Detecting Fraud With Oversampling Techniques and Sparsity Contraints

Prabina Pokharel
p2pokharel@ucsd.edu

Yandong(Dennis) Xiang
yaxiang@ucsd.edu

Jingyu Zhang
jiz036@ucsd.edu

Gal Mishne
gmishne@ucsd.edu

Yusu Wang
yusuwang@ucsd.edu

Abstract

Fraud detection is prevalent now more than ever due to the massive surge in the usage of online platforms. Many techniques exist to combat fraud; however, they often fail to capture the imbalanced class structure in data involving fraudulent activities. It's important to tackle such concern so we can harness its power to correctly predict anomalies. So, the question remains: How can we effectively detect and mitigate fraudulent activities, especially when faced with imbalanced datasets? Our research contributes to the study of such concern with a model that harnesses the strengths of many existing techniques from different domains. We propose a solution that utilizes a combination of oversampling techniques and sparsity constraints to balance and predict fraud data.

Website: https://www.prabina.me/Detecting-Fraud-With-Oversampling-Techniques-and-Sparsity-Contraints/
Code: https://github.com/yandongxiang/GNN-DSC180A

1 Introduction

In recent years, there has been a huge surge in the usage of online platforms like Reddit, Yelp, and Amazon. While the usage of such platforms has positively influenced our daily lives, fraudulent activities is amid this rapid increase in digital data volume and anonymity of online networks. Fraudulent behaviors are executed and hidden in plain sight of this vast amount of online records. Thus, enhancing fraud detection is crucial to protect us from fraudsters against harmful scams, and even criminal activities.

1.1 Discussion of Prior Work and Our Proposed Solution

With fraud detection in mind, researchers have devised models aimed at enhancing specific aspects of the task. One such paper whose goal is to detect fake news injects data into a social graph refinement component that iteratively updates the edge weights using a learnable degree correction mask (Wu and Hooi 2023). This allows for an improvement of edge noise in graph datasets and to join information with a GNN-based detector for better optimization.

Another paper whose task is to detect group frauds in e-commerce platforms utilized an autoencoder to concatenate weighted structural features with the attributes of customer vertices to improve accuracy (Yu et al. 2023).

These are two of the many techniques used to detect fraud. As we can see, there are some solutions in place to detect fraudulent activities; however, such models don't tackle the imbalanced structure of fraud datasets very well.

So, our proposed solution is the combination of 2 techniques: GraphSMOTE (Zhao, Zhang and Wang 2021) and SparseGAD (Gong et al. 2023). GraphSMOTE, although not commonly used for fraud detection, is an oversampling technique that identifies minority class nodes and creates new nodes that resemble those minority class data points, thus helping it balance. Since datasets involving fraud detection are hugely imbalanced, GraphSMOTE will help us balance the class distribution in the graph. We will then take the output of this GraphSMOTE and feed it into SparseGAD, known for anomaly detection by introducing sparsity constraints. Such constraints help highlight significant connections, and anything that substantially deviates from that will be looked into further for fraud. Upon applying these techniques, we feed in the output to our Graph Neural Networks (GNNs) to obtain a result. We will discuss our model in more detail in the Methods section of this paper.

By tackling this fraud detection task, we hope to contribute to protecting us against scams.

1.2 Datasets

We'll be implementing our model on three datasets: Amazon, Yelp, and Reddit. These datasets are obtained from the Deep Graph Library (DGL) and PyGOD. Below are the links to access these datasets:

- Amazon Dataset
- Yelp Dataset
- Reddit Dataset

1.2.1 Info on the Datasets

Amazon Dataset: This dataset includes product reviews under the Musical Instruments category. This is a binary classification task where users with more than 80% helpful votes are labeled as benign entities and those with less than 20% are labeled as fraudulent. Positive (fraudulent) to Negative (benign) ratio is 1:10.5, which shows that it's imbalanced.

Yelp Dataset: This dataset includes hotel and restaurant reviews. This is a binary classification task where it is divided into filtered (spam) and recommended (legitimate). Here, Positive (spam) to Negative (legitimate) ratio is 1:5.9, which shows that this dataset is also imbalanced.

Reddit Dataset(Liu et al. 2022): This dataset consists of Reddit posts. The node refers to the community that a certain post belongs to. The nodes are connected if the same user comments on both communities. This dataset contains 10,984 posts with an average degree of 15.3. The ratio between our training and testing dataset is 1:1. The ground truth states that there is a 3.3% anomaly (outlier), which shows that, along with the other two datasets, this is also imbalanced.

All of the datasets we choose for our model come from a reputable library (DGL and PyGOD), and have been used in many research papers. They will be useful for our model since it's tasked to fix imbalances while doing anomaly detection.

2 Methods

We are tackling this task using GNNs, or more specifically Graph Anomaly Detection (GAD). Previous works on anomaly detection exist; however, we observed that none of them attempted to create a GraphSMOTE model specifically designed for anomaly detection. This is striking because while the GraphSMOTE model performs well on imbalanced datasets, it is not designed to capture anomalous users among the common users.

GraphSMOTE is designed to add synthetic nodes to the graph to balance the dataset. In the case of graph anomalous user detection, the anomalous users only represent a small portion of the dataset. To balance out the dataset, we added synthetic anomalous nodes into the graph to create a balanced, amplified dataset. The GraphSMOTE model typically runs an edge generator to generate edges on the synthetic nodes to find their links to other nodes. To not disturb the connection between the synthetic node and other nodes, we used a specific method to both preserve node connections and create synthesized nodes.

The GraphSMOTE model alone does not capture the heterophily nature of anomalous nodes. Anomalous users often show greater dissimilarity with their connected users despite their attempt to blend into common users by establishing fake connections with other users. After

using the SMOTE method to create a balanced dataset, we implemented SparseGAD techniques to add sparsity into the graph and generate a learnable adjacency matrix (homey matrix) to identify whether the connected nodes are similar or dissimilar. In this way, we can allow our model to both address the imbalanced nature of the dataset and observe the nature of heterophily in anomalous nodes.

2.1 Synthetic Node Generation

In particular, we would use the SMOTE method to generate synthetic nodes. First, we determine which class label belongs to the minority class. In our case, the minority class is typically the anomalous users, as they generally occur less frequently than the common users. To generate synthetic nodes to amplify the minority class, we use the upsampling method to randomly replicate nodes and reproduce their connections with their neighbors. The reason we aim for identical neighbors for the synthetic nodes is to maintain a similar level of heterogeneity for the links of the minority classes. We then add randomized minor differences in the features of synthetic nodes to create distinctions between synthetic nodes and the original nodes, as in the SMOTE method. To clarify, the reason why we avoid randomized differences in link connections is that we do not want to disrupt the model's ability to identify the heterophilic nature of the anomalous nodes.

2.2 Homey Graph Generation

Instead of the original adjacency matrix, we utilized the "homey" adjacency matrix concept from the GraphSMOTE model. In this model, a single-layer GCN (Graph Convolution Neural Network) embedding is implemented to calculate the cosine similarity of the adjacent node pairs, following this function, where each item in the homey matrix H is computed with cosine similarity.

$$H_{uv} = \text{cosine similarity} = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \in [-1, 1] \tag{1}$$

Here, H_{uv} ranges from -1 to 1. When $H_{uv} = 1$, it means the node pair is homophilic; when $H_{uv} = -1$, the node pair is heterophilic. When the node pair has no relation, $H_{uv} = 0$.

2.3 Sparsification

Although we have calculated cosine similarity for the homey matrix, we still haven't addressed the nature of anomalous users camouflaging themselves among the common users. Thus, we also need the sparsification method from SparseGAD to filter out elements in the adjacency matrix. We use δ as the threshold to remove unnecessary neighbors.

$$H_{uv} = \begin{cases} 0 & \text{if } H_{uv} \leq \delta \\ H_{uv} & \text{otherwise} \end{cases} \tag{2}$$

After setting δ as a threshold, SparseGAD further utilizes KNN to limit the number of necessary connections. Finally, the GAD-oriented regularization developed in SparseGAD is employed to further sparsify the graph, preventing faulty links from hindering the model's ability to distinguish anomalous users.

2.4 Learning Objective

In the end, we use our model to classify the users into normal users and abnormal users based on the node features and neighborhood similarity. We use the following ROC-AUC score.

$$\text{ROC\_AUC} = \frac{1}{m(m-1)} \sum_{i=1}^{m} \sum_{j=1, j \neq i}^{m} AUC_{ij} \tag{3}$$

We use the ROC-AUC score as the metric to measure whether our model outperforms the baseline models.

3 Implementation Details

3.1 Main Paths

Our approach to detecting fraud involves three main paths, each of which utilizes a different preprocessing technique before applying GNN models (such as GCN, GAT, or GraphSage).

- **Baseline Path:** Here, we directly input the dataset into the chosen GNN model. This path serves as our baseline for comparison against the other paths.
- **GraphSMOTE Path:** In this path, we apply the GraphSMOTE technique discussed in the Methods section before feeding it into the chosen GNN model. Again, GraphSMOTE is an oversampling technique, which helps address the imbalance in the dataset by generating synthetic nodes.
- **GraphSMOTE with New Adjacency Matrix Path:** In this path, we utilize GraphSMOTE and the principles of SparseGAD. Here, the SparseGAD receives the balanced dataset generated by GraphSMOTE and incorporates consine similarity matrix and sparsity constraints which helps highlight significant connections and anomalies in the graph.

3.2 Hyperparameter Tuning

After applying each path, we perform hyperparameter tuning on the models, focusing primarily on adjusting the learning rate to optimize performance. This step ensures that the model learns the most relevant features from the dataset and improves its ROC-AUC score. Upon experimenting with different implementations, we found that learning rate matters the most in improving our metrics. We found that a learning rate of 0.001 works best for the Amazon dataset and a learning rate of 0.00001 works best for Yelp and Reddit datasets. We run 300 epochs for both Pretrain and Finetune models. In both cases, we select one model from GCN, GAT, and GraphSage and apply one layer of it in the encoder and 2 layers of it with 64 nodes in the hidden layer in the classifier.

3.3 Decision Stage

Finally, we use the trained GNN model to classify nodes as either fraudulent or non-fraudulent based on their features and neighborhood structure. We evaluate the model's performance using the ROC-AUC score based on its ability to correctly identify anomalies in the dataset.

4 Results

Now, let's compare the models' performance on each of these datasets.

Table 1: Performance of different models on various datasets

Models	**Yelp (%)**	**Reddit (%)**	**Amazon (%)**
GCN	73.01	55.26	81.26
GAT	72.22	62.93	85.07
GraphSage	**76.94**	**64.86**	84.82
GraphSmote+GCN	63.37	45.02	73.94
GraphSmote+GAT	53.74	55.94	75.56
GraphSmote+GraphSage	64.08	56.46	89.83
Modified GraphSmote+GCN	58.83	43.80	74.65
Modified GraphSmote+GAT	54.08	50.00	78.86
Modified GraphSmote+GraphSage	67.36	56.12	**90.02**

As we can see in Table 1, the GraphSage model performs best on Yelp and Reddit datasets, and the GraphSage with Modified GraphSMOTE performs best on the Amazon dataset.

We first aim to explain why GraphSAGE outperforms GCN and GAT on the Yelp and Reddit datasets, while achieving a similar ROC-AUC on the Amazon dataset. Our intuition is that this discrepancy arises from the structural differences between the datasets. As observed in Figure 1, both the Yelp and Reddit datasets exhibit relatively sparser structures, whereas the Amazon dataset presents a more condensed structure. This condensed structure in

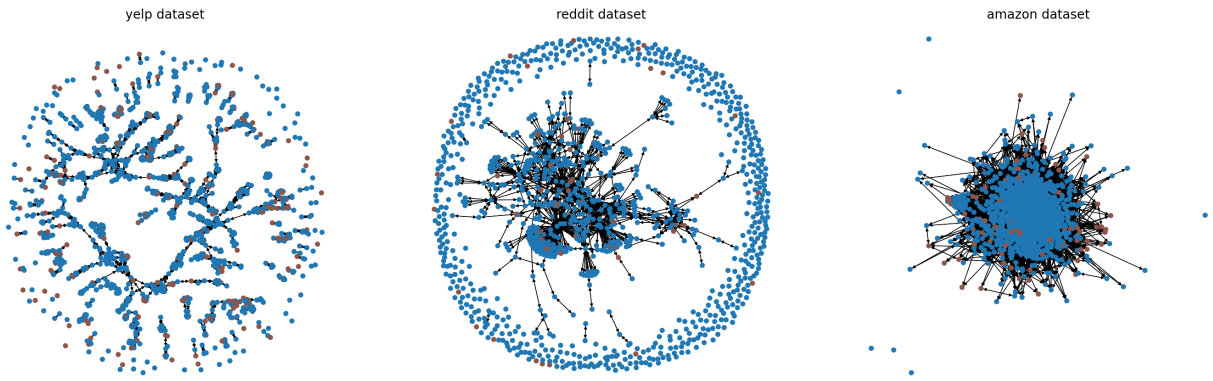

Figure 1: General Structure of the Datasets

the Amazon dataset allows the weights in the GAT model on each neighbor's features to have greater significance compared to those in the Yelp and Reddit datasets. On the other hand, the GraphSAGE model samples and aggregates from the neighboring nodes. On the Yelp and Reddit datasets, the simplicity of the GraphSAGE method facilitates better model tuning and results in improved convergence, thereby enabling the GraphSAGE model to outperform the GAT model on these datasets.

The main reasons why we believe our GraphSAGE performs well on the Yelp and Reddit datasets, and GraphSAGE with Modified GraphSMOTE on the Amazon dataset, are related to the degrees of connectivity and the number of connected nodes. Below, we present the degree distribution of the datasets.

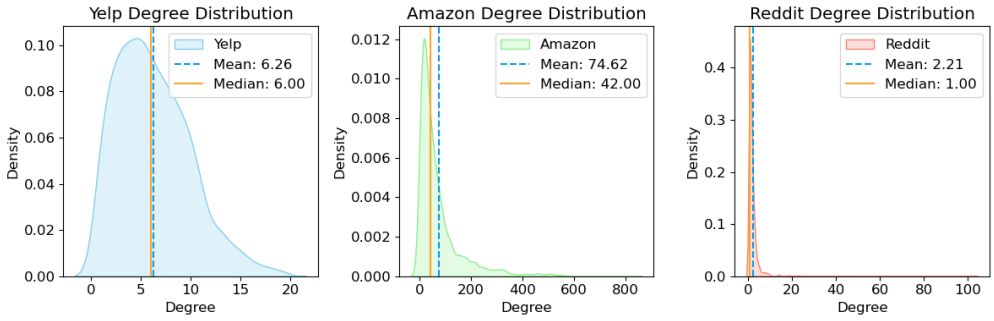

Figure 2: Degree Distribution of the Datasets

As we see in Figure 2, in the Yelp and Reddit datasets, the median number of degrees is lower (6 and 1 respectively) vs. Amazon (which has 42). This discrepancy may reflect variations in the connectivity and structural complexity of the datasets, which directly influence the performance of fraud detection models. Furthermore, Yelp and Reddit datasets have more nodes with only self-loops (106 and 625 respectively) compared to Amazon (which has 6).

This lack of connectivity in the Yelp and Amazon datasets limits the potential for synthetic nodes generated by the SMOTE method to establish meaningful connections within the graph. As a result, the ability of the SMOTE method to effectively balance the class distri-

bution while maintaining dataset structure may be constrained in Yelp and Reddit datasets whereas flourished in the Amazon dataset. Thus, the GraphSMOTE and the Modified GraphSMOTE models were unable to surpass the performance of the baseline GraphSage model.

Similarly, in the Amazon dataset, both the GraphSMOTE and Modified GraphSMOTE models, when paired with the GCN and GAT classifiers, failed to outperform the baseline models. However, our GraphSMOTE and Modified GraphSMOTE models that used the GraphSage classifier improved the ROC-AUC score by more than 5%. This suggests that the sampling and aggregating method employed by GraphSAGE is less prone to overfit to the added synthetic users, while GCN and GAT are more likely to be negatively affected by the changes in data structure introduced by GraphSMOTE. However, due to the variance in the final ROC-AUC score, there is not a significant difference between the GraphSMOTE and Modified GraphSMOTE models with the GraphSAGE classifier. Nevertheless, there is a significant increase in the ROC-AUC score between the GraphSMOTE and Modified GraphSMOTE models with the GAT classifier. It seems that GAT benefits from the sparsified cosine-similarity matrix, which allows the GAT classifier to focus more on determining the cosine similarity between nodes when weighing neighboring nodes. Thus resulting in a 3% increase in the ROC-AUC score of the Modified GraphSMOTE model with GAT classifier compared to the original GraphSMOTE model.

This disparity in performance highlights the importance of tailoring the model to the characteristics of the dataset.

5 Conclusion

In this paper, we introduced GraphSMOTE and SparseGAD together to perform graph-based anomaly detection on imbalanced datasets. Under GraphSMOTE, we added synthetic nodes to the graph to balance the dataset. With SparseGAD, we accounted for both heterophilic and homophilic dependencies among nodes to streamline graph structure and collectively refine distinctive node representations for detecting anomalies. Our experiments show that, due to the lack of connectivity in Yelp and Reddit nodes, the modified model is unable to surpass the performance of the GraphSage model. It however showed that modified GraphSMOTE with GraphSage performs best on the Amazon dataset for detecting fraudulent users.

In the future, we would like to refine the model to accommodate variability in datasets such as degrees of connectivity and number of connected nodes.

References

Gong, Zheng, Guifeng Wang, Ying Sun, Qi Liu, Yuting Ning, Hui Xiong, and Jingyu Peng. 2023. "Beyond Homophily: Robust Graph Anomaly Detection via Neural Sparsification." In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI*.

Liu, Kay, Yingtong Dou, Yue Zhao, Xueying Ding, Xiyang Hu, Ruitong Zhang, Kaize Ding, Canyu Chen, Hao Peng, Kai Shu, Lichao Sun, Jundong Li, George H Chen, Zhihao Jia, and Philip S Yu. 2022. "BOND: Benchmarking Unsupervised Outlier Node Detection on Static Attributed Graphs." In *Advances in Neural Information Processing Systems 35*. Curran Associates, Inc. [Link]

Wu, Jiaying, and Bryan Hooi. 2023. "DECOR: Degree-Corrected Social Graph Refinement for Fake News Detection." In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

Yu, Jianke, Hanchen Wang, Xiaoyang Wang, Zhao Li, Lu Qin, Wenjie Zhang, Jian Liao, and Ying Zhang. 2023. "Group-based Fraud Detection Network on e-Commerce Platforms." In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. New York, NY, USA Association for Computing Machinery. [Link]

Zhao, Tianxiang, Xiang Zhang, and Suhang Wang. 2021. "GraphSMOTE: Imbalanced Node Classification on Graphs with Graph Neural Networks." In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. ACM. [Link]