



HAL
open science

Privacy-Preserving Behavioral Anomaly Detection in Dynamic Graphs for Card Transactions

Farouk Damoun, Hamida Seba, Radu State

► **To cite this version:**

Farouk Damoun, Hamida Seba, Radu State. Privacy-Preserving Behavioral Anomaly Detection in Dynamic Graphs for Card Transactions. 2024. hal-04707065

HAL Id: hal-04707065




<https://hal.science/hal-04707065v1>

Preprint submitted on 24 Sep 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Privacy-Preserving Behavioral Anomaly Detection in Dynamic Graphs for Card Transactions

Farouk Damoun^{1,2}(✉) , Hamida Seba^{2,3} , and Radu State¹ 

¹ University of Luxembourg, Luxembourg, Luxembourg
farouk.damoun@uni.lu, radu.state@uni.lu

² Université Claude Bernard Lyon 1, Lyon, France
farouk.damoun@univ-lyon1.fr, hamida.seba@univ-lyon1.fr

³ UCBL, CNRS, INSA Lyon, LIRIS, UMR5205, Villeurbanne, France

Abstract. Anomaly detection in financial transactions poses significant privacy challenges. This paper introduces a federated learning (FL) framework for Privacy-Preserving Behavioral Anomaly Detection using Graph Neural Networks (GNNs) on dynamic graphs to model cardholder transactions. We incorporate anonymization-based and noise-based privacy-preserving methods for feature engineering and a domain-specific negative sampling technique to train models without labeled data, making it suitable for real-world applications. Our results, benchmarked on synthetic and real-world datasets, show that deep learning-based outperform clustering-based methods, with F1-scores of 0.91 ± 0.02 and 0.87 ± 0.04 , respectively. Additionally, using the anomaly score as a feature in fraud detection models yields a $1.76\% \pm 0.54\%$ improvement in F1-score, enhancing fraud detection performance while preserving privacy.

Keywords: Federated Learning · Dynamic Graphs · Finance

1 Introduction

Anomaly detection in financial transactions, particularly within the context of credit card transactions, is a critical task in combating fraud. According to the European Central Bank [1], fraud in the Single Euro Payments Area (SEPA) reached 1.33 billion Euros in 2012 alone, marking a 14.8% increase compared to 2011. Furthermore, online fraud accounted for \$3.5 billion in losses in 2012, with an alarming 30% increase from 2010 [2]. Traditional anomaly detection models for fraudulent behavior often rely on attribute value data generated from transactional records, utilizing both supervised and unsupervised learning methods to identify suspicious activities [3, 4].

Recently, graph-based methods have emerged as a powerful approach to identifying suspicious financial activities, by using both data attributes and graph topological information to detect anomalies [5]. These methods have shown their effectiveness in leveraging both graph structure and attribute data to enhance anomaly and fraud detection in various applications. Comprehensive survey studies on anomaly detection [6], including those using graph-based methods [7–9],

and those specifically focused on fraud detection [10, 11], highlight the need to expand data collection to identify rare and unusual patterns or outliers within datasets that differ significantly across institutions.

Therefore, institutions dealing with the same anomaly and fraud detection challenges are increasingly interested in collaborative learning environments, where training occurs across distributed datasets without the need for raw data exchange, thereby safeguarding sensitive financial information [12, 13], such in Federated Learning (FL) systems. Recent advancements in FL have demonstrated its potential to enhance fraud detection across multiple financial entities while maintaining the privacy of both individuals and institutions [14]. However, the use of shared models in federated learning exposes institutions to significant risks, as highlighted in [15]. In the case of tabular data, the nature of attribute values can lead to privacy leaks, especially when downstream tasks are trained to optimize cross-entropy-like objective functions [16–18].

In this work, we propose a novel federated learning framework for Privacy-Preserving Behavioral Anomaly Detection, leveraging Graph Neural Networks (GNNs) to model cardholder transactions as dynamic graphs. We utilize ego-centric graph modeling in [19] as an **anonymization-based privacy mechanism** [20] to eliminate the need for personally identifiable information (PII) in the training data, such as the cardholder’s unique identification number. Additionally, a **noise-based privacy mechanism** [21] is employed to further protect sensitive data attributes. These privacy-preserving techniques are essential in mitigating the risk of Gradient Leakage [17], where gradients derived from sensitive financial data can reveal patterns related to individuals. By introducing noise, our framework ensures that even if the gradients are analyzed, sensitive patterns remain obfuscated, safeguarding the cardholder’s privacy. By utilizing dynamic graphs, this framework enables anomaly detection based on behavioral patterns rather than isolated events, providing a comprehensive understanding of fraudulent activities. By incorporating a **domain-specific negative sampling** technique, our framework effectively trains anomaly detection models without relying on labeled data, making it applicable to real-world scenarios where labeled datasets are unavailable.

Our main contributions are:

- We propose a federated learning framework for privacy-preserving behavioral anomaly detection in credit card transactions, using dynamic graphs to model transaction patterns within a collaborative learning environment.
- We develop privacy-preserving feature engineering, as a noised-based privacy mechanism, by encoding spatial and temporal financial attributes while maintaining privacy and preserving key characteristics.
- We introduce a domain-specific negative sampling strategy to improve fraud detection by generating relevant negative samples using dynamic graphs.
- We demonstrate the framework’s effectiveness on two datasets, showing its ability to enhance detection accuracy while protecting individual privacy.

2 Related Work

Graph-level embedding techniques, such as Graph Kernels [22] and Graph2Vec [23], have become essential for representing entire graphs as vectors, enabling tasks like classification, similarity computation, and anomaly detection [22, 23]. While traditional methods like graph kernels can be computationally intensive, newer approaches like Graph2Vec offer unsupervised learning of graph-level representations. Recent methods, including Graph Neural Networks (GNNs) leverage message passing algorithms to capture comprehensive node representations across the graph [24, 25]. However, these approaches primarily address static graphs. Dynamic graph embedding techniques extend these methods to evolving graphs, adapting to changes in structure over time, which is vital in domains like social networks and financial markets [26]. This study adapts dynamic graph embeddings for detecting fraudulent transactions in financial systems.

Federated Learning (FL). FL enables the collaborative training of models on distributed data [12], which is crucial for handling sensitive financial information. Recent studies have leveraged FL for fraud detection, focusing on supervised learning frameworks with labeled data [14, 27]. These approaches demonstrate FL’s potential to improve fraud detection accuracy without requiring data sharing among institutions. Additionally, FL has shown effectiveness in detecting complex financial crimes across multiple entities, further reinforcing its significance in real-world financial applications [13]. However, existing federated fraud detection methods rely on training with raw data attributes, which exposes individual’s sensitive data to potential breaches in case of gradient leakage. Our work seeks to address this issue by developing a framework that preserves the privacy of individuals, specifically cardholders, and participating institutions, by extending prior research on using GNN models to a privacy-preserving environment, introducing a noise-based privacy mechanism. Furthermore, the negative sampling strategy helps avoid label ambiguity and allows training without relying on labeled data, enhancing privacy and robustness.

3 Methodology

3.1 General Framework

At a high level, it comprises several key components: dynamic graph construction, negative sampling, node and graph-level feature extraction using GNNs, and anomaly detection via a fully connected neural network (FCNN). The framework operates within a FL setup, ensuring that sensitive data remains private while enabling collaborative model training across multiple participants.

Dynamic Graph Construction. We model the transaction data for each cardholder c_i as a dynamic graph $G_t^{(i)}$ at time t , defined as:

$$G_t^{(i)} = (V_t^{(i)}, E_t^{(i)}, X_t^{(i)}, T_t^{(i)}),$$

where the graph nodes $V_t^{(i)} = \{v_j\}$ are associated with $(merchant_id, (lat_j, lon_j))$ attributes transformed to features space denoted as $X_t^{(i)}$, where lat_j and lon_j denotes the location coordinates and $merchant_id$ denotes the merchant unique identifier. And the edges $E_t^{(i)} = \{e_{jk}\}$ represent transactions between consecutive merchants, where we consider valid sequences of consecutive transactions relevant when occurring within a two-hour rolling window, commonly used in [28,29]. An edge $e_{jk} \in E_t^{(i)}$ connects the nodes v_j and v_k , indicating a transaction timestamp from v_k after v_j , denoted as $T_t^{(i)}$. As new transactions occur, the graph dynamically evolves to reflect the ongoing changes in the cardholder’s transaction history, providing accurate behavioral representation, unlike the static graph approach used in [19].

The sequence of dynamic graphs of $c_i \in \mathcal{C}$, where $\mathcal{C} = \{c_i\}_{i=1}^K$ denoted as $G^{(i)} = \{G_{t1}^{(i)}, G_{t2}^{(i)}, \dots, G_{tN}^{(i)}\}$ over time N to reflect the consecutive nature of transactions, where each dynamic graph $G_t^{(i)}$ builds upon a cumulative structure and transactions since the first active transactions, the same for the previous graph $G_{t-1}^{(i)}$. The graph collection $G^{(i)}$ effectively captures the temporal dependencies in the cardholder’s transaction behavior.

Negative Sampling. Inspired from [30], we train the proposed framework to distinguish between normal and anomalous patterns within cardholder dynamic graphs, we generate "noisy" or perturbed graphs $\tilde{G}_t^{(i)}$ that deviate from typical transactional patterns of c_i observed in $G_{t-1}^{(i)}$. The generation process is denoted as:

$$\tilde{G}_t^{(i)} = \mathcal{N}\left(G_{t-1}^{(i)}\right)$$

where $\mathcal{N}(\cdot)$ applies domain-specific perturbations to $G_{t-1}^{(i)}$. Detailed steps are provided in Section 3.3.

Feature Extraction for Node and Graph Representations. At each time step t for a given cardholder c_i , the node features $H_t^{(i,0)}$, denoted for simplicity as $H_t^{(0)} \in \mathbb{R}^{|n_0| \times d}$, consist of feature engineered attributes, and the process is detailed in Section 3.2.

To extract higher-level features, a shared GNN architecture with L layers is initialized and deployed across all participants in the federation. At each layer l , the node embeddings $H_t^{(l)} \in \mathbb{R}^{|n_l| \times d_l}$ are updated through graph convolution operation [31] defined as follows:

$$H_t^{(l+1)} = \sigma\left(A_t^{(l)} H_t^{(l)} W^{(l)}\right),$$

where $W^{(l)} \in \mathbb{R}^{d_l \times d_{l+1}}$ are the trainable weight matrices, $A_t^{(l)} \in \mathbb{R}^{|n_l| \times |n_l|}$ is the adjacency matrix at time t encoding the graph structure and $\sigma(\cdot)$ represents the ReLU activation function.

To derive graph-level representations, a pooling mechanism is applied after each GNN layer, progressively reducing the number of nodes by a factor of $p\%$. The number of nodes at layer $l + 1$ is defined as $n_{l+1} = \lceil |V_t^{(l)}| \times p \rceil$. The

pooling operation simultaneously updates the node feature matrix $H_t^{(l)}$ and the adjacency matrix $A_t^{(l)}$, resulting in a coarsened graph [32], we denoted by:

$$H_t^{(l+1)}, A_t^{(l+1)} = \text{Pooling}_{\lceil p\% \rceil} \left(H_t^{(l)}, A_t^{(l)} \right).$$

This iterative process continues until the final layer, where all nodes are pooled into a single cluster, producing a final embedding vector $q_t^{(i)}$ that represents the entire graph. This graph-level representation encapsulates the transaction behavior of the cardholder c_i at time t .

In certain GNN architectures, the pooling layer is trained [33–37], eliminating the need for a predefined parameter $p\%$. Similarly, for the graph convolution operation, alternative approaches such as message-passing frameworks [38, 39] can be used within the proposed framework.

Discriminative Behavioral Anomaly Detection with FCNN. The final graph embedding $q_{t-1}^{(i)}$ and $q_t^{(i)}$ serves as input to a fully connected neural network (FCNN) that acts as a discriminative model for behavioral anomaly detection, denoted as $f : \mathbb{R}^d \rightarrow [0, 1]$. The FCNN consists of multiple dense layers (MLP) with ReLU activation functions, followed by a softmax layer that outputs the anomaly score. The model is trained end-to-end using a cross-entropy loss function:

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^N [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)],$$

where y_i represents the true label, and \hat{y}_i is the predicted probability for the i -th generated sample, where $f(q_{t-1}^{(i)}, q_t^{(i)}) \rightarrow 1$ and $f(q_{t-1}^{(i)}, \tilde{q}_t^{(i)}) \rightarrow 0$, where $Anomaly\ Score = f(q_{t-1}^{(i)}, q_t^{(i)})$, $i \in \{1, 2, \dots, K\}$.

In our FL setting, each client generates a finite number of samples N and computes gradients based on their local model updates and shares only these gradients with the central server. The server aggregates these gradients to update the global model, ensuring that the learned representations are robust and privacy-preserving.

3.2 Feature Engineering with Privacy Preservation

Inspired by the position encoding architecture in Transformer [40], we feature engineer spatial and temporal attributes in financial data as positional embeddings. To address the data breach issue [16], we propose transforming low-dimensional features with privacy-preserving properties into high-dimensional representations of size d as the embedding size, which are then fed into local models, where a noise-based privacy mechanism is applied, as detailed in Algorithm 1.

Location. Building on the foundation of multi-scale location encoders [41], we adapt this approach to be unbounded, as in a FL environment, the minimum and maximum grid scales could compromise the privacy of individual participants. Given latitude ϕ and longitude λ in degrees, the coordinates are first converted to radians, denoted as ϕ_{rad} and λ_{rad} . To generate the high-dimensional representation, we define the scaling factors \mathbf{s}_ϕ and \mathbf{s}_λ as follows:

$$\mathbf{s}_\phi = \frac{2\pi\phi_{\max}}{C_{\text{earth}}} \cdot \frac{1}{2^k}, \quad \mathbf{s}_\lambda = \frac{2\pi}{2^k}, \quad k \in \{1, 2, \dots, d\}.$$

The latitude \mathbf{l}_ϕ and longitude τ_λ vectors are defined as:

$$\mathbf{l}_\phi = [\sin(\phi_{\text{rad}}) \cdot \mathbf{s}_\phi, \cos(\phi_{\text{rad}}) \cdot \mathbf{s}_\phi]^\top, \quad \mathbf{l}_\lambda = [\sin(\lambda_{\text{rad}}) \cdot \mathbf{s}_\lambda, \cos(\lambda_{\text{rad}}) \cdot \mathbf{s}_\lambda]^\top$$

Contrary to [41], the final location representation is obtained by summing the latitude and longitude vectors; $\mathbf{l}_{\text{location}} = \sigma(\mathbf{l}_\phi + \mathbf{l}_\lambda)$.

Here, σ is a non-linear function. The final vector representation preserves both the directional and distance properties.

Temporal. To minimize the risk of compromising FL while handling temporal features such as using one-hot encoding [16], we transform time features into a dense and continuous space. Given a timestamp, the temporal data is embedded in a high-dimensional vector space to capture cyclical patterns at various granularities decomposed into hours, minutes, seconds, days, and months.

To generate the high-dimensional representation, we define the scaling factor \mathbf{s}_k for each dimension k as:

$$\mathbf{s}_k = \frac{1}{k} \sin\left(\frac{2\pi k}{d}\right), \quad k \in \{1, 2, \dots, \frac{d}{2}\}$$

For each temporal component $\lambda \in \{\text{hour, minute, second, day, month}\}$, with periodicity N_λ , the vector representation is defined as:

$$\tau_\lambda = \left[\sin\left(2\pi \frac{\lambda}{N_\lambda}\right) \cdot \mathbf{s}_k, \cos\left(2\pi \frac{\lambda}{N_\lambda}\right) \cdot \mathbf{s}_k \right]^\top$$

The final temporal representation is obtained by summing all granularities vectors; $\tau_{\text{temporal}} = \sigma(\sum_\lambda \tau_\lambda)$, this vectorial representation effectively captures the cyclical nature of time across different temporal scales, enabling robust temporal modeling.

Merchant ID. Given a set of unique merchant IDs V in the entire dataset, each merchant ID is represented as a dense vector of size d , stored in a matrix $\mathbf{M} \in \mathbb{R}^{|V| \times d}$, where d is the dimensionality of the embedding. For a given merchant ID $m \in V$, the corresponding row vector \mathbf{m}_m is selected from \mathbf{M} , making \mathbf{M} a merchant lookup table. In this work, the embedding matrix \mathbf{M} is initialized randomly using normal distributions with standard deviations of 0.1. These embeddings are then fine-tuned during model training rounds to capture merchant-specific patterns in transaction data across the federation.

Privacy-Preserving Feature Engineering. Building upon the generalized differential privacy framework presented in [21], we extended the application of the privacy preservation to encoded financial data attributes—location (L_{location}), temporal (T_{temporal}), and merchant ID (M)—through a personalized differential privacy mechanism coupled with a vectorial shuffling scheme to generate $L'_{\text{location}}, T'_{\text{temporal}}, M'$ as noise-based privacy mechanism for privacy-preserving feature matrices, as detailed in Algorithm 1.

Following [30], we apply a (ϵ_i, δ_i) -Private Shuffle to perturb feature vectors before their use in our federated model, as described in Algorithm 1. The algorithm begins by randomly permuting the elements of a feature vector v using a fixed random seed. Each element is then perturbed with noise introduced by the Gaussian mechanism [42], where the noise level is determined by specific privacy parameters ϵ_i and δ_i assigned to each feature. This personalized perturbation allows for differential control over privacy budgets across features. Subsequently, the algorithm clips the perturbed vector elements to the range $[-1, 1]$ to maintain feature integrity. The final step involves shuffling the vector to further obscure the original feature order, enhancing privacy as proved in [21].

For a given cardholder c_i , the initial node features \mathbf{X}_i are constructed by integrating location information $\mathbf{L}'_i \subset L'$ and merchant ID attributes $\mathbf{M}'_i \subset M'$, expressed as $\mathbf{X}_i = \mathbf{L}'_i + \mathbf{M}'_i$. This element-wise addition captures the combined spatial and merchant characteristics of each node.

Algorithm 1 (ϵ_i, δ_i) -Private Shuffle for Feature Vector

- 1: **Input:** Feature vector $v = (v_1, \dots, v_d) \in [-1, 1]^d$, privacy budget $\epsilon = (\epsilon_1, \dots, \epsilon_d) > 0$, privacy parameter $\delta = (\delta_1, \dots, \delta_d) \geq 0$, random seed s
 - 2: **Output:** Perturbed and shuffled vector $\tilde{x} \in [-1, 1]^d$
 - 3: Set random seed s
 - 4: Random permutation $r : [d] \leftarrow [d]_s$
 - 5: **for** $j \in [d]$ **do**
 - 6: $\tilde{x}_j \leftarrow x_j + \mathcal{N}(0, \frac{\delta_{r(j)}}{\epsilon_{r(j)}})$
 - 7: $\tilde{x}_j \leftarrow \min(\max(\tilde{x}_j, 1), -1)$
 - 8: **end for**
 - 9: $\tilde{x} \stackrel{r}{\leftarrow} \tilde{x}$
 - 10: **return** \tilde{x}
-

Temporal information $\mathbf{T}'_i \subset T'$ is then incorporated into the message-passing process, where ϕ is a simple element-wise addition. The normalized features are updated using:

$$\mathbf{H}_i^{(0)} = \phi \left(\mathbf{X}_i, \bigoplus_{j \in \mathcal{N}(i)} \tanh(\mathbf{X}_j + \mathbf{T}'_{ij}) \right)$$

where $\mathcal{N}(i)$ denotes the set of neighbors for node i . This approach effectively captures interactions by leveraging both spatial and temporal information, enabling the input feature space to incorporate complex patterns in graph-structured data.

3.3 Negative Sampling with Graph Configuration for Model Training

To effectively train the federated model, we introduce a domain-specific negative sampling strategy that leverages graph configuration perturbations based on various types of credit card fraud, illustrated in Figure 1. Rather than generating negative samples from unrelated cardholder graphs, our approach directly

perturbs the graph $G_t^{(i)}$, resulting in contextually relevant and computationally efficient negative local instances.

For each graph, a graph configuration $\mathcal{C} \in \{\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3\}$ is randomly selected from a predefined set, with each configuration designed to simulate a specific type of fraud attack by altering the topology, node features, or both. The configurations are defined as follows:

- Topology Alteration with Stable Node Features (\mathcal{C}_1):

$$\tilde{A}_t^{(i)}, \tilde{E}_t^{(i)} = \mathcal{F}_T(A_t^{(i)}, E_t^{(i)}), \quad \tilde{X}_t^{(i)} = X_t^{(i)}$$

- Node Feature Alteration with Stable Topology (\mathcal{C}_2):

$$\tilde{A}_t^{(i)} = A_t^{(i)}, \quad \tilde{E}_t^{(i)} = E_t^{(i)}, \quad \tilde{X}_t^{(i)} = \mathcal{F}_V(X_t^{(i)})$$

- Topology and Node Feature Alteration (\mathcal{C}_3):

$$\tilde{A}_t^{(i)}, \tilde{E}_t^{(i)} = \mathcal{F}_T(A_t^{(i)}, E_t^{(i)}), \quad \tilde{X}_t^{(i)} = \mathcal{F}_V(X_t^{(i)})$$

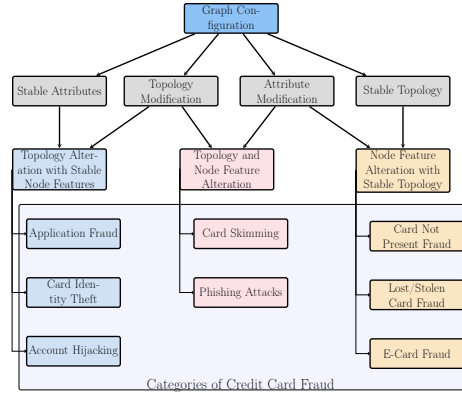


Fig. 1: Categorization of credit card fraud types based on graph configuration manipulations, illustrating how different alterations in topology and node features correspond to specific fraud scenarios [43].

Here, \mathcal{F}_T represents the randomization process applied to the adjacency matrix and edge set, perturbing the graph’s topology. Meanwhile, \mathcal{F}_V involves node injection and swapping, where $\tilde{X}_t^{(i)} \subset X_t$, selectively altering node features within the graph to simulate more complex fraud patterns. These perturbations result in domain-specific negative samples $\tilde{G}_t^{(i)}$ that remain contextually relevant to the cardholder’s transactional behavior.

4 Experimental Settings

4.1 Experimental Design

To evaluate the proposed framework, we use two public financial datasets and simulate a FL environment with multiple clients. Each dataset is divided into

five subsets to create local datasets uniformly distributed by cardholder ID, resulting in varying data sizes due to individual cardholder total transactions. Each local dataset is split by cardholder ID, with 75% for training and 25% for testing. To compare graph pooling methods for cardholder behavior representation under privacy protection, we use the F1 score as the primary metric, as it balances precision and recall, particularly in imbalanced scenarios. Additionally, we report Accuracy, Precision, Recall, and ROC-AUC for a comprehensive performance evaluation. All metrics are averaged across all test sets of the federation participants.

4.2 Parameter Settings

The model parameters were tuned using 5-fold cross-validation to ensure optimal performance. For the GNN model, we configured $L = 5$ layers with a node and edge embedding dimension of $d = 128$, the same dimension used for all data attributes in the vectorial representation. The weight matrices were initialized using Xavier initialization, and ReLU was used as the activation function throughout. We applied a 5% pooling factor at each layer, with a dropout rate of 0.1 to prevent overfitting. The FCNN consisted of 2 dense layers, using ReLU activations with a Softmax function in the final layer. The Adam optimizer, with a learning rate of 0.001, was used for training. In the FL framework, 5 clients were employed, each training locally for 10 epochs, with 200 communication rounds. The FedProx aggregation method [44] is used to merge gradient over all training rounds. For the personalized DP mechanism [45], the privacy budget ϵ was uniformly set between 0.01 and 10, with a constant privacy parameter $\delta = 10^{-4}$, and a fixed shuffling seed was maintained across all clients.

4.3 Datasets

We employ two financial datasets from different regions and economic contexts: one from a Brazilian bank (ELO BR)⁴ and one from a United States bank (IBM US)⁵. The datasets are tabular with the following schema fields: `cardholder_id` (as `INT`), `merchant_id` (as `INT`), `timestamp` (as `Timestamp`), `amount` (as `NUMERIC`), `lat` (as `DOUBLE`), `lon` (as `DOUBLE`), and `MCC` (as `VARCHAR`), which stands for Merchant Category Code, offering detailed insights into credit card transactions. All data types are denoted according to SQL data types.

The original datasets comprise 327,541 merchants from Brazil (ELO BR) and 99,554 merchants from the United States (IBM US), along with 322,862 customers (ELO BR) and 4,967 customers (IBM US). In line with previous studies [28, 29], we pre-processed the datasets by filtering out inactive cardholders with less than 3 transactions per month to concentrate on more significant transaction patterns.

4.4 Baselines

We compare our proposed framework against state-of-the-art baselines, categorized into deep learning-based and clustering-based graph pooling methods. The

⁴ Kaggle | Elo Merchant Category Recommendation: [link](#) .

⁵ Synthetic Transaction Dataset: [link](#) .

deep learning-based methods include **TopKPooling** [33,34], which selects top-K nodes using a learned scoring function to preserve the most informative nodes; **SAGPooling** [34,35], a self-attention based method that assigns attention scores to nodes, focusing on the most relevant parts of the graph; **ASAPooling** [36], which captures local substructures by clustering nodes before pooling to preserve important local information; and **DiffPool** [37], a hierarchical method that creates a coarser graph by learning a differentiable soft assignment of nodes to clusters, effectively capturing global graph properties. For clustering-based graph pooling, methods such as K-Means, Agglomerative Clustering, and Spectral Clustering are used at the node level with k clusters, followed by **Max-Pooling**, which selects the maximum value within each node’s features [46,47], and **Avg-Pooling**, which computes the average node features within each cluster to yield a smooth and generalized graph representation [46,47].

5 Experimental Results

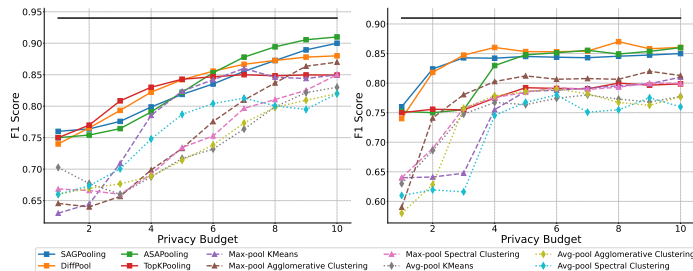


Fig. 2: Impact of privacy budget on F1 score across various graph pooling methods, solid black line refer to the centralized model.

5.1 Privacy Budget and Model Utility

In the synthetic dataset, deep learning methods achieve significant improvements in model performance as the privacy budget increases. At $\epsilon = 2$, the F1 score starts at 0.7 and progressively improves, reaching 0.95 at $\epsilon = 10$, closely approaching the centralized model’s performance. This improvement occurs as the noise applied to the features decreases, allowing for better data representation. A similar trend is observed in the ELO dataset, where F1 scores rise from 0.65 at $\epsilon = 2$ to 0.9 at $\epsilon = 10$. The results highlight that higher privacy budgets lead to enhanced feature utility, particularly since the attribute features are sensitive to any noising mechanisms. Although a privacy budget of 10 is generally considered high, we consider it acceptable in this work due to the sensitivity of the data attributes and the application of privacy mechanisms at the cardholder level. To maximize model performance while maintaining privacy, ϵ is uniformly sampled from a range of 0.01 to 10 for each cardholder, ensuring personalized privacy protection following [45].

In contrast to earlier findings, clustering-based methods show gains with increasing privacy budgets, but even at the highest privacy budget, their improvements are modest compared to deep learning methods. In the synthetic dataset,

Max-Pooling methods stabilize at an F1 score of 0.85, while Avg-Pooling methods reach 0.8 by $\epsilon = 10$. On the ELO dataset, these methods demonstrate similar behavior, but neither fully converges with the centralized model’s performance. These observations suggest that while all models benefit from increased privacy budgets, deep learning methods are more effective in utilizing the additional data fidelity. The consistent results across both datasets demonstrate the robustness of the proposed framework in providing better model utility and privacy trade-offs in a federated learning environment.

Dataset	Pooling Model	Clustering Algorithm	Accuracy	Precision	Recall	F1-Score	ROC-AUC
Synthetic Data	Centralized Model		0.98 \pm 0.02	0.96 \pm 0.02	0.94 \pm 0.03	0.95 \pm 0.02	0.97 \pm 0.02
	Max-pool	KMeans	0.88 \pm 0.04	0.87 \pm 0.04	0.86 \pm 0.05	0.86 \pm 0.04	0.90 \pm 0.04
		Agglomerative Clustering	0.89 \pm 0.04	0.88 \pm 0.04	0.87 \pm 0.05	0.87 \pm 0.04	0.91 \pm 0.04
	Avg-pool	Spectral Clustering	0.88 \pm 0.05	0.85 \pm 0.05	0.85 \pm 0.05	0.85 \pm 0.05	0.89 \pm 0.05
		KMeans	0.86 \pm 0.05	0.84 \pm 0.05	0.83 \pm 0.06	0.83 \pm 0.05	0.88 \pm 0.05
	TopKPooling	Agglomerative Clustering	0.87 \pm 0.05	0.85 \pm 0.05	0.84 \pm 0.06	0.84 \pm 0.05	0.89 \pm 0.05
		Spectral Clustering	0.86 \pm 0.06	0.83 \pm 0.06	0.82 \pm 0.06	0.82 \pm 0.06	0.87 \pm 0.06
	SAGPooling	—	0.89 \pm 0.03	0.86 \pm 0.03	0.85 \pm 0.04	0.85 \pm 0.03	0.91 \pm 0.03
	ASAPooling	—	0.93 \pm 0.02	0.91 \pm 0.02	0.90 \pm 0.02	0.90 \pm 0.02	0.95 \pm 0.02
	DiffPool	—	0.94\pm0.02	0.92\pm0.02	0.91 \pm 0.02	0.91\pm0.02	0.96\pm0.02
	DiffPool	—	0.91 \pm 0.03	0.89 \pm 0.03	0.92\pm0.03	0.88 \pm 0.03	0.94 \pm 0.03
	Brazilian Bank	Centralized Model		0.94 \pm 0.02	0.91 \pm 0.02	0.93 \pm 0.03	0.92 \pm 0.02
Max-pool		KMeans	0.83 \pm 0.05	0.82 \pm 0.05	0.81 \pm 0.06	0.81 \pm 0.05	0.85 \pm 0.05
		Agglomerative Clustering	0.84 \pm 0.05	0.83 \pm 0.05	0.82 \pm 0.06	0.82 \pm 0.05	0.86 \pm 0.05
Avg-pool		Spectral Clustering	0.83 \pm 0.06	0.80 \pm 0.06	0.79 \pm 0.06	0.80 \pm 0.06	0.84 \pm 0.06
		KMeans	0.81 \pm 0.06	0.79 \pm 0.06	0.78 \pm 0.07	0.78 \pm 0.06	0.82 \pm 0.06
TopKPooling		Agglomerative Clustering	0.82 \pm 0.06	0.80 \pm 0.06	0.79 \pm 0.07	0.79 \pm 0.06	0.83 \pm 0.06
		Spectral Clustering	0.81 \pm 0.07	0.78 \pm 0.07	0.77 \pm 0.07	0.78 \pm 0.07	0.82 \pm 0.07
SAGPooling		—	0.84 \pm 0.04	0.81 \pm 0.04	0.80 \pm 0.05	0.80 \pm 0.04	0.85 \pm 0.04
ASAPooling		—	0.88 \pm 0.03	0.86 \pm 0.03	0.85 \pm 0.04	0.85 \pm 0.03	0.89 \pm 0.03
DiffPool		—	0.89\pm0.03	0.87\pm0.03	0.86 \pm 0.04	0.86 \pm 0.03	0.90\pm0.03
DiffPool		—	0.86 \pm 0.04	0.84 \pm 0.04	0.87\pm0.04	0.87\pm0.04	0.87 \pm 0.04

Table 1: Performance of Different Graph Pooling Methods for Federated Anomaly Detection (Including Centralized Model).

5.2 Anomaly Detection Results

In this subsection, the reported performance of various graph pooling methods is under a high privacy budget $\epsilon = 10$. The results in Table 1 demonstrate that GNN-based pooling methods, such as ASAPooling and DiffPool, consistently outperformed traditional techniques. ASAPooling, in particular, achieved impressive metrics, with a ROC-AUC of 0.96 ± 0.02 , accuracy of 0.94 ± 0.02 , and F1 score of 0.91 ± 0.02 across all test sets. This highlights the ASAPooling’s capability to maintain data privacy while preserving critical subgraph structures, essential in federated settings. DiffPool also performed strongly, especially in recall, with an average score of 0.92 ± 0.03 , indicating effectiveness in detecting a wide range of anomalous behaviors generated by the negative sampling strategy 3.3, particularly those influenced by temporal patterns in transactional data.

The success of GNN-based methods can be largely attributed to their use of message passing and graph convolution for feature extraction, which is crucial for the performance of anomaly detection models.

Clustering-based methods and traditional graph pooling techniques, such as Max-pool and Avg-pool, showed limited performance gains, even with an increased privacy budget. The minimal improvements suggest inherent limitations in these methods ability to generalize across diverse and distributed datasets, particularly since clustering algorithms were not federated and operated independently and both the Max and Avg pooling are static graph pooling layers.

Max-pool achieved an average F1 score of 0.85, while Avg-pool reached only 0.80, indicating that both approaches struggle with the complexity of federated graph data. These findings point to the need for more advanced models, like federated clustering algorithms, which could better manage privacy constraints and improve feature extraction at the FCNN level.

5.3 Application to Fraud Detection

Dataset	Client	Local	Local+AS	FL	FL+AS	Diff
		F1	$\Delta\%$ in F1	$\Delta\%$ in F1	$\Delta\%$ in F1	$\Delta\%$ in F1 ($\Delta\%$)
Synthetic Data	1	0.81	+2.0%	+3.6%	+5.1%	+1.5%
	2	0.83	+2.3%	+3.0%	+4.6%	+1.6%
	3	0.82	+2.1%	+3.3%	+5.5%	+2.2%
	4	0.80	+2.5%	+3.9%	+6.1%	+2.2%
	5	0.82	+2.4%	+3.8%	+5.6%	+1.8%
Brazilian Bank	1	0.73	+1.9%	+5.0%	+7.8%	+2.8%
	2	0.71	+2.1%	+5.5%	+7.4%	+1.9%
	3	0.74	+1.6%	+6.3%	+7.0%	+0.7%
	4	0.70	+2.0%	+5.7%	+7.2%	+1.5%
	5	0.72	+1.8%	+6.1%	+7.5%	+1.4%

Table 2: Impact of Federated Anomaly Score (AS) Integration on Federated Fraud Detection Performance, Measured by F1 Score Improvements.

We evaluated the integration of the Federated Anomaly Score (AS) derived from our proposed anomaly detection framework, which is backed by DiffPool GNN-based methods, into a federated fraud detection model. The results, as shown in Table 2, demonstrate a significant improvement in the F1 scores when AS is used as an input feature. For the synthetic dataset, incorporating AS led to an increase in F1 score of 1.5% to 2.2% between all clients, while the Brazilian Bank dataset showed gains ranging from 0.7% to 2.8%. These enhancements underscore the effectiveness of the AS, particularly due to its ability to capture diverse transactional anomalous patterns through DiffPool. Overall, the integration of the anomaly score within the federated learning framework significantly boosts fraud detection accuracy while preserving privacy. The consistent performance improvements across different datasets and clients highlight the robustness and applicability of the proposed method in real-world financial environments, where maintaining data privacy is crucial.

6 Conclusion

In this paper, we have introduced a comprehensive framework for privacy-preserving behavioral anomaly detection, specifically for credit card transactions. By applying a novel negative sampling approach to dynamic graphs, our federated learning (FL) model effectively captures the evolving nature of financial transactions without compromising cardholder sensitive data. The integration of advanced privacy-preserving feature engineering and domain-specific negative sampling has resulted in a significant improvement in fraud detection accuracy, exceeding traditional methods by over 1.76% in F1 score while maintaining individual

privacy. Future work will explore the scalability of the proposed method by testing the time complexity and communication overhead in FL when scaling the number of clients by involving several financial institutions. Additionally, further optimizations for real-time processing and hardware acceleration will be considered to improve its application in large-scale financial systems.

Acknowledgment

We acknowledge the invaluable knowledge exchange facilitated by the Data Analysis Lab team and our industry partners' experts. Partial funding for this work was provided by the Luxembourg National Research Fund (FNR) under grant number 15829274, supplemented by ANR-20-CE39-0008.

References

1. ECB. Technical report. Technical report, European Central Bank, 2014.
2. S McAlearney and TJXD Breach. Ignore cost lessons and weep. *CIO*, page 565, August 7 2008.
3. EWT Ngai, Y Hu, YH Wong, Y Chen, and X Sun. The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature. *Decision Support Systems*, 50(3):559–569, 2011.
4. Amlan Srivastava, Amlan Kundu, Shamik Sural, and Arun K Majumdar. Credit card fraud detection using hidden markov model. *IEEE Transactions on Dependable and Secure Computing*, 5(1):37–48, 2008.
5. Leman Akoglu, Mary McGlohon, and Christos Faloutsos. Oddball: Spotting anomalies in weighted graphs. In *Proc. Pacific-Asia Conf. Knowl. Discovery Data Mining*, pages 410–421, 2010.
6. Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: a survey. *ACM Computing Surveys*, 41(3):1–58, 2009.
7. Leman Akoglu, Hanghang Tong, and Danai Koutra. Graph based anomaly detection and description: a survey. *Data Mining and Knowledge Discovery*, pages 626–688, 2015.
8. Stephen Ranshous, Sihang Shen, Danai Koutra, and Sarah Harenberg. Anomaly detection in dynamic networks: a survey. *Computational Statistics*, pages 223–247, 2015.
9. Kumar Anand, Jyoti Kumar, and Kumar Anand. Anomaly detection in online social network: a survey. In *ICICCT 2017*, pages 456–459. IEEE, 2017.
10. Somnath Bhattacharyya, Sanjay Jha, K Tharakunnel, and JC Westland. Data mining for credit card fraud: a comparative study. *Decision Support Systems*, 50(3):602–613, 2011.
11. Abdalhamid Abdallah, Mohd Afizi Maarof, and Azween Zainal. Fraud detection system: a survey. *Journal of Network and Computer Applications*, 68:90–113, 2016.
12. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*. PMLR, 2017.
13. Toyotaro Suzumura, Yi Zhou, Ryo Kawahara, Nathalie Baracaldo, and Heiko Ludwig. Federated learning for collaborative financial crimes detection. In *Federated learning: A comprehensive overview of methods and applications*, pages 455–466. Springer, 2022.

14. Wenbo Zheng, Lan Yan, Chao Gou, and Fei-Yue Wang. Federated meta-learning for fraudulent credit card detection. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 4654–4660, 2021.
15. Pengrui Liu, Xiangrui Xu, and Wei Wang. Threats, attacks and defenses to federated learning: issues, taxonomy and perspectives. *Cybersecurity*, 5(1):4, 2022.
16. Mark Vero, Mislav Balunović, Dimitar Iliev Dimitrov, and Martin Vechev. Tableak: Tabular data leakage in federated learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pages 35051–35083. PMLR, 2023.
17. Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. idlg: Improved deep leakage from gradients. *arXiv preprint arXiv:2001.02610*, 2020.
18. Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. *Advances in neural information processing systems*, 32, 2019.
19. Farouk Damoun, Hamida Seba, Jean Hilger, and Radu State. G-hin2vec: Distributed heterogeneous graph representations for cardholder transactions. In *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*, pages 528–535, 2023.
20. Latanya Sweeney. k-anonymity: A model for protecting privacy. *International journal of uncertainty, fuzziness and knowledge-based systems*, 10(05):557–570, 2002.
21. E Chen, Yang Cao, and Yifei Ge. A generalized shuffle framework for privacy amplification: Strengthening privacy guarantees and enhancing utility. In *Proceedings of the AAAI Conference*, volume 38, pages 11267–11275, 2024.
22. Pinar Yanardag and SVN Vishwanathan. Deep graph kernels. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1365–1374. ACM, 2015.
23. Annamalai Narayanan, Mahinthan Chandramohan, Rajasekar Venkatesan, Lihui Chen, Yang Liu, and Shantanu Jaiswal. graph2vec: Learning distributed representations of graphs. In *Proceedings of the 13th International Workshop on Mining and Learning with Graphs*, pages 1–8, 2017.
24. Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2017.
25. Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.
26. Charu C. Aggarwal and Karthik Subbian. Evolutionary network analysis: A survey. *ACM Computing Surveys*, 47(1):10:1–10:36, 2014.
27. Mustafa Abdul Salam, Khaled M Fouad, Doaa L Elbably, and Salah M Elsayed. Federated learning model for credit card fraud detection with data balancing techniques. *Neural Computing and Applications*, 36(11):6231–6256, 2024.
28. Anish Khazane, Jonathan Rider, Max Serpe, Antonia Gogoglou, Keegan Hines, C Bayan Bruss, and Richard Serpe. Deeptrax: Embedding graphs of financial transactions. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 126–133. IEEE, 2019.
29. Riccardo Di Clemente, Miguel Luengo-Oroz, Matias Travizano, Sharon Xu, Babu Vaitla, and Marta C González. Sequences of purchases in credit card data reveal lifestyles in urban populations. *Nature communications*, 9(1):3330, 2018.
30. Wenchao Yu, Wei Cheng, Charu C Aggarwal, Kai Zhang, Haifeng Chen, and Wei Wang. Netwalk: A flexible deep embedding approach for anomaly detection in dynamic networks. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2672–2681, 2018.

31. Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.
32. Chuang Liu, Yibing Zhan, Chang Li, Bo Du, Jia Wu, Wenbin Hu, Tongliang Liu, and Dacheng Tao. Graph pooling for graph neural networks: Progress, challenges, and opportunities. *arXiv preprint arXiv:2204.07321*, 2022.
33. Hongyang Gao and Shuiwang Ji. Graph u-nets. In *international conference on machine learning*, pages 2083–2092. PMLR, 2019.
34. Boris Knyazev, Graham W Taylor, and Mohamed Amer. Understanding attention and generalization in graph neural networks. *Advances in neural information processing systems*, 32, 2019.
35. Junhyun Lee, Inyeop Lee, and Jaewoo Kang. Self-attention graph pooling. In *International conference on machine learning*, pages 3734–3743. pmlr, 2019.
36. Ekagra Ranjan, Soumya Sanyal, and Partha Talukdar. Asap: Adaptive structure aware pooling for learning hierarchical graph representations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 5470–5477, 2020.
37. Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. *Advances in neural information processing systems*, 31, 2018.
38. Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1263–1272, 2017.
39. Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2016.
40. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.
41. Gengchen Mai and et al. Multi-scale representation learning for spatial feature distributions using grid cells. In *International Conference on Learning Representations (ICLR)*, 2020.
42. Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3*, pages 265–284. Springer, 2006.
43. Tej Paul Bhatla, Vikram Prabhu, and Amit Dua. Understanding credit card frauds. *Cards business review*, 1(6):1–15, 2003.
44. Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.
45. Xinyang Li, Xiaoqian Ma, Xiao Wang, Yuxin Ding, and Qian Zhang. Personalized privacy-preserving methods for centralized and federated learning. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1271–1280, 2020.
46. Alexandre Duval and Fragkiskos Malliaros. Higher-order clustering and pooling for graph neural networks. In *Proceedings of the 31st ACM international conference on information & knowledge management*, pages 426–435, 2022.
47. Sung Moon Ko, Sungjun Cho, Dae-Woong Jeong, Sehui Han, Moontae Lee, and Honglak Lee. Grouping matrix based graph pooling with adaptive number of clusters. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8334–8342, 2023.