Graph Neural Network-Based Anomaly Detection System: Insights into Credit Card Transaction Anomaly Analysis

Boram Kim[1] **Guebin Choi**[1]
[1]Department of Statistics, Jeonbuk National University

Introduction

Credit card ubiquity brings challenges in detecting unauthorized use, highlighting the need for vigilant monitoring by issuers. Traditional fraud detection, dependent on analyzing high transaction amounts, often fails to capture complex fraudulent patterns. A shift towards graph-based analytics is emerging, with bipartite (as shown in Figure 1) and tripartite graphs (illustrated in Figure 2) being primary methods for mapping customer-merchant transactions [2].

Bipartite graphs, commonly utilized in fraud detection, are challenged by their inability to efficiently detect anomalies on an individual transaction basis, as they require grouping data by customers and merchants, which may obscure specific transaction details. Tripartite graphs, on the other hand, though they facilitate transaction-level analysis, often face significant computational inefficiencies and complexities, making them less practical for large-scale applications and potentially reducing model performance due to the intricate nature of the connections they map.

Figure 1: An illustration of a bipartite graph showing the relationship between customers (colored figures) and stores (shopping carts) where lines indicate the connections between them.

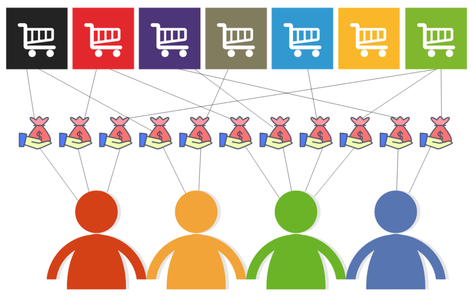

Figure 2: A representation of a tripartite graph, depicting the three-way relationships among stores (shopping carts), transactions (gift icons), and customers (colored figures), with dashed lines connecting each customer to their respective transactions at various stores.

Our method uses time differences between transactions for improved graph connectivity, offering a scalable and adaptable fraud detection approach with the following advantages:

- Advantages over non-graph-based methods:
  - It counters the over-reliance on transaction amounts for detecting fraud.
- Advantages over existing graph-based methods:
  - It is more predictive and less prone to overfitting without merchant data.
  - It provides computational efficiency over other graph models.
  - It simplifies graph analysis with straightforward edge structures.

Data

The dataset under consideration contains credit card transaction records from European cardholders captured in the month of September 2013. It encompasses transactions that were conducted across a network of 800 merchants and includes data from 1,000 individual cardholders. Originally, this dataset is composed of 23 distinct features that offer various insights into each transaction. However, for the purposes of this study, we have selectively utilized only a subset of these features that are deemed most relevant to our analysis objectives.

Table 1: A list of data variables used in the study

Variable	Description
`index`	Identifier for each row, row number.
`cc_num`	The customer's credit card number
`trans_date_and_time`	Trading Hours
`amt`	Transaction amount
`is_fraud`	Indicate whether the transaction is fraudulent (0: legitimate, 1: fraudulent)

Proposed Method

Let's say the given data is $\mathbf{X}, \mathbf{y}$, where $\mathbf{X}$ is a matrix with N rows and $\mathbf{y}$ is a vector of length N. $\mathbf{y}$ contains labels indicating the presence of fraud, while $\mathbf{X}$ represents the design matrix necessary for predicting $\mathbf{y}$. We interpret the given data as a graph. Let $\mathcal{I}$ be the set of `cc_num`. Moreover, let $\mathcal{T}_i, i \in \mathcal{I}$, be the set of transaction times for the ith customer. The given data can be expressed as $\mathcal{D} := \{(\mathbf{X}_{i,t}, \mathbf{y}_{i,t}) : i \in \mathcal{I}, t \in \mathcal{T}_i\}$. To represent the graph structure, we need to define nodes and edges. The set of nodes is defined as $\mathcal{V} = \{v_{i,t} : i \in \mathcal{I}, t \in \mathcal{T}_i\}$. Note that $|\mathcal{V}| = \sum_{i\in\mathcal{I}} |\mathcal{T}_i| = N$. The links between data points are defined as $\mathcal{E} = \bigcup_{i\in\mathcal{I}}\{(v_{i,t}, v_{i,s}) : t, s \in \mathcal{T}_i \text{ and } t \neq s\}$, considering connections only within the same customer. For a fixed $i \in \mathcal{I}$, the (t, s)-th elements of $\mathbf{W}_i$ are defined as $\exp(\frac{-|t-s|_2^2}{\theta})$ where $t, s \in \mathcal{T}_i$, and the $N \times N$ weight matrix $\mathbf{W}$ is defined as the following block matrix:

$$\mathbf{W} = \begin{bmatrix} \mathbf{W}_1 & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_2 & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{W}_3 & \dots & \mathbf{0} \\ \dots & \dots & \dots & \dots & \dots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{W}_{|\mathcal{I}|} \end{bmatrix}$$

Note that if $\mathbf{W} = \mathbf{I}$, it implies a classical tabular data structure rather than a graph data structure. $\mathbf{A}$ is the adjacency matrix corresponding to $\mathbf{W}$.

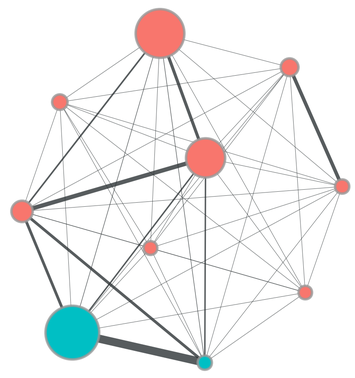

Figure 3: The graph representation corresponds to the credit card number 4.503100e+18, associated with the customer named Katherine Tucker.

Figure 3 visualizes the data as understood by the proposed method. The figure visualizes the graph for a fixed i. The size of the nodes represents the volume of transactions. Transactions that are closer in time are represented with darker weights. The color of the nodes indicates whether the transaction is fraudulent, with fraudulent transactions depicted in blue.

To predict $\mathbf{y}$, we stacked graph convolution layers [1] as follows:

$$\mathbf{H}^{(l+1)} = \sigma\big(\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}\mathbf{H}^{(l)}\mathbf{\Theta}^{(l)})\big)$$

Here, $\mathbf{D}$ is the degree matrix of $\mathbf{A}$, $\mathbf{\Theta}^{(l)}$ represents the learnable parameters, and σ denotes the activation function. $\mathbf{H}^{(l)}$ is the input for the lth layer, and $\mathbf{H}^{(l+1)}$ is the output, where for $l = 0$, $\mathbf{H}^{(0)} = \mathbf{X}$, and the output of the final layer is defined as $\mathbf{y}$.

Results & Discussions

Results

To compare models, the performance of existing graph-based methods was evaluated against the proposed approach. Additionally, for a broader comparison with non-graph-based techniques, tree-based methodologies known for their exceptional performance in tabular data analysis, such as XGBoost, LightGBM, and CatBoost, were also included. The results are as follows. However, among the graph-based methods, the tripartite was excluded due to poor fit performance, even after hyperparameter tuning (According to [2], the F1 score was around 55).

Table 2: Summary of Classification Results

Method	Graph-based	F1-score	Precision	Recall
XGBoost		88.05	88.70	87.41
LightGBM		88.52	89.38	87.67
CatBoost		88.65	89.52	87.80
GNN (bipartite)	✓	72.91	69.88	76.22
Proposed	✓	**90.82**	**86.25**	**95.91**

Discussions

Below is a summary of the analysis of the results.

- Tree-based models such as XGBoost, LightGBM, and CatBoost generally outperformed algorithms based on bipartite and tripartite graphs.
- However, considering the performance variations due to model differences after representing data with bipartite and tripartite graphs, it cannot be concluded that graph-based methods are unequivocally incorrect.
- The proposed method outperformed others by effectively utilizing time data. For example, if transactions at 11:00 and 11:30 are fraudulent, one at 11:15 is likely also fraudulent, a pattern our method captures well.

References

[1] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[2] Claudio Stamile, Aldo Marzullo, and Enrico Deusebio. *Graph Machine Learning: Take graph data to the next level by applying machine learning techniques and algorithms*. Packt Publishing Ltd, 2021.