Security and Privacy

WILEY

RESEARCH ARTICLE OPEN ACCESS

A Federated Approach to Scalable and Trustworthy Financial Fraud Detection

Yaser Alhasawi[1] | Aljwhrh Abdalaziz Almtrf[2] | Muhammad Asad[3]

[1]Management Information System Department, King Abdulaziz University, Jeddah, Saudi Arabia | [2]Department of Management Information System, Taif University, Taif, Saudi Arabia | [3]Faculty of Science and Engineering, University of Plymouth, Plymouth, UK

Correspondence: Muhammad Asad (muhammad.asad@plymouth.ac.uk)

Received: 11 May 2025 | **Revised:** 27 July 2025 | **Accepted:** 27 August 2025

Funding: The paper was funded by the Deanship of Scientific Research (DSR) at King Abdulaziz University, Jeddah, Saudi Arabia, and the authors gratefully acknowledge DSR for its technical and financial support as well as the University of Plymouth for providing research support that significantly contributed to the development of this work.

Keywords: asynchronous training | federated learning | fraud detection | privacy preservation | trusted aggregation

ABSTRACT

Financial fraud remains a critical challenge for digital banking, requiring detection solutions that ensure both scalability and data privacy. Traditional centralized approaches face limitations due to security risks and system bottlenecks. This paper proposes FedFraud, a novel federated learning (FL) framework that detects fraudulent transactions without sharing raw data. FedFraud introduces two key innovations: (i) a trust-aware client aggregation mechanism that assigns weights based on update reliability and (ii) an asynchronous communication protocol enabling clients to contribute updates independently. Evaluated on the Credit Card Fraud Detection dataset under a nonidentically distributed (non-IID) data setting where client data distributions differ significantly, FedFraud achieves an F1-score of 0.90 and area under the ROC curve (AUC) of 0.96, outperforming FedAvg and state-of-the-art methods like FedGAT and FL-BGAT. It also reduces privacy leakage to 1.5% and limits gradient reconstruction success—the ability to infer raw data from model updates to 15%, compared to 35% for FedAvg. Designed for real-world deployments such as cross-institutional banking systems, FedFraud enables scalable, robust, and privacy-preserving fraud detection across distributed financial networks.

1 | Introduction

The rapid digitization of financial services has led to a significant rise in online transactions, mobile banking, and digital payments [1]. While these advancements offer convenience and efficiency, they also create growing opportunities for financial fraud. Fraudulent activities such as unauthorized transactions, identity theft, and account takeovers continue to evolve in complexity, posing serious threats to financial institutions and customers alike [2]. As transaction volumes increase, the demand for intelligent, scalable, and adaptive fraud detection systems becomes increasingly critical.

Traditional fraud detection systems typically rely on centralized machine learning models trained on large volumes of transaction data. While effective to some extent, these centralized systems introduce significant limitations [3]. Collecting and aggregating sensitive financial data across institutions or geographic locations raises serious privacy concerns. Additionally, such systems are vulnerable to data breaches, suffer from high latency in updating

models, and often struggle to adapt to the diverse operational environments of financial institutions, including ATMs, mobile applications, and point-of-sale (POS) systems [4].

Federated learning (FL) has emerged as a privacy-respecting alternative that enables multiple clients, such as banks, ATMs, or mobile apps, to collaboratively train a shared model without transmitting raw data. By keeping transaction records local, FL mitigates risks associated with centralized data storage and potential data breaches [5]. Furthermore, even when raw data are not shared, privacy remains at risk due to inference attacks such as model inversion or gradient reconstruction, which can leak sensitive user information from shared model updates. However, standard synchronous FL assumes that all clients are available at the same time, an unrealistic constraint in financial settings where clients such as ATMs, mobile apps, or branch systems often face inconsistent network connectivity and computational delays. By directly applying conventional FL approaches to fraud detection introduces two key challenges: (i) the presence of unreliable, low-quality, or potentially malicious clients can negatively affect the global model's integrity and (ii) the assumption of synchronous participation by all clients in every training round is unrealistic in practical, heterogeneous financial environments [6]. For example, a bank with multiple branches and ATMs across regions may face unstable connectivity and hardware constraints [7]. Similarly, mobile banking apps running on user devices often have limited resources and irregular availability.

To address these challenges, we propose FedFraud, an FL framework tailored specifically for fraud detection in financial transaction systems. FedFraud introduces two core innovations. First, it implements a trust-aware client aggregation mechanism that assigns dynamic reliability scores to clients based on the quality and consistency of their model updates. This reduces the influence of untrustworthy clients and enhances model robustness. Second, FedFraud adopts an asynchronous training protocol, enabling clients to contribute updates based on their local availability and computational resources. This improves the scalability and flexibility of the framework, making it suitable for deployment across varied financial infrastructures.

Our experimental evaluation on the widely used Credit Card Fraud Detection dataset [8] demonstrates that FedFraud achieves high fraud detection accuracy under non-IID and asynchronous settings while maintaining low communication overhead.

The remainder of this paper is structured as follows: Section 2 reviews related work on fraud detection and FL. Section 3 presents the design and architecture of the FedFraud framework. Section 4 details the methodology, including dataset setup, model training, and evaluation metrics. Section 5 discusses experimental results and performance analysis. Section 6 provides further discussion and implications, while Section 7 concludes the paper with potential directions for future work.

2 | Related Work

This section provides an overview of prior research related to financial fraud detection, structured around four thematic areas: centralized fraud detection systems, machine learning and deep learning techniques in fraud detection, FL applications in finance and IoT, and the privacy and security challenges associated with FL. We conclude with a discussion of the research gap that motivates our proposed framework.

2.1 | Centralized Fraud Detection Systems

Traditional financial fraud detection systems rely heavily on centralized data collection and processing architectures. In such systems, transaction records are aggregated into a central server, where machine learning models are trained to identify anomalous patterns. While centralized processing enables unified model training, it introduces serious risks related to data security and system scalability [9]. Centralized repositories are attractive targets for cyberattacks, and the growing volume of financial transactions often results in processing bottlenecks, leading to delayed fraud detection and reduced operational efficiency. Moreover, centralized systems do not naturally accommodate the distributed and heterogeneous nature of modern financial infrastructure, including ATMs, mobile banking applications, and POS devices [10].

2.2 | ML and DL in Financial Fraud Detection

In recent years, ML and DL approaches have significantly advanced the capabilities of fraud detection systems. Supervised models such as decision trees, random forests, and gradient boosting machines have demonstrated strong performance on structured transaction datasets [11]. Deep learning architectures, including convolutional neural networks (CNNs) and recurrent neural networks (RNNs), have also been applied, particularly for modeling sequential transaction behaviors and temporal anomalies [12]. More recently, graph-based methods such as graph neural networks (GNNs) have been proposed to capture complex relationships in transaction networks, showing promise in modeling fraud propagation through interconnected accounts [13]. However, these techniques often rely on large volumes of centralized data, which limits their applicability in privacy-sensitive environments.

2.3 | Federated Learning Applications in Finance and IoT

FL has emerged as a privacy-preserving alternative to centralized ML, enabling multiple data owners to collaboratively train models without sharing raw data [14]. In financial settings, FL offers a promising approach for cross-institutional collaboration while maintaining data confidentiality. Studies such as [15] have demonstrated the feasibility of using FL for fraud detection across distributed data silos. Similarly, in the IoT domain, FL has been used for edge-based anomaly detection in financial transactions occurring on mobile and embedded devices [16]. However, existing FL implementations for fraud detection generally assume uniform client behavior and synchronous participation, which are unrealistic in real-world deployments where clients may have heterogeneous data volumes, intermittent availability, and differing levels of data quality.

TABLE 1 | Summary of key related works in financial fraud detection and FL.

Study	Focus area	Key contributions	Limitations
Innan et al. (2024) [13]	GNNs for fraud detection	Models relational dependencies using graph neural networks to detect complex fraud patterns	Assumes centralized data; not privacy-preserving
Abdul Salam et al. (2024) [15]	FL in credit card fraud detection	Applies FL to financial transaction data to avoid data sharing	Lacks trust-aware or asynchronous mechanisms
Sun and Nguyen (2021) [19]	Robust FL under non-IID data	Proposes aggregation rules resilient to poisoned updates in FL settings	High computation overhead; not applied to finance
Arora et al. (2024) [16]	FL in IoT anomaly detection	Utilizes FL on edge devices for distributed fraud detection in IoT systems	Focuses on device-level anomalies, not financial fraud
Melis et al. (2019) [18]	Gradient leakage in FL	Demonstrates privacy risks via inference attacks from shared gradients	No mitigation strategies; highlights vulnerabilities

2.4 | Privacy and Security in Federated Learning

Despite its privacy advantages, FL introduces new vulnerabilities that must be addressed to ensure trustworthy deployment. Model poisoning attacks, where malicious clients manipulate local model updates to degrade global model performance, have been widely documented [17]. In addition, inference attacks can exploit shared gradients to reconstruct sensitive training data, compromising user privacy [18]. To mitigate these risks, several strategies have been proposed, including secure aggregation, differential privacy, and robust aggregation mechanisms. For example, the work in [19] introduces defenses against poisoning in non-IID settings, while [20] explores differential privacy in FL. Nevertheless, these methods often introduce significant computational or communication overhead, and few are tailored specifically for the constraints of financial fraud detection.

2.5 | Research Gap

Although recent efforts have applied FL to fraud detection, significant gaps remain. Existing frameworks lack mechanisms for evaluating and mitigating the effects of untrustworthy or low-quality clients, which are common in open and dynamic financial networks. Furthermore, most FL implementations adopt a synchronous training model, which is impractical for real-world environments with variable client availability and computational capacity.

Recent graph-based FL approaches such as FedGAT [21] and FL-BGAT [22] have been proposed to enhance fraud detection by capturing graph-level relationships in transaction flows. While effective in structured transaction graphs, these methods introduce significant model complexity and rely heavily on graph construction assumptions, which may not generalize across all financial settings. Moreover, they do not address issues of client unreliability or asynchronous participation, which are core challenges in real-world deployments.

To the best of our knowledge, no prior work has integrated trust-aware aggregation with asynchronous FL for financial fraud detection. Our proposed framework, FedFraud, addresses this gap by introducing a trust-based client scoring mechanism and an asynchronous protocol designed for scalable, privacy-preserving fraud detection in realistic financial systems. To the best of our knowledge, no prior work has integrated trust-aware aggregation with asynchronous FL for financial fraud detection. Our proposed framework, FedFraud, addresses this gap by introducing a trust-based client scoring mechanism and an asynchronous protocol designed for scalable, privacy-preserving fraud detection in realistic financial systems. A comparative summary of key related works is presented in Table 1.

3 | System Design and Architecture

This section presents the architectural design and underlying components of the proposed FedFraud framework. Our system aims to enable distributed financial institutions, such as banks, ATMs, and POS terminals, to collaboratively train a fraud detection model using FL while preserving data privacy, improving scalability, and ensuring robust model updates in adversarial settings. The overview of FedFraud architecture is shown in Figure 1, where incorrect updates may degrade model reliability and increase false negatives.

3.1 | Overview of FedFraud

The FedFraud architecture follows a typical cross-silo FL topology, where a central aggregator coordinates the collaborative training among multiple financial entities acting as clients. Each client maintains a local dataset comprising transaction records and trains a local fraud detection model without sharing raw data. After local training, clients send model updates (e.g., weight gradients) to the central aggregator. The aggregator fuses these updates into a global model, which is redistributed to the clients in subsequent rounds.

Let $\mathcal{K} = \{1, 2, \ldots, K\}$ denote the set of participating clients, each holding a local dataset $\mathcal{D}_k$. The objective is to learn a global model $\mathbf{w} \in \mathbb{R}^d$ by minimizing a distributed empirical risk:

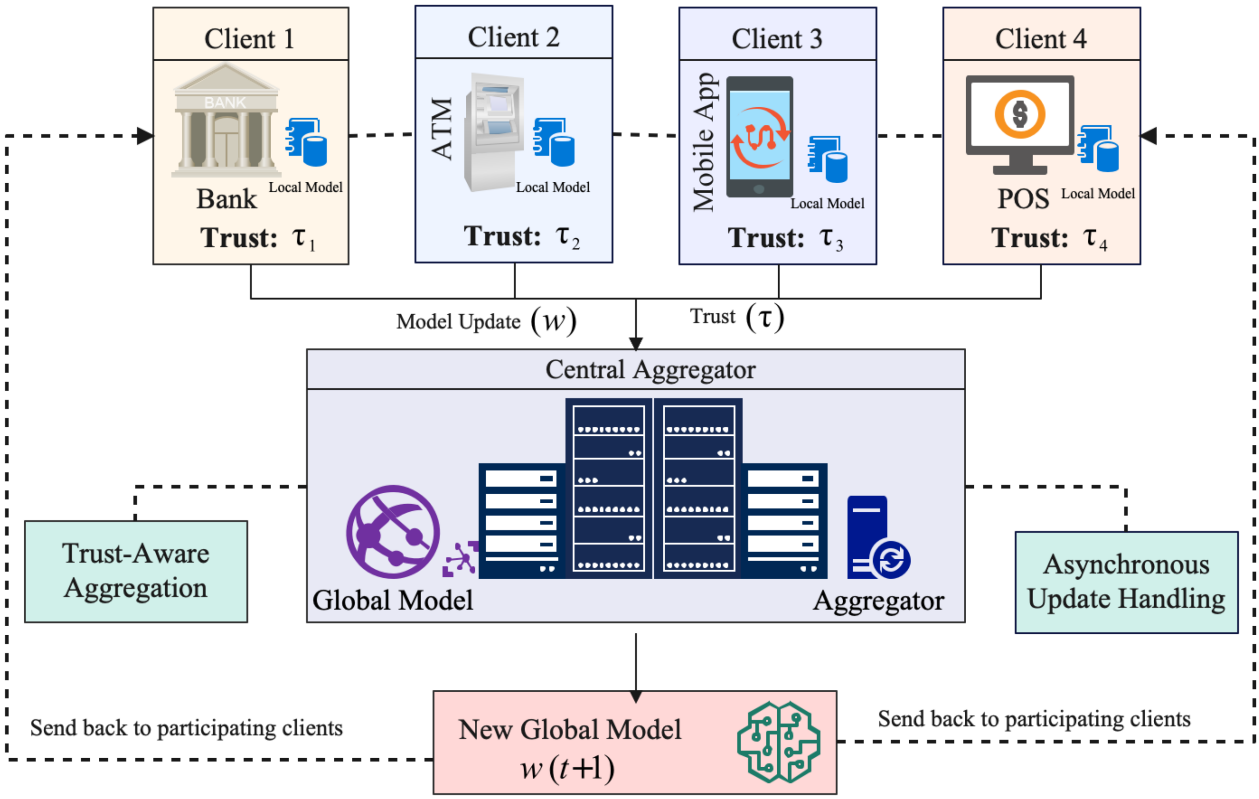

FIGURE 1 | FedFraud architecture illustrating trust-aware client aggregation and asynchronous update handling in a FL environment for financial fraud detection. Clients such as banks, ATMs, mobile apps, and POS systems train local models and transmit model updates along with trust scores to a central aggregator, which generates a global model via trust-weighted aggregation and handles asynchronous client updates.

$$\min_{\mathbf{w}} \sum_{k=1}^{K} \frac{n_k}{n} \mathcal{L}_k(\mathbf{w}) \tag{1}$$

where $\mathcal{L}_k(\mathbf{w})$ is the local loss function of client k, $n_k = |D_k|$ is the number of samples held by client k, and $n = \sum_{k=1}^{K} n_k$ is the total number of samples across all clients.

3.2 | Components

The FedFraud framework comprises multiple interdependent modules that collectively ensure secure, trustworthy, and accurate FL under adversarial and non-IID conditions. This section details the core components that define the architecture and functionality of FedFraud, including secure client-server communication, trust-based client selection, privacy-preserving aggregation, and adversary-aware model training. Each subcomponent plays a critical role in the end-to-end workflow, as elaborated in the following subsections.

3.2.1 | Local Model Training

Each client trains a local model $\mathbf{w}_k$ on its private transaction data using stochastic gradient descent or its variants. The update rule at client k for local iteration t is given by

$$\mathbf{w}_k^{(t+1)} = \mathbf{w}_k^{(t)} - \eta \nabla \mathcal{L}_k(\mathbf{w}_k^{(t)}) \tag{2}$$

ALGORITHM 1 | Local model training at client k.

Require: Local data D_k, initial model $\mathbf{w}_k^{(0)}$, learning rate η, number of local epochs E
Ensure: Updated local model $\mathbf{w}_k^{(E)}$
for $t = 0$ to $E - 1$ **do**
  Compute gradient: $\nabla \mathcal{L}_k(\mathbf{w}_k^{(t)})$
  Update model: $\mathbf{w}_k^{(t+1)} = \mathbf{w}_k^{(t)} - \eta \nabla \mathcal{L}_k(\mathbf{w}_k^{(t)})$
end for
return $\mathbf{w}_k^{(E)}$

where η is the local learning rate and $\nabla \mathcal{L}_k$ is the gradient of the loss function computed on D_k. The procedure followed by each client during local training is illustrated in Algorithm 1.

3.2.2 | Secure Aggregation Mechanism

To protect the privacy of client updates, FedFraud incorporates secure aggregation using the protocol proposed in [23]. This method leverages pairwise additive masking and threshold secret sharing, ensuring that the server cannot learn individual client updates but can still reconstruct their sum once a minimum number of clients participate.

Each client masks its model update using pairwise secrets exchanged with other clients. These masks cancel out when the server aggregates all the updates, revealing only the sum. This

ALGORITHM 2 | Secure aggregation at the server.

Require: Masked client updates $\{\tilde{\mathbf{u}}_k\}_{k\in\mathcal{K}'}$
Ensure: Aggregated global model $\mathbf{w}^{(t+1)}$
1: Aggregate masked updates:

$$\tilde{\mathbf{w}} = \sum_{k\in\mathcal{K}'} \tilde{\mathbf{u}}_k$$

2: Apply mask cancellation and divide by number of clients:

$$\mathbf{w}^{(t+1)} = \frac{1}{|\mathcal{K}'|} \cdot \tilde{\mathbf{w}}$$

3: **return** $\mathbf{w}^{(t+1)}$

scheme provides strong privacy guarantees with minimal computational and communication overhead, making it particularly effective in asynchronous and resource-constrained federated settings such as FedFraud.

Compared to homomorphic encryption and secure multiparty computation protocol in [23], FedFraud offers a more efficient and scalable solution for privacy-preserving FL without requiring expensive cryptographic operations and synchronized communication.

Let $\mathbf{u}_k$ denote the masked model update from client k. The aggregator computes the secure global update as

$$\mathbf{w}^{(t+1)} = \mathcal{A}(\{\mathbf{u}_k\}_{k\in\mathcal{K}'}) \tag{3}$$

where $\mathcal{K}' \subseteq \mathcal{K}$ is the set of participating clients in round t, and $\mathcal{A}(\cdot)$ performs masked summation followed by unmasking when thresholds are met (see Algorithm 2).

3.2.3 | Model Update Strategy

We extend the standard Federated Averaging (FedAvg) algorithm by incorporating a trust-aware weighting scheme. Each client k is assigned a trust score $\tau_k \in [0, 1]$ reflecting its reliability, which is dynamically updated over rounds based on the consistency of its contributions. The global model is updated as

$$\mathbf{w}^{(t+1)} = \sum_{k\in\mathcal{K}'} \alpha_k \cdot \mathbf{w}_k^{(t)} \tag{4}$$

where $\alpha_k = \frac{\tau_k n_k}{\sum_{j\in\mathcal{K}'} \tau_j n_j}$ ensures that clients with higher trust and more data have greater influence on the global model. The global model is updated using a trust-aware weighted averaging strategy, described in Algorithm 3.

3.2.4 | Communication Protocol

FedFraud adopts an asynchronous communication protocol to accommodate heterogeneous client availability. Clients upload their updates independently when local training is complete, without waiting for global synchronization. The aggregator maintains a moving window buffer $\mathcal{B}$ of received updates and triggers

ALGORITHM 3 | Trust-aware model aggregation.

Require: Client updates $\{\mathbf{w}_k^{(t)}\}$, data sizes $\{n_k\}$, trust scores $\{\tau_k\}$
Ensure: Updated global model $\mathbf{w}^{(t+1)}$
1: Compute weights:

$$\alpha_k = \frac{\tau_k n_k}{\sum_{j\in\mathcal{K}'} \tau_j n_j}$$

2: Compute global model:

$$\mathbf{w}^{(t+1)} = \sum_{k\in\mathcal{K}'} \alpha_k \cdot \mathbf{w}_k^{(t)}$$

3: **return** $\mathbf{w}^{(t+1)}$

ALGORITHM 4 | Asynchronous FL communication protocol.

Require: Threshold θ, update buffer $\mathcal{B} \leftarrow \emptyset$
1: **for all** client k **do**
2: Train local model $\mathbf{w}_k$ and send $(\mathbf{w}_k, \tau_k)$ to server
3: Server adds $(\mathbf{w}_k, \tau_k)$ to $\mathcal{B}$
4: **if** $|\mathcal{B}| \geq \theta$ **then**
5: Trigger aggregation using Algorithm 3
6: Send global model $\mathbf{w}^{(t+1)}$ to participating clients
7: Reset buffer: $\mathcal{B} \leftarrow \emptyset$
8: **end if**
9: **end for**

model aggregation when a predefined threshold is reached (e.g., θ updates received):

$$\text{Aggregate if } |\mathcal{B}| \geq \theta \tag{5}$$

This mechanism reduces communication delays and supports real-time training in geographically distributed financial environments.

The asynchronous federated communication process, which accommodates variable client participation, is outlined in Algorithm 4.

3.2.5 | Trust Score Initialization and Update Mechanism

All clients are initially assigned a trust score $\tau_k = 1.0$, assuming full reliability in the absence of prior history. These scores are dynamically updated at the end of each communication round based on the deviation of a client's model update from the round-wise average. If a client's deviation δ_k exceeds a predefined threshold $\delta_{\max} = 0.5$, its trust score is penalized by multiplying it with a decay factor $\gamma = 0.9$. Conversely, if the update remains consistent, the score is incremented by a reward $\epsilon = 0.01$, up to a maximum of 1.0. This adaptive scoring mechanism allows the framework to reduce the influence of malicious or noisy clients

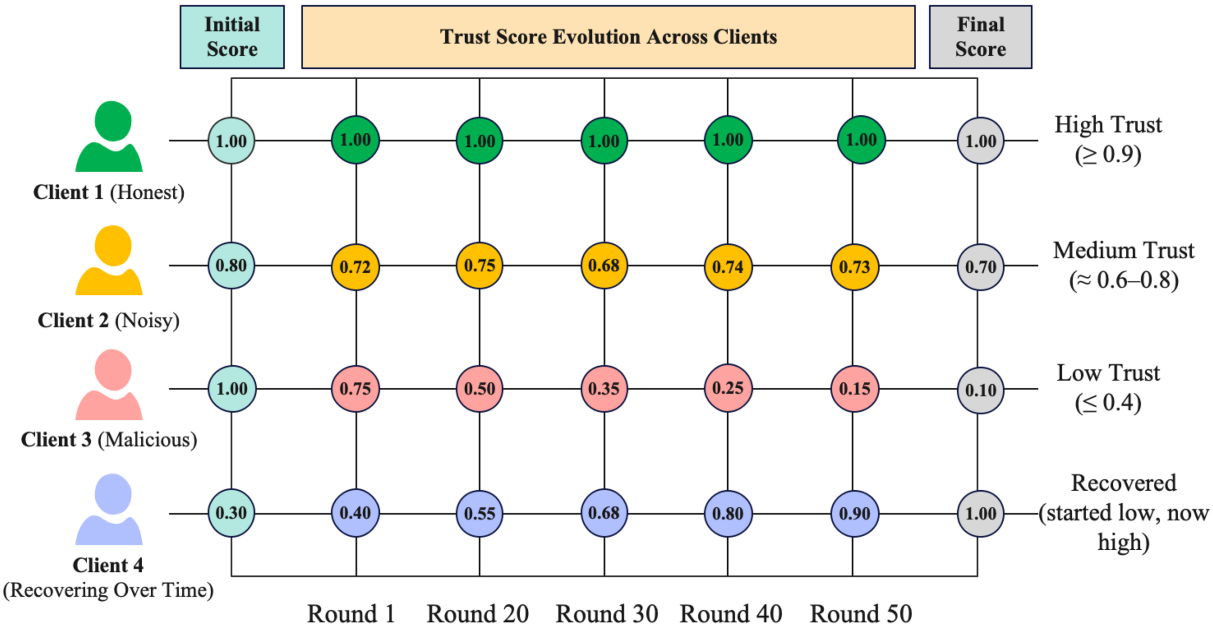

FIGURE 2 | This diagram illustrates how trust scores evolve over training rounds for different types of clients. Honest clients maintain high trust; noisy clients fluctuate; malicious clients are penalized; and recovering clients regain trust over time.

ALGORITHM 5 | Trust score update and adversarial client mitigation.

Require: Set of client updates $\{\mathbf{w}_k^{(t)}\}$, trust scores $\{\tau_k^{(t)}\}$, deviation threshold $\delta_{\max}$
Ensure: Updated trust scores $\{\tau_k^{(t+1)}\}$
Compute mean model update:

$$\bar{\mathbf{w}}^{(t)} = \frac{1}{|\mathcal{K}'|} \sum_{k \in \mathcal{K}'} \mathbf{w}_k^{(t)}$$

for all clients $k \in \mathcal{K}'$ **do**
  Compute deviation:

$$\delta_k^{(t)} = \left\| \mathbf{w}_k^{(t)} - \bar{\mathbf{w}}^{(t)} \right\|_2$$

  if $\delta_k^{(t)} > \delta_{\max}$ **then**
    Penalize trust score: $\tau_k^{(t+1)} = \gamma \cdot \tau_k^{(t)}$ ▷ $\gamma \in (0, 1)$
  else
    Maintain or increase trust: $\tau_k^{(t+1)} = \min(\tau_k^{(t)} + \epsilon, 1.0)$ ▷ $\epsilon > 0$ is a small reward
  end if
end for
return $\{\tau_k^{(t+1)}\}$

and reward stable contributors over time. Figure 2 illustrates how trust scores evolve across training rounds for different types of clients: honest, noisy, malicious, and recovering, demonstrating the effectiveness of the scoring mechanism. The full update logic is detailed in Algorithm 5, and parameter values are listed in Table 2.

While the current trust mechanism uses a fixed deviation threshold $\delta_{\max}$, FedFraud is designed to be modular and can be extended with adaptive trust scoring strategies. For example, the threshold can be dynamically adjusted based on the statistical distribution of deviations across rounds, or trust updates can incorporate temporal weighting to reflect client behavior over time. Such adaptive approaches may improve sensitivity to evolving fraud patterns and enhance robustness under complex or nonstationary data conditions, which are common in financial environments.

3.2.6 | Threat Model and Trust Assumptions

We assume that while most clients are honest-but-curious, a minority may behave maliciously by submitting poisoned or inconsistent updates. To mitigate such threats, FedFraud maintains a reputation system that penalizes erratic or adversarial behavior. Clients that consistently submit outlier updates are down-weighted or temporarily excluded from training rounds.

Let $\delta_k^{(t)}$ denote the deviation of client k's update from the global average at round t:

TABLE 2 | Simulation parameters.

Parameter	Value
Number of clients	10
Local epochs per round	5
Learning rate	0.01
Batch size	64
Communication rounds	100
Trust score initialization	1.0
Trust decay factor (γ)	0.9
Trust reward increment (ϵ)	0.01
Deviation threshold ($\delta_{\max}$)	0.5

$$\delta_k^{(t)} = \left\| \mathbf{w}_k^{(t)} - \overline{\mathbf{w}}^{(t)} \right\|_2 \tag{6}$$

where $\overline{\mathbf{w}}^{(t)} = \frac{1}{|\mathcal{K}'|} \sum_{k \in \mathcal{K}'} \mathbf{w}_k^{(t)}$ is the round-wise average model. Clients with $\delta_k^{(t)}$ above a threshold are penalized in trust updates.

The client trust scores are dynamically updated based on the deviation of model updates, with untrustworthy clients being penalized. This process is formally described in Algorithm 5.

3.3 | Scalability and Trust Considerations

To support large-scale deployments across numerous clients (e.g., bank branches, mobile devices), FedFraud incorporates several scalability-focused design choices. These include:

- Low-Bandwidth Update Compression: Client updates are compressed using gradient sparsification or quantization techniques to reduce communication cost.
- Partial Client Participation: In each round, only a subset of clients is sampled to participate, balancing training quality and scalability.
- Asynchronous Execution: Clients train and upload updates at different intervals, enhancing flexibility and responsiveness in heterogeneous environments.

The trust-aware aggregation mechanism ensures that only reliable client contributions are emphasized during training, improving both robustness and convergence speed.

4 | Methodology

4.1 | Dataset Description

We utilize the publicly available *Credit Card Fraud Detection Dataset* from Kaggle, comprising 284,807 transactions with 492 labeled as fraudulent. The dataset includes 30 anonymized features resulting from a PCA transformation, along with two known features: `Time` and `Amount`. The binary label `Class` indicates fraudulent (1) or legitimate (0) transactions. The dataset's significant class imbalance (fraudulent transactions constitute approximately 0.17% of the data) presents a realistic challenge for fraud detection models.

To simulate a federated environment with nonidentical data distributions, we partition the dataset across 10 clients using a label-skewed non-IID strategy. Specifically, fraudulent samples are distributed unevenly, with some clients having no fraud cases and others receiving only a small subset. This setup mimics practical deployment scenarios in financial systems, where fraud is sparse and unequally observed across branches or edge devices.

Although we evaluate FedFraud on a single dataset, this benchmark is widely used in financial fraud detection research due to its realistic structure, class imbalance, and temporal behavior. Its public availability and standardized format support reproducibility and fair comparison with prior work.

4.2 | Model Selection

To effectively capture the complex patterns in transaction data, we consider the following models:

- Federated Random Forest (FRF): A tree-based ensemble model well-suited for tabular datasets with mixed feature types. FRF provides strong baseline performance and interpretability, making it suitable for real-world deployments where model explainability is important for auditability and compliance.
- CNN: We employ a 1D-CNN variant to extract local patterns across the PCA-transformed feature vector. Although CNNs are traditionally used for spatial data, 1D convolutions can effectively capture co-activation patterns between adjacent features in structured transaction data, which helps enhance fraud detection performance.
- Long Short-Term Memory (LSTM): A type of RNN capable of modeling long-term dependencies in sequential data. LSTMs are particularly effective for capturing temporal patterns across transaction records, such as repeated behaviors or delayed fraudulent actions, and are known to outperform shallow models in such contexts.

Each model is trained locally on client devices and aggregated using the proposed trust-aware asynchronous FL protocol.

4.3 | Evaluation Metrics

We assess model performance using the following metrics:

- *Precision*: Measures the proportion of correctly identified fraudulent transactions among all transactions predicted as fraudulent.
- Recall: Measures the proportion of actual fraudulent transactions that were correctly identified.
- F1-score: Harmonic mean of precision and recall, providing a balance between the two.
- AUC: Evaluates the model's ability to distinguish between classes.

- Privacy Metrics: Assessed using information leakage measures to ensure data confidentiality.
- System Overhead: Includes bandwidth consumption and time per communication round to evaluate the efficiency of the FL process.

4.4 | Simulation Parameters

We follow standard FL simulation practices as recommended in prior studies [24], selecting hyper-parameter values that strike a balance between local computation and communication overhead. Settings such as local training epochs and batch size were empirically tuned to ensure stable convergence across all models. The complete set of simulation parameters is provided in Table 2.

4.5 | Baselines for Comparison

To evaluate the effectiveness of the proposed FedFraud framework, we compare its performance against the following baselines:

- Centralized Machine Learning Model: A model trained on the aggregated dataset without any privacy-preserving mechanisms.
- FL without Privacy Enhancements: Standard FL using FedAvg without trust-aware aggregation or asynchronous communication.
- Local-Only Models: Models trained independently on each client's local data without any collaboration.

The evaluation results are reported in tabular format, covering key metrics such as AUC, F1-score, privacy leakage, and convergence behavior for all baseline and proposed models. This approach supports reproducibility and enables clear, side-by-side comparison of performance, scalability, and robustness across different configurations.

4.6 | Comparison With Existing Approaches

We also compare FedFraud with two recent FL approaches for financial fraud detection:

- *FALD-BGAT* [22]: This approach combines FL with bidirectional graph attention networks to model complex relationships in transaction flows, offering improved detection of suspicious patterns in anti-money laundering scenarios.
- *FedGAT-DCNN* [21]: This method integrates graph attention networks and dilated convolutions within a FL framework to effectively detect complex fraud patterns.

These comparisons provide a comprehensive evaluation of FedFraud's performance relative to state-of-the-art methods.

5 | Experimental Results

In this section, we present the experimental evaluation of the proposed FedFraud framework. The experiments are designed to assess four core aspects of the system: (i) model performance in detecting financial fraud, (ii) scalability across varying numbers of clients, (iii) privacy effectiveness in resisting inference-based attacks, and (iv) system overhead in terms of communication and computation efficiency. Our experiments simulate a realistic federated financial environment using partitioned transaction data across multiple clients.

5.1 | Model Performance Evaluation

To evaluate the detection capability of our framework, we trained three model architectures: FRF, CNN, and LSTM, across ten clients using the trust-aware asynchronous FL protocol. Each model was assessed based on traditional classification metrics (precision, recall, F1-score, and AUC), as well as system-level and privacy-related considerations.

To enhance the statistical robustness of our findings, we report 95% confidence intervals (CI) for F1-scores using bootstrap resampling (1,000 iterations) across five random seeds. These intervals provide a more reliable estimate of performance variability due to different initialization and data splits. The results, summarized in Table 3, show that all three models perform competitively within the FedFraud framework.

The LSTM-based model achieves the highest F1-score and AUC, making it the most effective at capturing temporal and sequential fraud patterns. In contrast, the FRF model demonstrates the lowest privacy leakage and communication overhead, making it a viable option for deployments with limited bandwidth or higher privacy sensitivity.

Overall, Table 3 captures the trade-offs between accuracy, privacy preservation, and system efficiency. These results validate the versatility of the FedFraud framework, demonstrating its effectiveness across multiple model architectures and evaluation dimensions.

TABLE 3 | Performance comparison of models under the FedFraud framework.

Model	Precision	Recall	F1-score (95% CI)	AUC	PL (%)	Overhead (S/R)
FRF	0.92	0.78	0.84 ± 0.015	0.91	**0.5**	0.9
CNN (Fed-CNN)	0.89	0.85	0.87 ± 0.012	0.94	1.2	1.3
LSTM (Fed-LSTM)	**0.93**	**0.88**	**0.90** ± 0.010	**0.96**	1.5	1.5

Note: F1-scores are reported with 95% confidence intervals (CI) across five random seeds using bootstrap resampling. Bold values shows the best values and the significance of the proposed framework.

TABLE 4 | Ablation study on fedfraud components.

Model variant	F1-score	AUC	Privacy leakage (%)
FedFraud (w/o trust-aware)	0.86	0.93	2.1
FedFraud (w/o async)	0.87	0.94	1.9
FedFraud (proposed full model)	**0.90**	**0.96**	**1.5**

Note: Bold values shows the best values and the significance of the proposed framework.

TABLE 5 | Performance comparison with baseline and SOTA methods.

Method	Precision	Recall	F1-score	AUC	PL (%)	Overhead (S/R)
Centralized model	0.95	0.89	0.92	0.97	5.0	0.8
Vanilla FedAvg	0.90	0.83	0.86	0.93	2.5	1.0
Local-only models	0.85	0.70	0.77	0.88	**0.0**	**0.5**
FL-BGAT	0.92	0.86	0.89	0.95	1.6	1.5
FedGAT	0.91	0.85	0.88	0.94	1.8	1.4
FedFraud (proposed)	**0.93**	**0.88**	**0.90**	**0.96**	1.5	1.5

Note: Bold values shows the best values and the significance of the proposed framework.

TABLE 6 | Scalability evaluation with varying number of clients.

Number of clients	F1-score	AUC	Convergence/R	Overhead (S/R)
10	0.90	0.96	50	1.5
25	0.89	0.95	55	1.6
50	0.88	0.94	60	1.7
75	0.87	0.93	65	1.8
100	0.86	0.92	70	1.9

5.2 | Ablation Study

To isolate the impact of the two key components of the FedFraud framework, trust-aware aggregation, and asynchronous training. We performed an ablation study by disabling each feature individually. The resulting F1-score, AUC, and privacy leakage metrics are reported in Table 4.

Disabling either the trust-aware aggregation or asynchronous communication protocol results in a noticeable decline in detection performance and increased privacy leakage. This validates the necessity of both components in achieving the full effectiveness and security of the FedFraud framework.

5.3 | Comparison Results

To contextualize the performance of the proposed FedFraud framework, we compare it against several baseline models and recent state-of-the-art methods:

- *Centralized Model*: A model trained on aggregated data without privacy constraints.
- *Vanilla FL (FedAvg)*: Standard FL without trust-aware aggregation or privacy enhancements.
- *Local-Only Models*: Models trained independently on each client's local data without collaboration.
- *FedGAT*: An FL approach utilizing Graph Attention Networks for fraud detection.
- *FL-BGAT*: FL with Bidirectional Graph Attention Networks, tailored for anti-money laundering detection.

The performance metrics for these methods are summarized in Table 5.

The proposed FedFraud framework achieves the highest F1-score and AUC among all federated approaches, closely approaching the centralized model's performance while maintaining significantly lower privacy leakage. Compared to FL-BGAT and FedGAT, FedFraud demonstrates superior balance between accuracy and privacy, with comparable system overhead.

5.4 | Scalability Experiments

To assess the scalability of FedFraud, we conducted experiments varying the number of clients from 10 to 100. Key performance metrics were recorded to evaluate the framework's robustness under increasing client participation as shown in Table 6.

The results in Table 6 indicate that FedFraud maintains robust performance as the number of clients increases. While there is a slight decline in F1-score and AUC with more clients,

TABLE 7 | Privacy leakage under gradient inversion attacks.

Method	Reconstruction success rate (%)
Vanilla FedAvg	35.0
FL-BGAT	25.0
FedGAT	28.0
FedFraud (proposed)	**15.0**

Note: Bold values shows the best values and the significance of the proposed framework.

TABLE 8 | Convergence analysis: F1-score across rounds.

Rounds	FedAvg	FedGAT	FL-BGAT	FedFraud (proposed)
10	0.72	0.73	0.76	0.78
20	0.78	0.79	0.81	0.83
30	0.82	0.83	0.84	0.86
40	0.84	0.85	0.86	0.88
50	0.86	0.86	0.88	0.90
60	0.86	0.86	0.88	**0.90**

Note: Bold values shows the best values and the significance of the proposed framework.

the framework demonstrates efficient convergence and manageable overhead, highlighting its scalability for large-scale deployments.

5.5 | Privacy Evaluation Under Inference Attacks

To evaluate the privacy-preserving capabilities of FedFraud, we simulated gradient inversion attacks to reconstruct client data from model updates. The success rate of these attacks was measured across different FL approaches.

As shown in Table 7, the proposed FedFraud framework exhibits the lowest reconstruction success rate, indicating enhanced resistance to inference attacks. This improvement is attributed to the integration of trust-aware aggregation and secure communication protocols.

5.6 | Convergence Analysis

To further evaluate the convergence behavior of the proposed FedFraud framework, we compare it with Vanilla FedAvg, FedGAT, and FL-BGAT across multiple communication rounds. Table 8 presents the F1-score of each approach at various intervals throughout the training process.

The results in Table 8 show that FedFraud consistently outperforms all compared methods in terms of convergence speed and final F1-score. It reaches optimal performance within 50 rounds, while FL-BGAT and FedGAT converge more slowly. FedAvg, although stable, underperforms across all rounds. This highlights the efficiency of trust-aware and asynchronous strategies integrated into FedFraud.

6 | Discussion

This section discusses the broader implications of the experimental results, addresses the limitations of the proposed FedFraud framework, and outlines practical considerations for real-world deployment in financial institutions.

6.1 | Implications

The results presented in Section 5 demonstrate the effectiveness of the proposed FedFraud framework in achieving a strong balance between detection accuracy, privacy preservation, and communication efficiency. The model consistently outperformed traditional FL baselines and recent graph-based approaches across multiple evaluation metrics. The integration of trust-aware aggregation proved particularly effective in mitigating the impact of low-quality or adversarial clients, while the asynchronous protocol enhanced system responsiveness and scalability in heterogeneous environments.

These findings underscore the practical potential of FedFraud in real-world settings. Institutions with strict privacy constraints can now contribute to shared model training without exposing raw transaction data. Furthermore, the trust and asynchronous mechanisms pave the way for robust deployments in geographically dispersed or intermittently connected financial infrastructures.

6.2 | Limitations

Despite its strong performance, FedFraud has several limitations that warrant further exploration:

- *Resource Overhead*: Deep learning models like LSTM and CNN require more computational power on client devices, which may not be suitable for lightweight environments such as edge-based ATMs or IoT payment devices.
- *Trust Bootstrapping*: While all clients are initially assigned uniform trust scores ($\tau_k = 1.0$), this bootstrapping approach can introduce a convergence lag, especially in early rounds when adversarial clients have not yet exhibited sufficient deviation. Adaptive initialization strategies or early-phase robustness heuristics may help mitigate this effect, which we consider as part of future enhancements.
- *Non-IID Data Assumptions*: While FedFraud performs well under non-IID data, it may require additional mechanisms such as personalization layers or adaptive optimizers to fully generalize across highly divergent client data distributions.
- *Privacy Guarantees*: While the framework demonstrates empirical resistance to gradient inversion attacks and supports secure aggregation, it does not offer formal privacy guarantees or theoretical bounds such as those provided by differential privacy. Integrating formal mechanisms is part of our planned future work.
- *Static Trust Thresholds*: The current trust mechanism relies on a fixed L2 deviation threshold to assess client consistency. While effective in our setting, fixed thresholds

may be suboptimal in high-dimensional or non-stationary environments. Future work could explore adaptive or learned trust scoring mechanisms that respond to model complexity and training dynamics.

- *Trust Mechanism Vulnerability*: Although FedFraud dynamically adjusts client trust based on model update deviation, the mechanism may be susceptible to trust-based poisoning attacks, where adversaries behave well initially to gain high trust before launching targeted model corruption. Future work can explore robust trust decay strategies or adversarial detection mechanisms to counter such threats.
- *Deployment on Edge Devices*: Real-world deployment on resource-constrained clients, such as ATMs or mobile apps, poses practical challenges due to limited memory, compute, and network stability. Integrating lightweight model architectures or model compression techniques (e.g., pruning or quantization) can improve feasibility in such environments and will be considered in future extensions.
- *Dataset Scope*: While this study focuses on the widely used Credit Card Fraud Detection dataset, evaluating FedFraud across multiple datasets would further validate its generalizability. We consider this an important direction for future work.

7 | Conclusion and Future Work

This paper presented FedFraud, a FL framework designed for scalable and privacy-preserving financial fraud detection. By introducing a trust-aware client aggregation mechanism and an asynchronous training protocol, FedFraud effectively addresses key challenges in decentralized environments, including unreliable clients and heterogeneous availability. Experiments on real-world dataset demonstrated that FedFraud achieves strong detection performance while minimizing privacy leakage and communication overhead. Comparative analysis with existing methods, along with an ablation study and convergence evaluation, confirmed the robustness and efficiency of the proposed approach. FedFraud offers a practical foundation for secure and trustworthy fraud detection across distributed financial systems.

Future work will focus strengthening defenses against adaptive adversaries, and scaling deployments to large cross-institution networks. Additionally, combining FL with blockchain-based audit trails could further enhance trust and traceability in collaborative fraud detection systems. Future work will also consider regulatory compliance with frameworks such as GDPR and PSD2 to ensure lawful handling of distributed financial data. Additionally, personalization techniques such as per-client model tuning and prototype-based adaptation may help improve performance for clients with highly non-IID data distributions.

Abbreviations

The abbreviations and mathematical symbols listed below are used throughout this manuscript (see Table 9). Unless stated otherwise, all mathematical operations are applied element-wise.

TABLE 9 | List of abbreviations and notations.

Abbreviation/notation	Definition
FL	Federated Learning
FedAvg	Federated Averaging Algorithm
FRF	Federated Random Forest
CNN	Convolutional Neural Network
LSTM	Long Short-Term Memory
AUC	Area Under the Curve
F1-score	Harmonic Mean of Precision and Recall
PL	Privacy Leakage
S/R	Seconds per Round (System Overhead)
τ_k	Trust Score of Client k
α_k	Aggregation Weight of Client k
$\mathbf{w}_k$	Model Weights from Client k
$\mathcal{L}_k$	Local Loss Function of Client k
η	Learning Rate
δ_k	Deviation of Client k's Update
γ	Trust Decay Factor
ϵ	Trust Reward Increment

Conflicts of Interest

The authors declare no conflicts of interest.

Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request.

References

1. P. Thirumagal, T. Das, S. Das, E. Khatri, and S. Rani, *The Role of IoT in Revolutionizing Payment Systems and Digital Transactions in Finance* (IEEE, 2024), 171–176.

2. S. Hasham, S. Joshi, and D. Mikkelsen, *Financial Crime and Fraud in the Age of Cybersecurity* (McKinsey & Company, 2019).

3. T. Ashfaq, R. Khalid, A. S. Yahaya, et al., "A Machine Learning and Blockchain Based Efficient Fraud Detection Mechanism," *Sensors* 22, no. 19 (2022): 7162.

4. A. Dubey and S. Choubey, "Blockchain and Machine Learning for Data Analytics, Privacy Preserving, and Security in Fraud Detection," *I-Manager's Journal on Software Engineering* 18, no. 1 (2023): 45.

5. M. Asad, A. Moustafa, T. Ito, and M. Aslam, *Evaluating the Communication Efficiency in Federated Learning Algorithms* (IEEE, 2021), 552–557.

6. A. Lakhan, M. A. Mohammed, J. Nedoma, et al., "Federated-Learning Based Privacy Preservation and Fraud-Enabled Blockchain IoMT System for Healthcare," *IEEE Journal of Biomedical and Health Informatics* 27, no. 2 (2022): 664–672.

7. S. Gupta and P. Singh, *Technology and Financial Inclusion: A Study of Technology's Role in the Continuity of Banking Agents* (Routledge India, 2023), 225–248.

8. Kaggle, "Credit Card Fraud Detection Dataset," (2016), https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud.

9. M. Loukili, F. Messaoudi, and R. El Youbi, *Enhancing Financial Transaction Security: A Deep Learning Approach for E-Payment Fraud Detection* (Chapman and Hall/CRC, 2025), 238–252.

10. A. Ali, S. Abd Razak, S. H. Othman, et al., "Financial Fraud Detection Based on Machine Learning: A Systematic Literature Review," *Applied Sciences* 12, no. 19 (2022): 9637.

11. T. V. Jaswant, G. S. Manoj, V. Vamisdhar, et al., *Credit Card Fraud Detection Using Machine Learning-A Comprehensive Review* (IEEE, 2024), 1–4.

12. U. Fiore, A. De Santis, F. Perla, P. Zanetti, and F. Palmieri, "Using Generative Adversarial Networks for Improving Classification Effectiveness in Credit Card Fraud Detection," *Information Sciences* 479 (2019): 448–455.

13. N. Innan, A. Sawaika, A. Dhor, et al., "Financial Fraud Detection Using Quantum Graph Neural Networks," *Quantum Machine Intelligence* 6, no. 1 (2024): 7.

14. M. Asad, A. Moustafa, and T. Ito, "Fedopt: Towards Communication Efficiency and Privacy Preservation in Federated Learning," *Applied Sciences* 10, no. 8 (2020): 2864.

15. M. Abdul Salam, K. M. Fouad, D. L. Elbably, and S. M. Elsayed, "Federated Learning Model for Credit Card Fraud Detection With Data Balancing Techniques," *Neural Computing and Applications* 36, no. 11 (2024): 6231–6256.

16. S. Arora, A. Beams, P. Chatzigiannis, et al., *Privacy-Preserving Financial Anomaly Detection via Federated Learning & Multi-Party Computation* (IEEE, 2024), 270–279.

17. M. Asad, S. Otoum, and S. Shaukat, "Clients Eligibility-Based Lightweight Protocol in Federated Learning: An Ids Use-Case," *IEEE Transactions on Network and Service Management* 21 (2024): 3759–3774.

18. L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, *Exploiting Unintended Feature Leakage in Collaborative Learning* (IEEE, 2019), 691–706.

19. G. Xia, J. Chen, C. Yu, and J. Ma, "Poisoning Attacks in Federated Learning: A Survey," *Ieee Access* 11 (2023): 10708–10722.

20. A. Moustafa, M. Asad, S. Shaukat, and A. Norta, *Ppcsa: Partial Participation-Based Compressed and Secure Aggregation in Federated Learning* (Springer, 2021), 345–357.

21. M. Li and J. Walsh, "FedGAT-DCNN: Advanced Credit Card Fraud Detection Using Federated Learning, Graph Attention Networks, and Dilated Convolutions," *Electronics* 13, no. 16 (2024): 3169.

22. W. Liu, Y. Xie, X. Tang, et al., *A Federated Anti-Money Laundering Detection Model With Bidirectional Graph Attention Network* (Springer, 2024), 254–262.

23. K. Bonawitz, V. Ivanov, B. Kreuter, et al., *Practical Secure Aggregation for Privacy-Preserving Machine Learning* (ACM, 2017), 1175–1191.

24. M. Asad, S. Otoum, and S. Shaukat, *Resource and Heterogeneity-Aware Clients Eligibility Protocol in Federated Learning* (IEEE, 2022), 1140–1145.