Graph Machine Learning For Credit Card Fraud Detection

Fernando Raverta, Sathwik Yarlagadda and Minh Nguyen

Abstract

This survey paper investigates the application of Graph Machine Learning (GML) in credit card fraud detection, a pressing issue in financial security. By reviewing literature, including a foundational overview [1] and specific GML solutions [2, 3, 4], this work categorizes the diverse methodologies employed. It leverages the taxonomy from [1], distinguishing techniques by graph type (plain, attributed, dynamic, heterogeneous) and anomaly detection focus (node, edge, subgraph, graph).

1 Introduction

Anomaly Detection aims to proactively recognize those observations which deviate significantly from the others in the sample [1]. The problem was first introduced in the statistics community on the 19th century. However, its popularity grow significantly as people began to use the Internet to make purchases, socialize and obtain information. Nowadays, detecting anomalies is a crucial task to prevent undesirables outcomes such as misinformation, financial frauds, network intrusions among others.

A data-oriented definition for the anomaly detection problem would be to define it as "the data mining process that aims to identify the unusual patters that deviate from the majority in a dataset". Traditional machine learning methods, which represent observations as feature vectors, can be used to classify these events as either anomalous or normal. However, this approach is not able to handle the information that comes from the relationship between the observations.

With the promise of being able to exploit the relationship between objects represented as graphs, it was coined the idea of performing Graph Anomaly Detection (GAD). As graph are non-euclidean structures, traditional machine learning technique may fall short to exploit their rich information. As a consequence, researchers have focused their attention on developing specific techniques to handle anomaly detection over graphs directly.

The survey is structured as follows: Section 1 outlines the GAD method taxonomy from [1]. Section 2 discusses credit card fraud detection, key challenges, and three graph machine learning solutions. The conclusion synthesizes the literature findings.

1.1 AGD techniques taxonomy

According to [1], GAD techniques can be classified according the model of graph they use and the ML task they aim to solve. Section 1.1.1 describe different graph model and section1.1.2 enumerates the ML task that are relevant for AGD.

1.1.1 Graph models for AGD

There are different type of graphs that can be used in AGD. [1] classifies them into 3 kind of structures:

- **Plain Graphs**: A static plain graph $G = (V, E)$ comprises a node set $V = \{v_i\}_{i=1}^{n}$ and an edge set $E = \{e_{i,j}\}$ where n is the number of nodes and $e_{i,j} = (v_i, v_j)$ denotes an edge between nodes v_i and v_j. The adjacency matrix $A = [a_{i,j}]_{n\times n}$ is a boolean matrix defined as usual.
- **Attribute Graphs**: A static attributed graph $G = (V, E, X)$ is a plain graph with an additional attribute matrix $X = [x_i]_{n\times k}$ consists of nodes' attribute vectors
- **Dynamic Graphs**: A dynamic graph $G(t) = \{V(t), E(t), X_v(t), X_e(t)\}$ comprises nodes and edges changing over time.

Other kinds of graph structures such as heterogeneous graphs or multigraph can be also suitable for AGD. Particularly, many solutions leverages heterogeneous graphs. A **heterogeneous graph** is a graph composed of different types of nodes and edges, representing diverse relationships and entities.

1.1.2 AGD tasks

An AGD machine learning task refers to a specific problem that can be addressed and solved using machine learning techniques and has a direct mapping to the anomaly detection problem. As it was introduced in [1], GAD can be defined as the task of identifying either anomalous nodes, edges, sub-graphs or graphs.

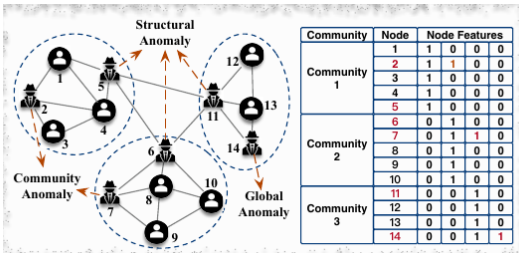
Anomalous node detection: This task focus on detecting those graph nodes which deviate significantly from the overall patter of the sample. This anomaly can be found in 3 different dimensions: global, structural and inside a

community. Figure 1 extracted from [1] is useful to understand the 3 possibilities. In this example, node 14 is considered a *global anomaly* because it is the only node having 1 in the 4th node feature. Nodes 5, 6 and 11 constitutes a *structural anomaly* as they are the only nodes to have links with other communities than the one they belong to. Lastly, nodes 2 and 7 are considered *community anomalies* as their features differentiate from the other inside their communities.

Anomalous edge detection: Sometimes it is convenient to focus in edge anomalies as they represent unusual relationships between real objects. For example, we could identify those edges that connects fraudsters and benign users or hacked websites with credit cards that made a purchased there and their data may be compromised.

Anomalous sub-graph detection: Some other patterns can be better learn by analyzing sub-graphs.

Anomalous graph detection: It is focused on identifying irregular or unusual patterns within a dataset of graphs, distinguishing these from typical graph structures.



Community	Node	Node Features			
Community 1	1	1	0	0	0
	2	1	1	0	0
	3	1	0	0	0
	4	1	0	0	0
	5	1	0	0	0
Community 2	6	0	1	0	0
	7	0	1	1	0
	8	0	1	0	0
	9	0	1	0	0
	10	0	1	0	0
Community 3	11	0	0	1	0
	12	0	0	1	0
	13	0	0	1	0
	14	0	0	1	1

Figure 1: 3 types of node anomalies: Structural, inside a Community and Global

2 Credit card fraud detection

Credit card fraud detection (CCFD) encompass detecting scenarios where cards are forged, stolen, or otherwise compromised, allowing fraudsters to make purchases under the guise of the legitimate cardholder. As most of the transaction are legitimate made by the card owners, the problem of detecting those fraudulent observations can be seen as anomaly detection problem.

However, the CCFD problem has specific challenges that the solution methods should tackle. Some of the most important ones are as follows:

- **Training data availability**: training data for fraud detection is very scarse due to its sensitive nature. Many datasets are either fully synthetic or the data has been altered so to not reveal sensitive information about credit card users.
- **Explainability**: Ideally, when users got a credit card transaction rejected, the bank or financial institution should be able to explain why the transaction got rejected so to be able to change this behavior.
- **Dynamic fraudster behavior**: Fraudsters are not static entities, they learn and adapt their behavior so to be able to bypass security measures. Also, they engage in camouflage to prevent systems from detecting their scams.
- **Class imbalance**: The rarity of fraud records compared to legitimate transactions creates a skewed dataset, complicating the training of models to detect fraud effectively.

These specific challenges highlights the need for developing specific solutions for the CCFD problem. In this survey, 3 approaches for solving CCFD problem using graph machine learning will be analyzed and described from a practical standpoint. The main focus will be to describe how each paper propose to model the transactions events, how the graph for transactions is built, and the specific model they propose to detect frauds.

2.1 Paper 1: ASA-GNN: Adaptive Sampling and Aggregation-Based Graph Neural Network for Transaction Fraud Detection

This paper claims that most of the approaches for CCFD problem fails to address the relationship among transition, among its features and fails to catch the dynamic changes of cardholders' behavior. The solution proposed extend the author previous work to avoid the oversmoothing issue.

Oversmoothing is a common issue in GML, particularly when dealing with GNNs. It refers to a phenomenon where, as the number of layers in a GNN increases, the node features begin to converge towards indistinguishable representations. This happens due to each layer of a GNN aggregates information from a node's neighbors. This issue exacerbate on the context of CCFD problem as fraudsters usually avoid interacting between them. As a consequence, the fraudster specific features becomes to camouflage among those of its neighbors making extremely difficult for the model to effectively differentiate which transactions are fraudulent.

The authors claim to solve this issue by proposing an adaptive sampling strategy and an aggregation method that help the model to learn discriminating representations for the nodes that are then leveraged to detected fraudulent transactions.

2.1.1 Proposed approach:

The proposed approach can be defined as a transductive node labeling approach over a heterogeneous graph. This can be summarized on the following steps:

1. Fix class imbalance on the dataset: Remedy dataset class imbalance by oversampling fraudulent transactions or undersampling legitimate ones to achieve a balanced ratio, such as 1:1 or 1:3. This looks crucial for the proposed solution to work properly.
2. Build a **Transaction Graph** (TG) as defined in definition 1 out of the transaction dataset. In TG, transactions are represented as nodes with attributes.
3. Use a specialized Graph Neural Network to classify the transactions as either fraudulent or not (nodes of the TG) as follows:
   (a) Layer 1 to K: Run ASA-GNN algorithm 1 to generate node embedding
   (b) Layer $K+1$: Run a classifier over the embedding.

Definition 1 (Transition Graph) *A **transaction graph (TG)** is a weighted multi-graph $G = (V, R, P, Weight, E)$ where*

1. *V is the set of $|R|$ nodes and each node v denotes a record $r \in R$ where R is the set of all records;*
2. *P is the set of $m (m \geq 1)$ logic propositions that are assertions with respect to the attributes of transaction records and are rules provided by an expert;*
3. *$Weight : P \to \mathbb{N}$ is a weight function;*
4. *$\mathcal{E} = \bigcup_{i=1}^{m} \{(a,b)_w^{p_i} \mid a \in V \ \wedge \ b \in V \ \wedge \ a \neq b \ \wedge \ p_i(a,b) = True \ \wedge \ w = Weight(p_i)\}$ is the set of all edges.*

Basically, in GNNs connecting 2 nodes means the information of both nodes have high chances of being aggregated. To exploit this characteristic for the CCFD problem, they consider 2 types of aggregation which are important: **device aggregation** and **temporal aggregation**. Device aggregation is aimed to exploit the fact that fraudster are constrained and usually commit fraud on a small number of devices. Meanwhile, temporal aggregation address the fact that fraudsters have a limited window to perform transactions before being noticed by the banks of card's owners. Considering the data contains information as in table 2a, this two ideas were translated into the following propositions:

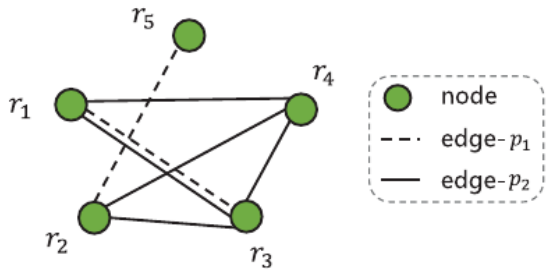
$$p_1(a,b) = \begin{cases} \text{True} & \text{if } a.\text{ip} = b.\text{ip} \wedge \\ & |a.\text{time} - b.\text{time}| \leq 0.5h \\ \text{False} & \text{otherwise} \end{cases} \quad (1)$$

$$p_2(a,b) = \begin{cases} \text{True} & \text{if } a.\text{mac} = b.\text{mac} \wedge \\ & |a.\text{time} - b.\text{time}| \leq 0.5h \\ \text{False} & \text{otherwise} \end{cases} \quad (2)$$

An example of the TG generated from a dataset and considering propositions 1 2 is shown in figure 2a.

ID	Trade Time	Trade IP	Trade MAC	Amount
r1	20200618 09:15	10.168.31.193	15-40-40-50-00-00	800
r2	20200618 09:20	10.168.31.193	11-40-50-50-00-00	300
r3	20200618 09:21	10.168.31.193	15-40-40-50-00-00	200
r4	20200618 09:25	10.168.31.193	12-40-50-50-00-00	150
r5	20200618 09:35	15.168.31.193	11-40-50-50-00-00	650
r6	20200618 10:15	15.168.31.193	12-40-00-50-00-00	930

(a) Transaction records sample



(b) TG for the transition sample table

Figure 2: TG for the transition sample table on the left and considering propositions 1 2

ASA-GNN approach

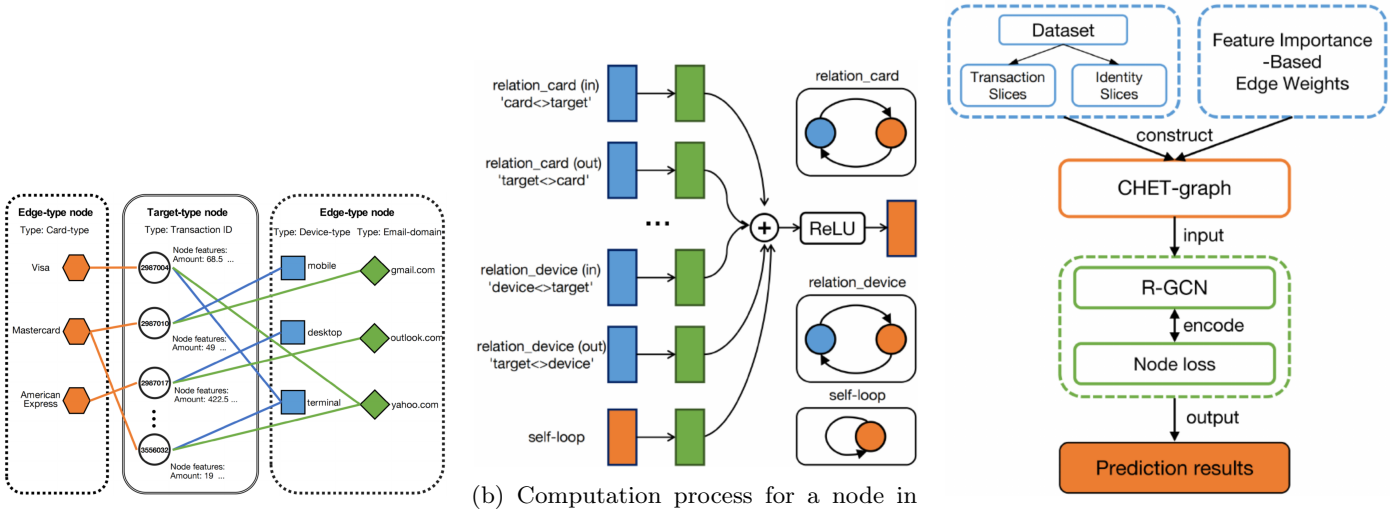
This section describes the behavior of the first k layers of the GNN proposed in this work. The aim of ASA-GNN is to extract node embedding meaningful for the CCDF problem. The full algorithm can be found in the appendix section A. The 3 main components are:

- **Neighbor sampling**: In order to avoid oversmoothing problem, they implement an approach for sampling neighbords based on cosine similarity and edge weight which help to filter noisy neighbors and retain crucial structural information. Nodes that are labeled as fraudulent are oversampled.

- **Neighbor Aggregation**: After the neighbor sampling process, an aggregation function is used to generate the embedding representations of v at each layer.
- **Adaptive Neighborhood Aggregation**: To further avoid oversmoothing, they propose a mechanism that deal with fraudster being camouflaged. Using the entropy measure they aggregate information from neighbors selectively based on their likelihood of being noisy or misleading.

2.2 Paper 2: FIW-GNN: A Heterogeneous Graph-Based Learning Model for Credit Card Fraud Detection [2]

The second paper introduces the Feature Importance-based Weighted Graph Neural Network (FIW-GNN) for CCFD. This method involves creating a heterogeneous graph, the CHET-graph, that effectively uses all dataset and can overcome missing values. It utilizes a feature importance approach, using Chi-squared statistics, to assign edge weights. The integration of the CHET-graph with the relational graph convolutional network (RGCN) and weighted edges forms the FIW-GNN model, designed to improve anomaly detection in credit card transactions.



(a) An example of the CHET-graph.

(b) Computation process for a node in CHET-graph using RGCN model

(c) FIW-GNN methodology

Figure 3: Illustrations of CHET-graph, RGCN computation, and FIW-GNN methodology

2.2.1 Proposed approach:

1. **CHET-graph Construction:** The CHET-graph construction is a four-step process to model transaction data for fraud detection. Initially, transaction records[B.1] from the dataset are divided into numerical transaction slices[B.2] and categorical identity slices[B.3], applying one-hot encoding to categorical features as needed. Nodes are then derived for each record (target-type nodes) and for categorical features in the identity slices (edge-type nodes). Edges are created to connect target-type nodes with edge-type nodes based on the transaction's identity features. An example of CHET-graph is shown in Figure 3a above.
   Definition 2 (CHET-graph) *Let $V_t = \{v_{t1}, v_{t2}, \ldots, v_{tN}\}$ be the set of target-type nodes and $V_e = \{v_{e1}, v_{e2}, \ldots, v_{eM}\}$ the set of edge-type nodes. Denote $V = V_t \cup V_e$ as the node set. For each of R relations including both canonical and inverse directions, E_r is the set of node pairs and A_r is the corresponding adjacency matrix under the relation r where $1 \leq r \leq R$. Denote $E = E_1 \cup \ldots \cup E_R$ and $A = \{A_1, \ldots, A_R\}$, then the heterogeneous graph $G = \{V, E, A\}$ is called a CHET-graph. Specifically, X of size $t_N \times d$ represents the feature matrix over all the target-type nodes V_t, and $W = \{W_1, W_2, \ldots, W_R\}$ to denote the weight matrices for R relations.*

2. **Feature Importance-based Edge Weights Application:** Edge types within the CHET-graph are defined by the attributes present in the identity slice, where the majority of these attributes contain textual information often considered as categorical data. To extract the importance of these categorical features, the authors suggest the use of two statistical method: the Chi-squared[B.4] and Mutual Information[B.5]. These methods assess the significance of the categorical variables in relation to the target variable, thereby assigning more informed weights to the edges and improve the mode's fraud detection capabilities.

3. **Node Feature Transformation with RGCN:** The RGCN[B.6] model stacks layers to update graph nodes based on relational dependencies. Two types of connections will be explored in CHET-graph: interactions between target-type nodes and edge-type nodes, and self-connections of target-type nodes. According to the paper, this setup

ensures that node representations at each subsequent layer are informed by the previous layer. Neighbording nodes's features are processed individually per relation type, combined with a normalization technique, and activiated through ReLU function. For classification, the model minimizes crooss entropy loss and uses softmax on the last layer's output. Finally, we will receive the prediciton results for evaluation in the later stage. Figure 3b shows the a computation example for a node in the CHET-graph utilizing RCGN model.

The proposed FIW-GNN framework in this paper employs a node classification technique of updating the features of nodes (target-type nodes) and utilizing these updated features to classify node. The methodology is applied on the a heterogeneous graph (CHET-graph) which aims to enahance the performance of CCFD.

2.3 Paper 3: Semi-supervised Credit Card Fraud Detection via Attribute-Driven Graph Representation

This paper presents a semi-supervised approach to the CCFD problem by employing a Gated Temporal Attention Network (GTAN) within an attribute-driven graph representation framework. The GTAN model leverages both labeled and unlabeled data to enhance detection capabilities, addressing the limitations of traditional methods that rely heavily on labeled data, which is often scarce and expensive to obtain. The proposed model can be defined as a node labeling method over a temporal graph, which is inherently dynamic as it incorporates temporal relationships among transactions to capture fraud patterns effectively.

2.3.1 Attribute Embedding

The GTAN framework utilizes attribute embedding to transform categorical features, such as card type, channel ID, and merchant type, into tensorial representations. This transformation involves a look-up operation followed by a Multi-Layer Perceptron (MLP) for dimensionality reduction and non-linear transformation. The process enhances the model's ability to interpret complex patterns from transaction data, as illustrated in Figure 4.

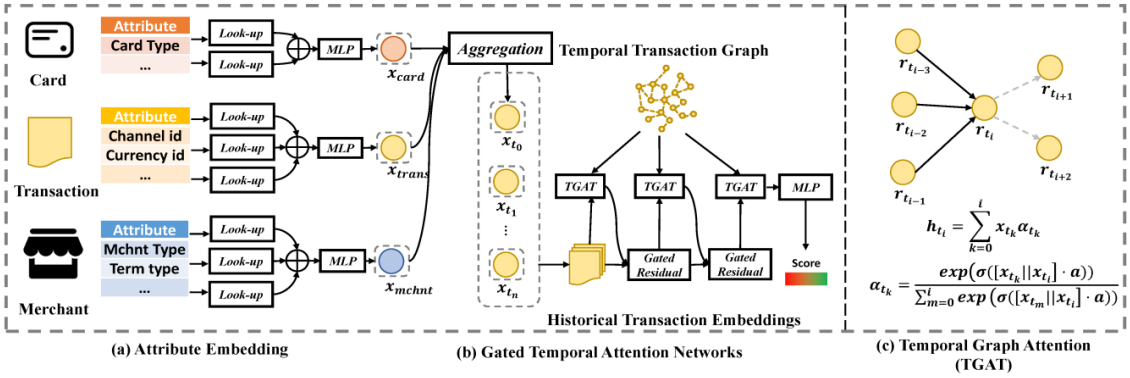


Figure 4: Attribute embedding and GTAN framework.

Each transaction record r_i is encoded through one-hot encoding and subsequently weighted by an embedding matrix E_{attr}, producing attribute-specific embeddings e_{attr}. These are aggregated into a composite embedding $x_{cat}^{(u)}$ for each transaction, where u indexes the transaction.

2.3.2 Gated Temporal Attention Mechanism

The GTAN model's core lies in its Gated Temporal Attention Mechanism, which processes a series of temporal embeddings $X = \{x_{t_0}, x_{t_1}, \ldots, x_{t_n}\}$. This mechanism dynamically assesses the relevance of neighboring transactions via an attention process, which is crucial for detecting fraud patterns over time.

$$\alpha_{t_k} = \frac{\exp\left(\sigma\left(\left[x_{t_i} \| x_{t_k}\right] \cdot \mathbf{a}\right)\right)}{\sum_{m=0}^{\tilde{t}} \exp\left(\sigma\left(\left[x_{t_i} \| x_{t_m}\right] \cdot \mathbf{a}\right)\right)}, \tag{3}$$

Here, σ represents the activation function, $\mathbf{a}$ is the learnable weight vector of the attention mechanism, and $\tilde{t}$ is the extent of the temporal attention span.

2.3.3 Temporal Graph Attention and Risk Propagation

The Temporal Graph Attention (TGAT) component enhances the model's ability to identify fraud patterns manifesting over time by aggregating neighboring node features via an attention-driven mechanism.

$$\alpha_{xt,xi} = \frac{\exp(\text{LeakyReLU}(\mathbf{a}^T[\mathbf{x}_t \parallel \mathbf{x}_i]))}{\sum_{x_j \in N(x_t)} \exp(\text{LeakyReLU}(\mathbf{a}^T[\mathbf{x}_t \parallel \mathbf{x}_j]))}, \tag{4}$$

where $\alpha_{xt,xi}$ indicates the attention coefficient between transactions. This allows the model to focus on transactions that significantly indicate potential fraud within a specific time frame.

2.3.4 Risk Embedding and Propagation

The model introduces risk embedding by incorporating manually annotated labels as categorical attributes within the transaction data. This approach facilitates the use of both labeled and unlabeled data to improve detection performance while addressing potential label leakage issues.

$$x_{ti} = x_{(ti)}^{\text{num}} + x_{(ti)}^{\text{cat}} + \tilde{y}_{(t1)} W_r, \tag{5}$$

Here, W_r are the learnable parameters for risk embedding, integrating risk factors based on historical fraud labels into the transaction graph to enhance predictive accuracy.

2.3.5 Fraud Risk Prediction

The risk-enhanced feature set is then processed through a two-layer MLP to predict fraud risk:

$$\hat{y} = \sigma(\text{PReLU}(HW_0 + b_0)W_1 + b_1), \tag{6}$$

where $\hat{y} \in \mathbb{R}^{N \times 1}$ denotes the fraud risk predictions across all transactions. The objective function is defined using binary cross-entropy:

$$L = -\frac{1}{N} \sum_{i=0}^{N} \left[y_i \log p(\hat{y}_i | X, A) + (1 - y_i) \log(1 - p(\hat{y}_i | X, A)) \right], \tag{7}$$

where y_i is the true label for each transaction. This formulation ensures that the model is trained to accurately predict fraud by learning from both the risk information and the transaction attributes.

2.4 Comparative Analysis and Results

The GTAN model was rigorously tested on three datasets: YelpChi, Amazon, and the collaboratively developed Financial Fraud Semi-supervised Dataset (FFSD). These evaluations were aimed at assessing GTAN's performance against established methods in both supervised and semi-supervised settings. GTAN consistently outperformed baseline models such as GEM, FdGars, and CARE-GNN, showcasing superior AUC, F1 scores, and average precision across all datasets. Notably, it achieved an AUC of 0.9241 on YelpChi and 0.9630 on Amazon, demonstrating robustness and high predictive accuracy. In the FFSD, GTAN's semi-supervised approach significantly enhanced detection capabilities, especially in scenarios with sparse labeled data.

3 Conclusion

In this survey we explored the literature on graph machine learning solution to detect fraudulent credit card transactions. Three recent approaches to the problem were described from a practical standpoint. Two of the solutions modeled transaction information using heterogeneous graphs, while the third utilized attribute graphs. Nonetheless, all three approaches addressed the problem through node labeling tasks.

The survey concludes that while challenges such as data scarcity and the need for explainability persist, GML looks as a promising approach in the fight against financial frauds. Lastly, the survey notes a gap in the discourse regarding real-time transaction processing—a critical aspect of credit card fraud detection. Future explorations must address how these methods can be practically employed by financial institutions for immediate fraud prevention to enhance client protection against fraud.

References

[1] Ma, Xiaoxiao, et al. "A Comprehensive Survey on Graph Anomaly Detection with Deep Learning." *arXiv*, arXiv:2106.07178, 14 June 2021. Available: https://arxiv.org/pdf/2106.07178.pdf.

[2] Yue Tian, et.al. "ASA-GNN: Adaptive Sampling and Aggregation-Based Graph Neural Network for Transaction Fraud Detection". Available: https://ieeexplore.ieee.org/abstract/document/10354439.

[3] Kuan Yan; Junbin Gao; Dmytro Matsypura, et. al. "FIW-GNN: A Heterogeneous Graph-Based Learning Model for Credit Card Fraud Detection". Available: https://ieeexplore.ieee.org/abstract/document/10302538.

[4] Sheng Xiang, et.al. "Semi-supervised Credit Card Fraud Detection via Attribute-Driven Graph Representation". Available: https://www.xiangshengcloud.top/publication/semi-supervised-credit-card-fraud-detection-via-attribute-driven-graph-representation/Sheng-AAAI2023.pdf.

Appendix A Paper 1

Algorithm 1 ASA-GNN Approach

Require: TG $G = (V, R, P, \text{Weight}, \mathcal{E})$, number of layers K, neighborhood sample size $\tilde{z}$, Weight: $\{w_1, \ldots, w_m\}$, non-linear activation function σ.
Ensure: embedding representation h_v^k of each node v

$h_v^0 \leftarrow r_v, \forall v \in V$ ▷ Initialization
for each layer $k = 1, 2, \ldots, K$ **do**
  for each $i = 1, 2, \ldots, \tilde{z}_k$ **do**
    // Neighbor sampling
    $N_v^k \leftarrow$ select neighbors from N_v according to Eq. 8;
    if $c_v = 1$ **then**
      over-sample neighbors according to Eq. 9;
    end if
  end for
end for
for each node $v \in V$ **do**
  // Aggregation
  $\alpha_{v,v'}^k \leftarrow$ Eq. 11;
  $h_{N_v}^k \leftarrow$ Eq. 13;
  $q_v^k \leftarrow$ Eq. 14;
  $h_v^k \leftarrow$;
  $h_v^k \leftarrow \frac{h_v^k}{\|h_v^k\|_2}, \forall v \in V$;
end for

$$P_{v,v'} = \frac{w_{v,v'} \leftrightarrow v, v'}{\sum_{v' \in N_v, v' \neq v} w_{v,v'} \leftrightarrow v, v'} \tag{8}$$

$$N_v^f = \{v' \in V | v' \notin N_v \wedge c_{v'} = 1 \wedge d(v, v') < d_f\} \tag{9}$$

$$h_{N_v}^k = \alpha_{v,v'}^k \cdot A^k \left(h_{v'}^{k-1}\right), \quad \forall v' \in N_v \tag{10}$$

$$\alpha_{v,v'}^k = \frac{\exp(\text{LeakyReLU}(e_{v'}^v))}{\sum_{i \in N_v} \exp(\text{LeakyReLU}(e_i^v))} \tag{11}$$

$$e_{v'}^v = f\left(W^k h_{v'}^k \| W^k h_v^k\right) \tag{12}$$

$$\mathcal{L} = \sum_i [y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i)] \tag{13}$$

$$D(v) = -\sum_{c \in C} P_c(v) \log(P_c(v)) \tag{14}$$

$$P_c(v) = \frac{|\{v' \in N_v | y_{v'} = c\}|}{|N_v|} \tag{15}$$

Appendix B Paper 2

B.1 Transaction Record

A transaction record r is a tuple of attributes in a payment instance, i.e., $r := \{a_1, a_2, \ldots, a_m, b_1, b_2, \ldots, b_n\}$, where a_i denotes the i-th node attribute, which is used as node feature of target-type in the CHET-graph, and b_j denotes the j-th edge attribute which is used to represent edge types and define nodes other than target-types.

B.2 Transaction Slice

Given a transaction record r, its transaction slice r_t is the set of attributes that reflect each transaction instance, containing information such as transaction amount, time interval, match situation, etc.: $r_t := \{a_1, a_2, \ldots, a_m\}$.

B.3 Identity Slice

Given a transaction record r, its identity slice r_i is the set of attributes representing the identity information within the transaction instance, such as card information, address, email domain, device information, etc.: $r_i := \{b_1, b_2, \ldots, b_n\}$. The identity slices reflect the relations between transactions.

B.4 Chi-squared Statistical Method

The Chi-squared statistical method employs the Pearson Chi-squared test to measure the independence of pairs of categorical variables, which are summarized in a contingency table. This method calculates the expected (E_i) and observed (O_i) frequencies for each category to compute the Chi-squared (χ^2) values, as shown in the equation:

$$\chi^2 = \sum \frac{(O_i - E_i)^2}{E_i}$$

where the summation covers all categories within the contingency table. O represents the actual observed count, and E denotes the expected count calculated from the table. These Chi-squared values are then used as the original feature importance scores.

B.5 Mutual Information Statistical Method

The Mutual Information (MI) statistical method assesses information gain and calculates entropy reduction for categorical variables conditioned on the target variable. Specifically, the conditional entropy of discrete variables X given Y is defined as:

$$H[X|Y] = -\sum_x \sum_y p(x,y) \log_2 p(x|y)$$

The sums run over all the categorical values of X and Y. This value represents the expected additional bits needed to encode X given that Y is known. The MI score is then obtained using the equation:

$$MI[X,Y] = H[X] - H[X|Y] = H[Y] - H[Y|X]$$

MI scores vary from 0 to infinity and are normalized to serve as feature importance values, with higher scores indicating greater importance. These values are pivotal for assigning weights to edges in the CHET-graph.

B.6 RGCN

The Relational Graph Convolutional Network (RGCN) processes large-scale relational data and is built upon the Graph Convolutional Network (GCN) framework. RGCN defines a propagation model for calculating the forward-pass update of a node in a relational multi-graph as follows:

$$h_i^{(l+1)} = \sigma\left(\sum_{r \in R} \sum_{j \in N_i^r} \frac{1}{c_{i,r}} W_r^{(l)} h_j^{(l)} + W_0^{(l)} h_i^{(l)}\right)$$

where $h_i^{(l)}$ is the hidden state of node v_i in the l-th layer of RGCN, σ denotes an activation function such as ReLU, N_i^r is the set of neighbor indices of node i under relation r, and $c_{i,r}$ is a problem-specific normalization constant that is set in advance.