Yahan Cong, Jade Yu

Prof. Shan Du

DATA 586 Project

April 28, 2024

PC-GNN on Fraud Detection

SUMMARY

From credit card fraud and spam emails targeting individuals' financial security to market fraud, which could undermine market integrity, fraud permeates everywhere in our modern-day life. Fraud detection is crucial to protect individuals, businesses and markets. In recent years, fraud detection approaches with graph-structured data have drawn a lot of attention due to the abundant relational information of the graph data. Traditional Graph Neural Network(GNN) based algorithms used for fraud detection include GCN and GraphSAGE. However, these methods suffer from poor performance when the label distribution of nodes is heavily skewed, which means they perform poorly for the minority but more important class, namely the fraudsters. This class imbalance problem is very common in real-life situations. To address this problem, PC-GNN is introduced. In this project, we replicated the result of PC-GNN on the benchmark dataset YelpChi and applied it to two other real-life datasets, of which the according graph structure is constructed by us. We then employed grid search to fine-tune the hyperparameters of the three runs and managed to compare the results with that of the two traditional GNN methods, which are GCN and GraphSAGE. The outcome shows that PC-GNN is superior to the two GNN models on the first two datasets. However, when it comes to the third dataset, we saw a drop in performance, for which we have several speculations.

INTRODUCTION

The task of fraud detection is undertaken in many areas including finance, health and management. Many techniques have been developed, among which graph-based ones have been focused on recently with graph data becoming ubiquitous. There have been numerous high-impact applications of GNN for fraud detection. The basic assumption of graph-based fraud detection is that there exist rich behavioral interactions between nodes, whether benign or malignant. However, the number of fraudsters can be far less than that of benign ones in real life. For example, in the real-world review dataset YelpChi from Yelp.com, 14.5% of the reviews are spam while others are regarded as recommended reviews. As a result, the performances of traditional GNN models are very likely to be disappointing.

There are three major challenges when designing GNN targeting class imbalance for fraud detection. First, the fraudster nodes may fabricate noisy information to make them hard to identify, embodying redundant link information. For example, spammers would employ benign accounts to post their spam reviews so that there will be many links between the spam review and the benign users, and the spammers would conceal themselves among benign users. Many feature-based or label-based similarities would fail to identify this kind of noisy neighbors since a fraudster could be in a close Euclidean distance from a benign one but their labels would be different. Second, there is a lack of necessary link information for fraudsters due to their camouflage. For example, fraudsters would avoid trading with each other in order not to be detected together. Last, in the imbalance settings, most neighbors of a center node will be classified as the majority class in the step of message aggregation of GNNs, resulting in the features of the minority class being diluted.

There are two directions of the potential solutions, namely re-sampling and re-weighting. Re-sampling methods balance the number of examples by oversampling the minority class or under-sampling the majority class. Re-weighting methods assign different weights to different classes or even different samples by cost-sensitive adjustments or meta-learning-based methods. When it comes to the class imbalance problem on graphs, DR-GCN is a pioneer work. DR-GCN is a dual-regularization graph convolutional network to handle multi-class imbalance graph representation. However, its performance on large graphs is seen as not satisfying enough. GraphSAGE is a sampling-based method which uses node sampling. It does not explore label distribution when sampling and keeps a fixed size of the neighborhood to deal with abundant links in the graph. As a result, for nodes with a larger size of the neighborhood, the corresponding information loss will decline its performance. To address the above challenges. PC-GNN brings a label-balancing sampler to pick nodes and edges to train and a neighborhood sampler to choose neighbors with a learnable parameterized distance function so that it can construct a relatively balanced subgraph.

In this project, we use GCN and GraphSAGE as baseline models to compare with PC-GNN. PC-GNN brings its unique advantages as shown in our project.

METHODOLOGY

PC-GNN (Pick and Choose GNN) addresses the severe imbalance between fraud and benign class. It transforms the target objects intended for prediction into nodes, and constructs relatively balanced subgraphs for every relation in the graph with its unique tools: Pick-label-sampler and Choose-neighbor-sampler.

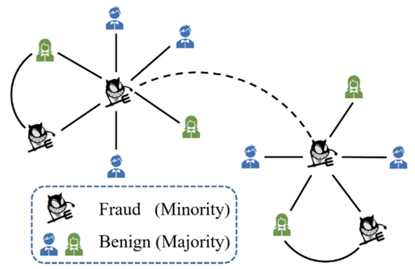
PC-GNN: PICK

To achieve a relatively balanced sub-graph for training, PC-GNN utilizes its pick-label-sampler to randomly select nodes from the graph as starting points for subgraph creation. The selection follows two principles: Firstly, it aims to increase the proportion of the minority class, which are the fraud nodes, in the sampling process. As illustrated by the selection formula below, LF(C(v)) represents the proportion of fraud and benign labels among the total nodes in the graph. Thus, the pick-label-sampler ensures that the fewer fraud nodes among the total nodes, the more likely they are to be selected during the sampling process. This ensures that more fraud nodes are included in the subgraph during the picking stage.

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

Secondly, nodes with a higher number of edges and more connected 1-hop neighbors are preferentially selected during the sampling process.$\|\hat{A}(:,v)\|^2$ represents the number of edges of a node. Nodes with more 1-top neighbors often contain richer information due to their complex connections and exert a greater influence on the overall graph structure. It facilitates a more balanced sampling process, helping to mitigate any potential biases in the subgraph across various dimensions.

PC-GNN: CHOOSE, undersampling and oversampling



After the picking step, a relatively balanced subgraph is constructed, consisting of the selected nodes and their 1-hop neighbors. As fraudsters often camouflage their activities within normal interactions, PC-GNN employs re-sampling to further adjust the number of neighbors in the graph to balance the subgraph and better capture the network patterns of fraud. The core of the choosing process is based on the distance and label between nodes, allowing for undersampling and oversampling of nodes. Using the weighted learnt embeddings D(h(v)) of each node, PC-GNN calculates the predicted probability difference D(v, u), which serves as the L1 distance between nodes. Based on the distances between nodes and their respective neighbors, PC-GNN performs undersampling within the subgraph, eliminating all unnecessary neighbors to reduce redundant connections between nodes.

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

In addition to reducing benign neighbors, Another question is: frauds tend to minimize their interactions. The lack of direct link causes transfer learning to become less effective because the information will dilute as it passes through other neighbors. To address this issue, PC-GNN implements an oversampling strategy for fraud nodes within the subgraph, creating edges between fraud nodes to enhance connectivity. In this process, PC-GNN searches for all fraud nodes within the subgraph and uses their pairwise distances D(u, v) to add nearby frauds as oversampling neighbors to the current fraud node's neighbor list. By creating edges between fraud nodes, learning network patterns from other frauds becomes easier and more correct, thus enhancing prediction accuracy. Therefore, a relatively balanced subgraph is achieved after undersampling all nodes in the subgraph and specifically oversampling fraud nodes.

Majority (benign): undersampling

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

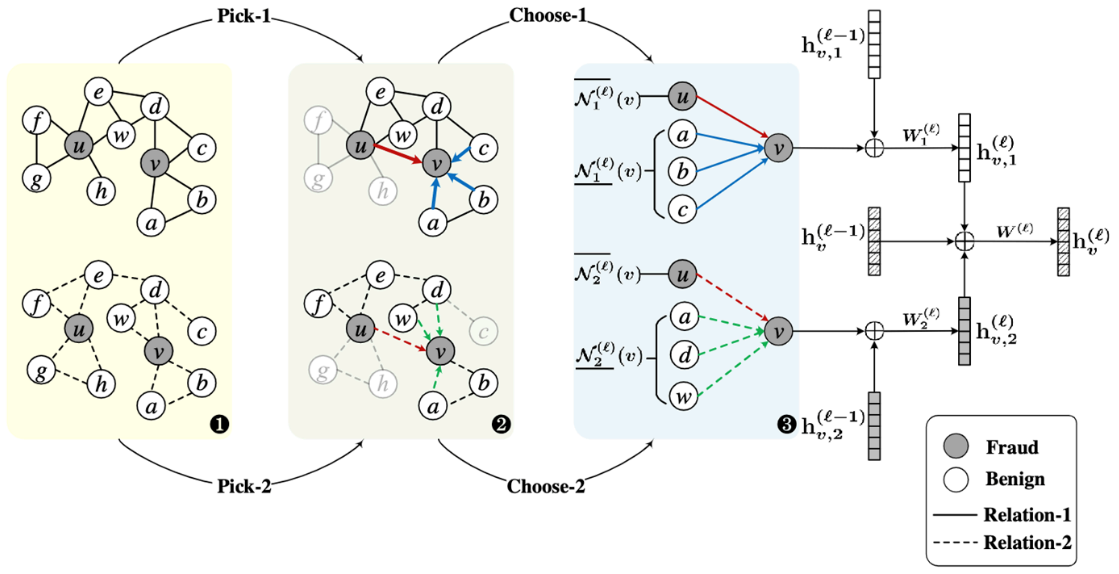
Minority: Fraud: both undersampling and oversampling:

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

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

$$\mathcal{N}_r^{(\ell)}(v) = \underline{\mathcal{N}_r^{(\ell)}}(v) \cup \overline{\mathcal{N}_r^{(\ell)}}(v)$$

PC-GNN: Aggregate



We have previously demonstrated how to create a relatively balanced graph by picking and choosing in a single relationship. In GNN, we often construct the graph structure using multiple data relationships. Consequently, we create relatively balanced sub-graphs for each of them and combine the nodes' learned embeddings from various relations. Firstly, we concatenate node's previous learnt embedding h(v,r) with the mean of its neighbors' previous learnt for one relationship r.

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

After single relationship information aggregation, we concatenate it with node's learnt embedding on other relationships.

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

We will get the PC-GNN's binary classification result by transferring the final h(v) by an MLP.

PC-GNN Training

The cross-entropy loss in PC-GNN consists of two parts: one part is the loss of distance resulting from neighbor-choosing, and the other part is the loss of prediction caused by MLP based on the node's learnt embedding. The total loss is $L = L_{gnn} + \alpha L_{dist}$. We use a balance parameter alpha to adjust the total loss through backward propagation.

$$\mathcal{L}_{\text{dist}} = -\sum_{\ell=1}^{L}\sum_{r=1}^{R}\sum_{v\in\mathcal{V}}\left[y_v \log p_{v,r}^{(\ell)} + (1-y_v)\log\left(1-p_{v,r}^{(\ell)}\right)\right]$$

$$p_{v,r}^{(\ell)} = \mathrm{D}_r^{(\ell)}\left(\mathbf{h}_{v,r}^{(\ell)}\right)$$

$$\mathcal{L}_{\text{gnn}} = -\sum_{v\in\mathcal{V}}\left[y_v \log p_v + (1-y_v)\log(1-p_v)\right]$$

$$p_v = \mathrm{MLP}\left(\mathbf{h}_v^{(L)}\right)$$

EXPERIMENTS

In this project, we have three research questions. First, we want to replicate the result of PC-GNN. By conducting extensive experiments on one public benchmark dataset and two real-world datasets, we want to verify the effectiveness and superiority of PC-GNN. Second, what is the model performance with respect to different hyperparameters. Third, we want to look into whether there is a difference in the performance of PC-GNN on datasets with different graph structures. If so, what could be the possible reasons?

Dataset

There are three datasets that we used.

The first is the classic YelpChi dataset. With this dataset, we want to detect fraudulent positive reviews on Yelp for 67,395 reviews for hotels and restaurants in Chicago by 38,063 reviewers. There exist 13.23% filtered reviews by 20.33% spammers in this dataset. The nodes in the graph of the dataset are reviews with 32-dimensional features. We conclude that this is a relation-driven graph structure which has three relations:

1) R-U-R denotes the reviews posted by the same user;
2) R-S-R denotes the reviews under the same product with the same star rating;
3) R-T-R denotes the reviews under the same product posted in the same month.

The FDCompCN dataset is used for detecting financial statement fraud of companies in China. The data is obtained from the China Stock Market and Accounting Research (CSMAR) database. Samples between 2020 and 2023 are selected, including 5,317 publicly listed Chinese companies traded on the Shanghai, Shenzhen, and Beijing Stock Exchanges. The nodes in the graph of the dataset are companies with 57-dimensional features. There exist 10.5% fraudulent nodes. We conclude that this is a behavior-linked graph structure which has three relations:

1) C-I-C represents the companies that have investment relationships;
2) C-P-C that connects companies and their disclosed customers;
3) C-S-C represents companies and their disclosed suppliers.

Finally, the AlibabaLoan dataset contains 7,000 samples from the loan dataset of Alibaba Cloud ‘Tianchi’, which has the information of both the borrower and lender and the loan details. We selected these samples from the original dataset, the size of which is 80,000 and the proportion of bad loans is 19.3%. The nodes of the graph are fraudulent loans. We transfer this Alibaba loan dataset as an attribute-sharing graph structure with four relations inside:

1) T-A-T represents loan information with similar amounts;
2) T-G-T denotes loan information with the same credit sharing;
3) T-P-T denotes loan information with the same purpose;
4) T-R-T denotes loan information from the same region.

Compared Methods

We compared the performance of PC-GNN to two other GNN models, which are GCN and GraphSAGE respectively.

- GCN: graph convolution network achieved by a localized first-order approximation of spectral graph convolutions.
- GraphSAGE: an inductive GNN model based on a fixed sample number of the neighbor nodes.
- PC-GNN: the framework which we want to scrutinize carefully. It conducts over-sampling on minority nodes and under-sampling on majority nodes.

Metrics

We used three metrics to measure the performance of the three models. F1-macro, AUC and GMean are adopted.

F1-macro is the unweighted mean of the F1 score of each class. It is a useful metric to evaluate the overall performance of a classifier across all classes, especially when class imbalance exists.

AUC is the area under the ROC Curve. AUC is also useful for evaluating the overall performance of the model.

GMean calculates the geometric mean of True Positive Rate (TPR) and True Negative Rate (TNR). It comes into handy when we want to test whether the model performs well on both majority and minority classes without favoring either.

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

Fine-tuning

The hyperparameters of PC-GNN that we tuned include train and test ratio, rho value, learning rate, weight decay, number of epochs and valid epochs. Both the model on fraud company and bad loan dataset went through 32 configurations. Since the performance of PC-GNN on YelpChi dataset is already very good, it only went through 8 configurations.

Results and Analysis

The performance of the three models on three datasets is as follows.

Table 1: Performance of GCN, GraphSAGE and PC-GNN on Three Datasets

Dataset	YelpChi			FDCompCN			AlibabaLoan		
Metric	F1-macro	AUC	GMean	F1-macro	AUC	GMean	F1-macro	AUC	GMean
GCN	0.4037	0.5375	0.4395	0.2739	0.6198	0.4298	0.4842	0.5383	0.5088
GraphSAGE	0.5883	0.7876	0.7078	0.4570	0.7321	0.6230	0.2034	0.6623	0.2047
PC-GNN	0.6871	0.8182	0.7216	0.5425	0.7829	0.6811	0.5648	0.6168	0.5913

PC-GNN is superior to the two previously proposed GNN models. It addresses the real-world problem of class imbalance and achieves great performance on all three datasets. However, it is concerning for us that there is a subtle drop in the performance of the models on the third dataset. Even the result of PC-GNN is far from perfect. Although its high score in GMean suggests that the classification result is impressive, its AUC score is lower than that of GraphSAGE. Regarding this problem, we have several speculations. We suspect that the graph structure of the third dataset that we constructed has too many node-to-node connections, resulting in the personality of nodes being concealed. The other assumption is that PC-GNN does not work very well on graphs constructed with attribute-sharing relations compared to those with relation-driven or behavioral-linked graphs.

CONCLUSION AND FUTURE WORK

In this project, we applied PC-GNN, a GNN-based imbalanced learning method to solve the class imbalance problem in graph-based fraud detection. It takes three steps to set up the framework: pick, choose and aggregate. The center nodes are picked with a label-balanced sampler to construct a balanced sub-graph for mini-batch training. Under a parameterized distance function, the neighborhood of the minority class is over-sampled and that of the majority class is under-sampled. Messages from selected neighbors and different relations are aggregated to obtain the final representations of the target. Experiments on three datasets clearly showcase the superiority of PC-GNN compared to two other GNN frameworks.

Our future work includes exploring the connection between the performance of PC-GNN and the properties of the graph-structured data it is applied on, go through more fine-tuning configurations if possible and conducting sensitivity analysis, and finally further addressing the problem of extreme class imbalance and misclassification.

REFERENCES

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

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

[3] Leman Akoglu, Hanghang Tong, and Danai Koutra. 2015. Graph Based Anomaly Detection and Description: A Survey. Data Min. Knowl. Discov. (2015).

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

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

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

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

[8] Thomas N. Kipf and Max Welling. "Semi-Supervised Classification with Graph Convolutional Networks". In: International Conference on Learning Representations (ICLR). 2017.

[9] Yang Liu et al. "Pick and Choose: A GNN-based Imbalanced Learning Approach for Fraud Detection". In: Proceedings of the Web Conference 2021. 2021, pp. 3168–3177.

[10] Bin Wu et al. "SplitGNN: Spectral Graph Neural Network for Fraud Detection against Heterophily". In: Proceedings of the 32nd ACM International Conference on Information and Knowledge Management. CIKM '23. 2023.