudsters. These features do not describe a specific property of a node but instead consider the relations between the node and others within the entire graph. As we will see, this will lead to improved discrimination between fraudsters and legitimate users.

In order to address the imbalance between the two classes in our supervised contrastive learning module, the training dataset is enriched with synthetic fraud nodes; these are created based on the statistical properties of each class's node features, independent of the main dataset (i.e., without having to introduce new edges linking synthetic nodes to existing ones). This way, the quality of the embeddings produced within the module is improved, as the embeddings become more robust and representative of the nodes of the dataset. Our approach was tested on two real-world datasets, against some of the recent GNN-based fraud detection approaches, in terms of F1-score, AUC, and recall. Our model showed significant improvements over the state-of-the-art, especially in the larger YelpChi dataset [10] where superior performance was attained, highlighting the effectiveness of out proposed solution.

The remainder of the paper is organized as follows. Section 2 discusses related literature and introduces relevant definitions. In Section 3 the proposed model is presented in detail. Section 4 provides training details, and discusses the performance of our model.

2 Background and Preliminaries

We begin by discussing relevant literature, before defining important notation and stating the main problem.

2.1 Graph-based Fraud detection

Traditional graph-based fraud detection methods utilize various data mining and machine learning techniques and can be classified in five distinct categories: community-based, probabilistic-based, structural-based, compression-based and decomposition-based approaches [9]. Community-based techniques focus on identifying anomalous subgraphs within networks by analyzing the clustering and community structures. Probabilistic-based methods utilize probabilistic models to capture the uncertainty and relationships between nodes in a graph, enabling the detection of unusual patterns. Structural-based approaches focus on the overall network structure, including node connections and properties, to identify anomalies based on deviations from expected patterns. Compression-based methods aim to reduce the complexity of network data while preserving essential information for anomaly detection, often by compressing the graph representation. Decomposition-based approaches break down complex networks into simpler components or subgraphs, allowing for the analysis of local anomalies and patterns within the network.

2.2 Graph Neural Networks and GNN-based Fraud Detection

Graph Neural Networks (GNNs) are a class of deep learning models designed for processing graph-structured data. Unlike traditional neural networks that work well with grid-structured data, GNNs excel in handling non-Euclidean and irregularly structured information, where entities are interconnected in a graph. They are particularly useful for tasks involving modeling relationships between nodes, uncovering patterns in complex networks, and predicting node properties.

The core architecture of GNNs involves iterative information aggregation from neighboring nodes which happens mainly with graph convolutions. A graph convolution predicts the features of the node in the next layer as a function of the node's neighbours' features. This allows the model to learn hierarchical and contextual representations, making GNNs suitable for node classification, link prediction, and graph classification.

GNNs that use a convolutional operator for the propagation of information between nodes are either spectral or spatial. Spectral approaches are based on graph signal processing and define the convolution operator in the spectral domain. Spatial approaches define convolutions directly on the graph based on the graph topology [17]. GCN [4] belongs to the spectral domain and is one of the most widely used types of GNN due to its combined simplicity and effectiveness. Another important type of GNN is GAT [12], a spatial approach that employs attention mechanisms to weigh the importance of neighboring nodes when processing information on a graph.

Recent works have enhanced GNNs for the specific purpose of discovering fraudsters. CARE-GNN [2] uses a label-aware similarity in order to choose meaningful neighbors and an adaptive threshold trained with reinforcement learning in order to filter neighbors for nodes. PC-GNN [7] tackles the problem by alleviating class imbalance, meaning by oversampling the minority class and undersampling the majority one, thus mitigating the structural inconsistency.

2.3 Contrastive Learning

Contrastive learning (CL) is a machine learning technique used widely in self-supervised computer vision tasks. CL approaches teach a model to pull together the augmentations of a target image (a.k.a., the "anchor") and a matching ("positive") image in embedding space, while also pushing apart the anchor from many non-matching ("negative") images. This process brings similar class embeddings closer together while pushing embeddings from different classes farther apart. To achieve close proximity between the embeddings of positively related pairs in a metric space, the goal is to make the representations invariant to irrelevant differences within positive pairs in the input space. At the same time, the model should learn a covariant representation for negatively related pairs in order to represent the large distances between their embeddings in the metric space [5].

Supervised contrastive learning (SCL) [3] takes the concept of self-supervised contrastive learning and extends it into a fully supervised environment by utilizing the labels of the dataset. It considers samples of the same class as positive examples and it contrasts all samples from different classes, thus creating negatives examples. Such works that have successfully employed SCL in the graph-based fraud detection problem are CACO-GNN [1] that improves the consistency of neighbor sampling by effectively compacting the feature camouflage of fraudsters in graphs and IMINF [15] that utilizes an interactive learning framework while also addressing imbalanced label distribution and inappropriate representation of the target node during relation aggregation.

2.4 Definitions and Problem statement

We go on to provide some necessary terminology.

Definition 1 (Attributed network/graph). *An attributed graph can be denoted as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{X}\}$, where $\mathcal{V} = \{v_1, \ldots, v_N\}$ is a set of nodes, $\mathcal{E}$ is the edge set, $\mathcal{X} \in \mathbb{R}^{N \times d}$ is the feature matrix and d is the number of features each node has. An extended attributed graph $\mathcal{G}' = \{\mathcal{V}, \mathcal{E}, \mathcal{X}'\}$ has an extended feature matrix $\mathcal{X}' \in \mathbb{R}^{N \times (d+k)}$ where k is the number of additional types of features added to $\mathcal{G}$. Hence, $\mathcal{X}_{v,:}$ is the feature vector (row) of node v and $\mathcal{X}_{:,j}$ is the feature vector (column) for feature j.*

Definition 2 (Multi-relational graph). *Given a multi-relational graph $\mathcal{G} = \{\mathcal{V}, \mathcal{X}, \mathcal{E}_r|_{r=1}^{R}, \mathcal{A}_r|_{r=1}^{R}, \mathcal{Y}\}$, $\mathcal{E} = \{E_1, \ldots, E_R\}$ is the edge set of R relations, $\mathcal{A} = \{A_1, \ldots, A_R\}$ is a set of corresponding adjacency matrices of R relations and $\mathcal{Y} = \{\mathcal{Y}_f, \mathcal{Y}_l\}$ are the labels of nodes $\in \mathcal{V}$, where $\mathcal{Y}_f$ corresponds to fraudsters and $\mathcal{Y}_l$ to legitimate nodes. For each node $v_i \in \mathcal{V}, x_i$ is a d-dimension feature vector and $y_i \in \mathcal{Y}$ is the label of the node.*

Problem 1 Given a multi-relational graph $\mathcal{G}$, the problem to be solved is the detection of fraud nodes in $\mathcal{G}$. Hence, 1) there are two types of labels, $\mathcal{Y} = \{\mathcal{Y}_f, \mathcal{Y}_l\}$ where $y_i = \mathcal{Y}_f$ is the label for a fraudster and $y_i = \mathcal{Y}_l$ for a legitimate user, and 2) the number of fraud nodes is much smaller than the number of legitimate ones, i.e., $\sum_{i=1}^{N} \mathbb{I}[y_i = \mathcal{Y}_f] \ll \sum_{i=1}^{N} \mathbb{I}[y_i = \mathcal{Y}_l]$, where $\mathbb{I}$ is the indicator function.

3 Methodology

In order to solve Problem 1, we will introduce a feature generation module, a synthetic fraud generation module and neighborhood sampling. This section outlines these three items, along with the training procedure, model configuration, and evaluation metrics utilized in the study.

3.1 Feature generation

For each node $v_i \in \mathcal{V}$ and for all relations $E_1, \ldots, E_R$ in a given multi-relational graph $\mathcal{G}$, we will calculate the probability distributions of six additional quantitative structural features for fraudster nodes (whose labels are $\mathcal{Y}_f$) and legitimate nodes (with label $\mathcal{Y}_l$). These additional features are proposed here for the first time and are used alongside the basic pre-existing features of each dataset (32 for the YelpChi and 25 for the Amazon [8] dataset). We chose to introduce the new features because the pre-existing features describe only atomic attributes of each node and are not by themselves sufficient - for example they do not capture the local structure of the graph. The additional structural features are:

- **degree**: the degree of the node,
- **second-order degree**: the number of nodes exactly 2 hops away from the target node,
- **opposite label count**: the number of neighbors of the opposite label compared to that of the node,
- **preferential attachment**: the product of the target node's degree and the average degree of its neighbors,
- **clustering coefficient** [16]: the degree to which the neighbors of the target node are interconnected,
- **structural similarity** [18]: the resemblance in the local structures of nodes within a complex network, quantified using the Kullback-Leibler (K-L) divergence to measure the difference between the probability distributions of node pairs.

After computing these structural features for each node, we go on to calculate their empirical distributions for each class. Then, for each node, we replace its feature values with a sample from that feature's distribution (for the class the node belongs to). We do this in order to "prevent" any fraudster from directly manipulating its features to desired values[*]. This process is denoted as,

$$x'_{struct} \sim \text{Distribution}(\mathcal{X}_{:,struct})$$

[*] Depending on the graph relation, some of the structural features described above may not be possible to calculate or may have a constant value across all nodes which is undesired. Thus, the number of additional structural features k may vary but will be at most 6 for any given dataset.

where x'_{struct} denotes the feature column of values produced for the additional structural feature named *struct*. Then, the structural feature values are appended to the node's existing "basic" features to form the new feature vector for the node. Hence, for a target node $v \in \mathcal{V}$ (fraudster or legitimate),

$$\mathcal{X}_{v,:} = \left[x_{basic_1}, \ldots, x_{basic_d}, x'_{struct_1}, \ldots, x'_{struct_k}\right] \tag{1}$$

is the new feature vector of the node v. $\mathcal{X}_{v,:} \in \mathbb{R}^{(d+k)}$ where d is the number of basic features and k is the dimension of the structural feature vector. The feature generation module is illustrated in Figure 2. As we will later see, this replacement of the original node features by samples from their class-specific distributions will lead to better performance. Experiments were done with maintaining node features at their original values, but they lead to inferior results.

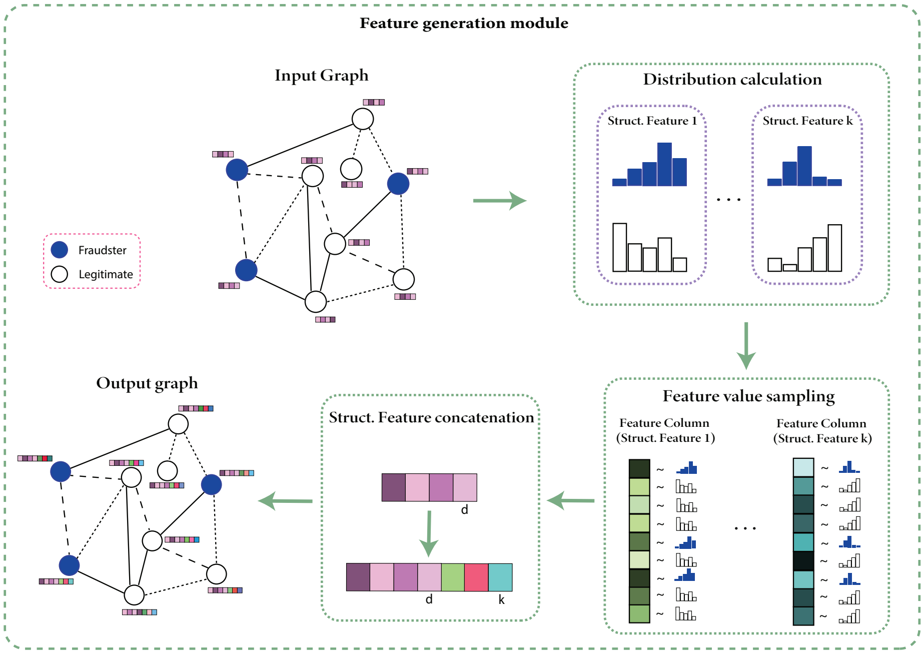

Fig. 2. Feature generation module showing the total process of enhancing an input graph $\mathcal{G}$ with additional structural features. At the end of the process an extended graph $\mathcal{G}'$ is created where each node has an extended feature vector.

3.2 Contrastive similarity

To address the challenge of camouflage, an additional similarity metric is employed along with feature similarity to differentiate between node embeddings.

This metric is termed contrastive similarity and is produced by utilising supervised contrastive learning. As in [1], we measure the contrastive similarity between nodes and the learned positive prototype of the minority class (i.e fraudsters). Thus, the positive prototype is the representative of its class, an idea from [14]. The positive prototype is not a real node transformed in the embedding space but is instead an artificial embedding that is trained in order to capture the most representative features of its class. Using a positive prototype obtained by the minority class as the anchor, instead of contrasting each target node with each fraudster, reduces computational complexity. For the central node v at l-th layer of the GNN under relation E_r, the contrastive embedding is[*],

$$C_{v,r}^{(l)} = \sigma\left(W_{v,r}^{(l)}, h_{v,r}^{(l-1)}\right) \tag{2}$$

where $h_{v,r}^{(l-1)}$ is the node embedding for node v at relation r learned in the layer $l-1$, $W_{v,r}^{(l)}$ is the parameter that will be learned, σ is the activation function (set to ReLU in this study). $C_{v,r}^{(l)}$ is the contrastive embedding and it is updated during the training of the model. We may then calculate the similarity score between the positive prototype and any target node as

$$S_{v,r}^{(l)} = Similarity\left(C_{v,r}^{(l)}, t_r^{(l)}\right) \tag{3}$$

where $t_r^{(l)}$ is a learnable parameter. $Similarity(\cdot)$ is any distance function, in our case the cosine distance,

$$cos(a,b) = \frac{a \cdot b}{||a|| \cdot ||b||}. \tag{4}$$

Eq. 3 will represent the probability of a node embedding being associated with a fraudster. This means that as the contrastive learning process progresses, the contrastive embedding of the fraudulent node will be close to that of the positive prototype.

Supervised contrastive learning loss Each contrastive embedding of a node is contrasted against the positive prototype, t^l

$$\mathcal{L}_{\text{Contra}}^{r} = -\sum_{l=1}^{L} \sum_{\substack{y_i=1,\\ y_j=0}} \left[\log \frac{\exp(t_r^{(l)} \cdot c_{i,r}^{(l)}/\tau)}{\exp(t_r^{(l)} \cdot c_{i,r}^{(l)}/\tau) + \sum_j \exp(t_r^{(l)} \cdot c_{j,r}^{(l)}/\tau)}\right] \tag{5}$$

where τ is the temperature hyperparameter. The overall contrastive learning loss function can be expressed as:

$$\mathcal{L}_{\text{Contra}} = \frac{1}{R} \sum_{r=1}^{R} \mathcal{L}_{\text{Contra}}^{r}. \tag{6}$$

[*] The notation for this and the following formulas are as in [1].

3.3 Synthetic fraud node generation

Previous works have introduced neighbor sampling to mitigate the imbalanced nature of the problem and the camouflage of the fraudsters. However, in the process of supervised contrastive learning, that imbalance is not addressed, resulting in fewer negative-positive pairs for the model to learn from. The addition of synthetic fraud nodes effectively addresses this issue, as illustrated in Figure 3.

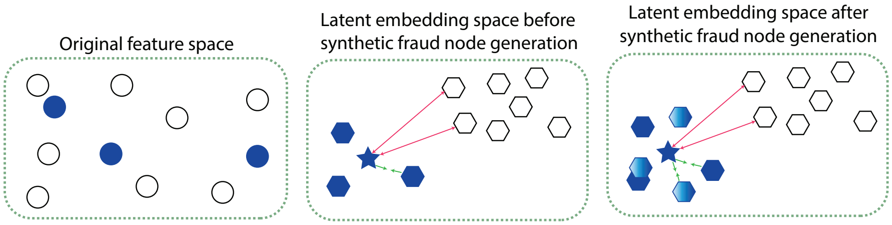
Original feature space
Latent embedding space before synthetic fraud node generation
Latent embedding space after synthetic fraud node generation

Fig. 3. Effect of contrastive learning and synthetic fraud generation. Blue points indicate the fraudsters, white ones indicate the legitimate users while blue gradient points are the synthesized fraudsters. The blue star denotes the prototype of the fraudster class. The representation quality of the positive (fraud) prototype is increased as it is contrasted with more positive elements.

Hence, in order to enhance the supervised contrastive learning process, synthetic nodes with labels $y_i = \mathcal{Y}_f$ are generated based on the distributions of the basic features and also on those of the new structural features. As this process occurs only in the SCL module, there is no need to establish new edges with existing nodes - only the synthetic nodes' features are to be generated. Thus, for each basic feature $\mathcal{X}_{:,basic}$ and structural feature $\mathcal{X}_{:,struct}$, we sample from their respective distributions to create synthetic values x'_{basic} and x'_{struct}. This sampling is denoted as,

$$x'_{basic} \sim \text{Distribution}(\mathcal{X}_{:,basic})$$

$$x'_{struct} \sim \text{Distribution}(\mathcal{X}_{:,struct})$$

Then, the synthetic feature values are combined to form the feature vector for the synthetic fraud node,

$$\mathcal{X}_{synthetic,:} = \left[x'_{basic_1}, \ldots, x'_{basic_d}, x'_{struct_1}, \ldots, x'_{struct_k}\right] . \tag{7}$$

3.4 Neighborhood sampling

Fraudulent entities adeptly disguise themselves by emulating the actions of legitimate users or establishing illicit connections with them. Consequently, directly integrating all neighboring nodes in the intra-relation aggregation stage becomes unfeasible. Instead, we will use a neighbor sampling technique that selects neighbors based on feature and contrastive similarity. The contrastive embedding measures the distance of the target node from the positive prototype. Contrastive similarity is the distance between two contrastive embeddings. Nodes with matching labels and the smallest contrastive similarity difference from the target node are sampled as neighbors. The set of contrastive similarity differences between a node v and its neighbors is defined as

$$\mathbb{D}_r^{(l)}(v) = \left\{|S_{v,r}^{(l)} - S_{u,r}^{(l)}|\right\}|_{A_r(v,u)>0} \tag{8}$$

The top k neighbors with the smallest difference in contrastive similarity are chosen, and this set of sampled neighbors is denoted as $\dot{\mathcal{N}}_r^{(l)}(v)$.

The feature similarity score focuses on the distance of the target node from other nodes based on their features in the latent space. A higher score signifies closer node proximity in terms of feature consistency. The set of feature similarities of a node v to its neighbors is defined as,

$$\mathbb{S}_r^{(l)}(v) = \left\{Similarity(h_{v,r}^{(l)}, h_{u,r}^{(l)})\right\}|_{A_r(v,u)>0} \tag{9}$$

The top k neighbors with the biggest feature similarity are sampled and this set is depicted as $\ddot{\mathcal{N}}_r^{(l)}(v)$. The set union of the contrastive similar neighbors and feature similar neighbors is the set of consistent neighbors of a target node v and is defined as,

$$\mathcal{N}_r^{(l)}(v) = \dot{\mathcal{N}}_r^{(l)}(v) \cup \ddot{\mathcal{N}}_r^{(l)}(v). \tag{10}$$

3.5 Intra-relation and Inter-relation aggregation

We will apply an intra-relation aggregation stage where for each target node v we calculate its embedding with respect to each relation R. Therefore, the intra-relation embedding of node v at relation r ($h_{v,r}^{(l)}$) is the aggregation of the features of v in the previous layer $l-1$, the contrastive embedding of v and the embedding of the features of the neighbors of v in the current layer. This statement can be described compactly as

$$h_{v,r}^{(l)} = \sigma\left(W_r^{(l)}\left(h_{v,r}^{(l-1)} \oplus c_{v,r}^{(l)} \oplus AGG_r\{h_{u,r}^{(l)}\}|_{r\in R}\right)\right), \tag{11}$$

where $\oplus$ is the concatenation operator, AGG_r is the mean aggregator and $W_r^{(l)}$ is the learnable weight matrix at layer l for relation r. We will also perform inter-relation aggregation, where the information from all relations is aggregated in order to update the embedding of the target node v,

$$h_v^{(l)} = \sigma\left(W^{(l)}\left(h_v^{(l-1)} \oplus \{h_{u,r}^{(l)}\}|_{r\in R}\right)\right), \tag{12}$$

where $W^{(l)}$ is the learnable weight matrix at layer l.

3.6 Training

This section outlines the training of the model, the steps taken in order to produce the final probability of a node being fraudster or legitimate and discusses the loss function of the model. The overall training of BRIE is shown in Algorithm(1).

Algorithm 1 BRIE

```
input
    multi-relational graph extended with structural features:
        $\mathcal{G}' = \{\mathcal{V}, \mathcal{X}', \mathcal{E}_r|_{r=1}^{R}, \mathcal{A}_r|_{r=1}^{R}, \mathcal{Y}\}$;
    batches, epochs, layers: B, E, L;
output
    vector representations $h_v^{(L)}$ for each node in $\mathcal{V}$
for $e \leftarrow 1, \ldots, E$ do
    for $b \leftarrow 1, \ldots, B$ do
        Construct subgraphs using nodes in b and the edges connecting them to their immediate neighbors
        for $l \leftarrow 1, \ldots, L$ do
            for $r \leftarrow 1, \ldots, R$ do
                Sample consistent neighbors for target node v using (3), (9), (10)
                Synthesize fraudster nodes for contrastive learning process using (7)
                $h_{v,r}^{(l)} \leftarrow (11), v \in \mathcal{V}_b$
            end for
            $h_v^{(l)} \leftarrow (12), v \in \mathcal{V}_b$
            $\mathcal{L}_{\text{Contra}} \leftarrow (5), (6)$
            $\mathcal{L}_{\text{GNN}} \leftarrow (14)$
            $\mathcal{L} \leftarrow (15)$
        end for
    end for
end for
```

After obtaining the final embedding $h_v^{(l)}$ for each node in $\mathcal{V}$, the embeddings are passed to a Multi-Layer Perceptron (MLP) in order to generate the vector p_v containing the probabilities with which each node belongs to the two classes:

$$p_v = MLP(h_v^{(L)}). \tag{13}$$

Then, the model is trained using a cross-entropy loss function,

$$\mathcal{L}_{\text{GNN}} = -\sum_{v \in \mathcal{V}} \left(y_v \log(p_v) + (1 - y_v) \log(1 - p_v) \right). \tag{14}$$

The overall loss function of the proposed model is

$$\mathcal{L} = \mathcal{L}_{\text{GNN}} + \lambda \mathcal{L}_{\text{Contra}}, \tag{15}$$

where λ is a hyperparameter. Figure4 illustrates the training process graphically.

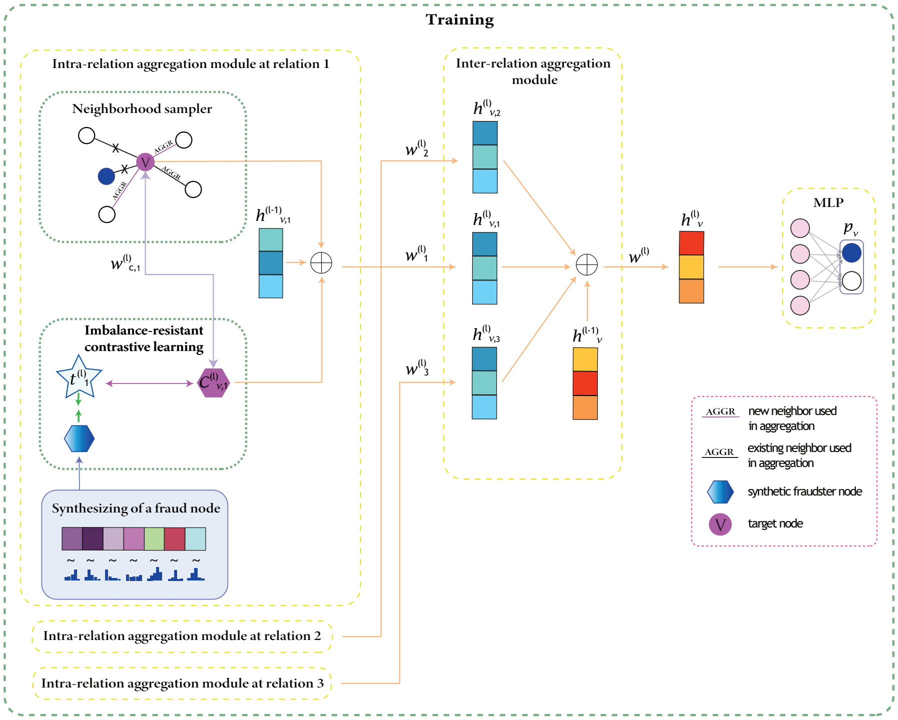
Training
Intra-relation aggregation module at relation 1
Neighborhood sampler
V
AGGR
AGGR
AGGR
h(l-1) v,1
w(l) c,1
Imbalance-resistant contrastive learning
t(l) 1
C(l) v,1
Synthesizing of a fraud node
Intra-relation aggregation module at relation 2
Intra-relation aggregation module at relation 3
Inter-relation aggregation module
w(l) 2
w(l) 1
w(l) 3
h(l) v,2
h(l) v,1
h(l) v,3
h(l-1) v
w(l)
h(l) v
MLP
p v
AGGR new neighbor used in aggregation
AGGR existing neighbor used in aggregation
synthetic fraudster node
V target node

Fig. 4. Training process for a node v. In the intra-relation aggregation the imbalance-resistant contrastive learning module, enhanced with synthesized fraud nodes collaborates with the neighborhood sampler in order to produce the node embedding $h_{v,r}$ which will feed the inter-relation aggregation. The inter-relation aggregation produces the final node embedding h_v for node v. The legitimacy of a node v is decided by a MLP at the end which transforms the node embedding h_v to a probability vector.

4 Experiments

This section gives practical details regarding our numerical experiments and compares the results with those of recent works. Our BRIE code is available at https://github.com/brie-user/brie-model. An ablation study was omitted due to page limits.

4.1 Implementation details and Metrics

For training, the non-fraud samples are undersampled in order to create a non-fraud-to-fraud ratio of 3:1 for the Amazon dataset and 2:1 for YelpChi. Synthesizing fraud nodes permits BRIE to use more non-fraud samples in each training epoch as opposed to [1] and [15] where they used a 1:1 ratio. The learning rate was set to 0.01, λ to 2, temperature τ to 0.2 and number of layers L to 1. The optimizer was AdamW, and the node embedding size was 64. Our model was trained using the mini-batch training approach. The batch size for Amazon was

256 while for YelpChi it was 1024. The training-test-validation split was 40%-20%-20% and the samples of each set were randomly chosen. To achieve the best parameters possible, the model was trained for a fixed number of epochs ($E = 120$). During training, at five-epoch intervals, the model was tested on the validation test and if the current parameters yielded better results than the previous ones (based on the AUC metric), then the currently best model parameters were updated. At the end of the training, the best-known parameters were restored in order to generate predictions on the test set.

To evaluate BRIE's performance at the task of fraudster-vs-legitimate classification, 3 metrics were chosen:

- F1-macro: the unweighted mean of the F1-score of each class
- AUC: measures a binary classification model's ability to distinguish between positive and negative examples. Insensitive to class imbalance.
- Recall: the proportion of positive instances correctly identified by the classification model.

4.2 Datasets

Our experiments were conducted on two real-world datasets[*]. The YelpChi dataset [10] collects hotel and restaurant reviews, filtered (spam) and recommended (legitimate) on Yelp. It treats reviews as nodes to which it assigns 32 handcrafted features. The dataset consists of three relations: 1) R-U-R: connects reviews posted by the same user; 2) R-S-R: connects reviews under the same product with the same star rating (1-5 stars); 3) R-T-R: connects two reviews under the same product posted in the same month.

The Amazon dataset [8] includes product reviews under the Musical Instruments category. It treats users as nodes which are endowed with 25 handcrafted features. The dataset consists of 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 text similarities (measured by TF-IDF) among all users. The statistics of these two datasets are shown in Table 1. The two chosen datasets are widely considered as benchmarks, and have been used by several studies in order to measure performance of fraud detection models.

4.3 Results and Comparisons with prior works

Several GNN models were compared at the task of fraud detection. The results are summarized in Table 2[*]. To ensure robustness and reliability of the results, the proposed BRIE model was evaluated five times using different random seeds. The reported performance metrics in Table 2 represent the averages obtained from these five independent runs. The "basic" benchmark GNN model, GCN, underperforms all other baseline models and BRIE, in both datasets. This

[*] found at https://github.com/YingtongDou/CARE-GNN/tree/master/data

[*] The experimental results were obtained directly from the authors' original papers and, for CACO-GNN, from the author's source code contribution.

Table 1. Dataset statistics

Dataset	#Nodes (Fraud%)	Relation	#Edges
YelpChi	45,954 (14.5%)	R-U-R	49,315
		R-T-R	573,616
		R-S-R	3,402,743
		ALL	3,846,979
Amazon	11,944 (9.5%)	U-P-U	175,608
		U-S-U	3,566,479
		U-V-U	1,036,737
		ALL	4,398,392

can be attributed to the transformation of the multi-relation graph to a single-relation graph which provokes significant semantic information loss. In the larger YelpChi dataset, our proposed BRIE model significantly outperforms all others in all metrics, with (approximately) a 14-point improvement in F-1 score, a 5-point improvement in AUC and an 8-point improvement in Recall over the next best model (IMINF). In the Amazon dataset, BRIE surpasses every model except IMINF (whose performance it closely approaches). Notably, among studies utilizing contrastive learning techniques, BRIE, CACO-GNN, and IMINF consistently achieve superior results in both datasets, compared to those that do not employ such methods.

Table 2. Comparison of different GNN models on Amazon and YelpChi datasets. Bold entries correspond to the highest scores achieved. The list of works cited is not exhaustive; additional competitive models (which are tested on the same datasets as ours but nevertheless do not perform better than BRIE) include [11,6].

Method	**Amazon**			**YelpChi**		
	F1-macro	**AUC**	**Recall**	**F1-macro**	**AUC**	**Recall**
GCN	55.91	75.25	67.81	47.81	54.98	53.87
CARE-GNN	-	89.73	88.48	-	75.70	71.92
PC-GNN	89.56	95.86	-	79.87	63.00	-
CACO-GNN	89.79	97.64	92.83	74.19	87.12	77.65
IMINF	**94.86**	98.56	**97.01**	78.12	94.13	87.27
BRIE	94.41	**98.92**	95.20	**92.40**	**99.16**	**95.33**

4.4 Conclusions

We proposed BRIE, a GNN-based model which performs well in fraud detection tasks on real-world datasets, particularly excelling in the YelpChi dataset where it achieves state-of-the-art effectiveness. By leveraging GNNs, supervised contrastive learning, additional integrated structural features, and synthetic fraud generation, BRIE effectively addresses the challenges of fraudster camouflage and class imbalance. Our experimental results showcase significant improvements in key metrics (F1-score, AUC, Recall) compared to existing models, positioning BRIE as a promising approach for enhancing fraud detection in attributed networks. Opportunities for further work include exploring the model's ability to dynamically adapt to evolving fraud tactics and patterns in real-time scenarios, enhancing its fraud detection capabilities in dynamic network environments.

References

1. Deng, Z., Xin, G., Liu, Y., Wang, W., Wang, B.: Contrastive graph neural network-based camouflaged fraud detector. Information Sciences **618**, 39–52 (2022)
2. Dou, Y., Liu, Z., Sun, L., Deng, Y., Peng, H., Yu, P.S.: Enhancing graph neural network-based fraud detectors against camouflaged fraudsters. In: Proc. 29th ACM Int'l Conf. on information & knowledge management. pp. 315–324 (2020)
3. Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., Krishnan, D.: Supervised contrastive learning. Advances in neural information processing systems **33**, 18661–18673 (2020)
4. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
5. Le-Khac, P.H., Healy, G., Smeaton, A.F.: Contrastive representation learning: A framework and review. Ieee Access **8**, 193907–193934 (2020)
6. Li, P., Yu, H., Luo, X., Wu, J.: Lgm-gnn: A local and global aware memory-based graph neural network for fraud detection. IEEE Transactions on Big Data (2023)
7. Liu, Y., Ao, X., Qin, Z., Chi, J., Feng, J., Yang, H., He, Q.: Pick and Choose: A GNN-Based Imbalanced Learning Approach for Fraud Detection. In: Proc. of the Web Conference 2021. p. 3168–3177. WWW '21, Association for Computing Machinery, New York, NY, USA (2021)
8. McAuley, J.J., Leskovec, J.: From amateurs to connoisseurs: Modeling the evolution of user expertise through online reviews. CoRR **abs/1303.4402** (2013)
9. Pourhabibi, T., Ong, K.L., Kam, B.H., Boo, Y.L.: Fraud detection: A systematic literature review of graph-based anomaly detection approaches. Decision Support Systems **133**, 113303 (2020)
10. Rayana, S., Akoglu, L.: Collective Opinion Spam Detection: Bridging Review Networks and Metadata. In: Proc. 21th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. p. 985–994. KDD '15, Association for Computing Machinery, New York, NY, USA (2015)
11. Shi, F., al.: H2-fdetector: A gnn-based fraud detector with homophilic and heterophilic connections. In: Proc. of the ACM Web Conference 2022. pp. 1486–1494 (2022)
12. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
13. Waikhom, L., Patgiri, R.: A survey of graph neural networks in various learning paradigms: methods, applications, and challenges. Artificial Intelligence Review **56**(7), 6295–6364 (2023)
14. Wang, P., Han, K., Wei, X.S., Zhang, L., Wang, L.: Contrastive learning based hybrid networks for long-tailed image classification. In: Proc. IEEE/CVF conf. on computer vision and pattern recognition. pp. 943–952 (2021)
15. Wang, X., Liu, Z., Liu, J., Liu, J.: Fraud detection on multi-relation graphs via imbalanced and interactive learning. Information Sciences **642**, 119153 (2023)
16. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world'networks. nature **393**(6684), 440–442 (1998)
17. Wu, Z., al.: A comprehensive survey on graph neural networks. IEEE transactions on neural networks and learning systems **32**(1), 4–24 (2020)
18. Zhao, J., Song, Y., Liu, F., Deng, Y.: The identification of influential nodes based on structure similarity. Connection science **33**(2), 201–218 (2021)
19. Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., Sun, M.: Graph neural networks: A review of methods and applications. AI open **1**, 57–81 (2020)