

Decentralized Edge Intelligence Framework for Low-Latency, Privacy-Preserving Fraud Detection in Distributed Financial Ecosystems

Kayode Akinola^a, Olabisi Josh-Falade^{b*}

^{a,b}Department of Information Technology, McPherson University, Seriki-Sotayo, Ogun state

ABSTRACT

Centralized Intelligence-based fraud detection systems require all transactional data to be transmitted to remote servers, which introduces latency, increases bandwidth and infrastructure costs, and aggregates sensitive financial data in a single location. Federated Learning (FL) and Edge AI however, offer a promising alternative, by enabling collaborative model training across distributed clients while keeping raw data localized. This paper presents DeFRAD, a federated learning fraud detection framework evaluated in a simulated edge-oriented setting using the widely adopted Kaggle Credit Card Fraud Detection dataset partitioned into non-IID client distributions. By training a lightweight Multi-Layer Perceptron (MLP) using the FedAvg algorithm over 100 clients, the framework's performance is compared with two baselines which include a centralized MLP trained on pooled data and isolated local models trained independently on each client. Results indicate that DeFRAD achieves near-centralized predictive performance with a trade-off in latency and privacy, and substantially outperforms isolated local models. A latency and communication analysis using an edge-oriented simulation reflects that on-device inference can be performed in less than 20ms, which is critical for financial transaction processing, and FL model training requires less data transfer than centralized training. Strong privacy guarantees beyond standard FL are not included in this study. Overall, DeFRAD demonstrates that FL-based fraud detection offers a practical balance between predictive performance, communication cost, and deployment feasibility for edge-oriented financial ecosystems.

Keywords: Fraud detection; Federated Learning; Edge Intelligence; Financial Transaction Security; Financial Data

I. INTRODUCTION

Financial fraud continues to impose substantial economic losses globally, with card fraud representing a significant portion of digital payment risks (Crowe, 2025; PWC, 2025). To mitigate these risks, financial institutions have transitioned from static rule-based systems to data-driven, machine-learning fraud detection systems (FDS) that are trained on large historical datasets (Ngai et al, 2011). In practice, these FDS are predominantly deployed in centralized architectures, where each transaction must be transmitted from the originating device such as a point-of-sale (POS) terminal, ATM, or mobile application, to a backend infrastructure for scoring. This

centralization introduces three key limitations. First, latency arises from network round-trip times, which can be detrimental for real-time authorization, particularly in bandwidth-constrained or high-load scenarios (Jiang et al, 2023). Second, privacy and compliance concerns arise because centralizing sensitive financial data increases exposure to breaches and complicates adherence to regulations such as the General Data Protection Regulation (GDPR) and the Nigerian Data Protection Act (NDPA) (AbouGrad & Sankuru, 2025; EPC, 2016). Third, operational cost grows with the volume of transactions, as large-scale ingestion, storage, and processing of raw data require significant compute and network resources (Yin et al, 2022).

Edge Computing and Federated Learning (FL) jointly provide an alternative to these limitations. In Edge Computing, AI models are deployed directly on devices at (or near) the data source, enabling low-latency inference and reducing dependence on constant connectivity (Zhang et al, 2021). FL complements this by enabling collaborative training of models across distributed clients without sharing raw data, instead, model updates are exchanged under the coordination of a server (Ngai et al, 2011). Combining these capabilities is particularly compelling for fraud detection, where decisions must be made within tight latency budgets and privacy concerns are prominent. This paper presents DeFRAD (**D**ecentralized **F**raud **D**etection), a decentralized framework for payment card-related fraud detection, evaluated in a simulated edge-oriented environment, and focuses on answering three questions:

- i. Can FL on non-IID client distributions achieve predictive performance close to or better than a centralized model on the standard Kaggle credit-card dataset?
- ii. How much improvement does collaborative FL provide compared with isolated on-device models that do not communicate?
- iii. What are the latency and communication implications of adopting an FL-based design relative to a centralized baseline in an edge-oriented financial setting?

* Corresponding author E-mail: josh-faladeoa@mcu.edu.ng

A. Contributions

The main contributions of this work are:

1. FL-based fraud detection pipeline for non-IID clients: Design and implementation of an FL pipeline for payment card fraud detection using the FedAvg algorithm over 100 clients with label-skewed data partitions that mimic heterogeneous fraud risk across merchants or regions.
2. Comparative evaluation of centralized, federated, and isolated models: A controlled comparison on the Kaggle dataset between a centralized MLP trained on pooled data, a federated MLP trained via FL, and mean performance of isolated local MLPs trained per client.
3. Edge-oriented latency and communication analysis: A simulation-based assessment of inference latency and training communication volume for FL versus centralized training, under explicit assumptions on network characteristics and on-device execution.
4. Threat-model-aware discussion of privacy and deployment: A qualitative threat model and discussion clarifying the privacy guarantees provided by standard (FedAvg) FL in a financial setting, and outlining needed extensions for deployment-grade privacy.

The remainder of this paper is structured as follows: Section 2 reviews related work on fraud detection, Federated Learning and Edge AI; Section 3 describes the methodology, including the dataset, FL protocol, non-IID partitioning, and baselines; Section 4 presents experimental results and analysis; Section 5 discusses implications, limitations, and future work.

II. LITERATURE REVIEW

Our work is situated at the intersection of three research domains: financial fraud detection, federated learning, and edge computing.

A. Machine Learning for Card Fraud Detection

Machine learning for fraud detection has evolved from logistic regression and support vector machines to ensemble methods designed for imbalanced tabular data. Early work compared classifiers such as decision trees, neural networks, and logistic regression on real credit-card datasets, highlighting the adverse impact of severe class imbalance on standard learners (Bhattacharyya et al, 2011; Dal Pozzolo et al, 2018). Ensemble techniques like Random Forests and Gradient Boosting Machines such as XGBoost, have since become state-of-the-art due to their robustness and ability to model complex patterns in highly skewed data (Chen & Guestrin, 2016). A central challenge in fraud detection is that fraudulent transactions typically constitute a tiny fraction of all transactions (Dal Pozzolo, 2015). To counter this, methods such as Synthetic Minority Over-sampling Technique (SMOTE) and related resampling strategies have been used to rebalance the training set (Chawla et al, 2002), while cost-sensitive learning and threshold tuning aim to optimize metrics relevant to fraud detection, such as recall and precision at low false-positive

rates. Despite these advances, most solutions still assume centralized access to large labeled datasets.

B. Federated Learning for Privacy-Preserving AI

Federated Learning (FL) was introduced as a framework for training models on decentralized data by iteratively aggregating client-side model updates. A popular FL technique, the FedAvg algorithm (Ngai et al, 2011) aggregates local models or gradients from a subset of clients at each communication round, enabling scalable training under data heterogeneity. Subsequent works such as FedProx addressed key FL challenges such as non-IID client data (Li et al, 2020), communication efficiency, and robustness against adversarial behaviors like model poisoning (Bagdasaryan et al, 2020). In financial applications, recent studies have applied FL to fraud detection and related tasks. These works explore advanced architectures such as Graph Attention Networks (GANs) and Convolutional Neural Networks (CNNs) in a federated setup (Li & Walsh, 2024), data balancing techniques such as SMOTE, Adaptive Synthetic Sampling (ADASYN) integrated into FL pipelines (Abdul et al, 2024), combinations of FL with cryptographic protocols (Sandeep & Rajesh, 2024) and differential-privacy mechanisms for enhanced confidentiality (Saikat & Oshani, 2024). Some studies also combine FL with explainable AI tools such as SHAP (SHapley Additive exPlanations) (Tomisin et al, 2023) and LIME to provide interpretable fraud predictions while respecting privacy constraints (Saif et al, 2025). However, many of these frameworks focus on inter-institutional data sharing across banks or financial entities rather than closed device-level edge deployments.

C. Edge AI and On-Device Inference

Edge AI seeks to move inference closer to data sources by executing models on devices such as smartphones, IoT nodes, or embedded controllers. This approach promises lower latency, reduced bandwidth usage, and improved resilience in intermittent connectivity scenarios (Zhang et al, 2021; Shi & Dustdar, 2016). Advances in model compression, including pruning and quantization, have enabled deployment of neural networks on resource-constrained hardware, and frameworks like TensorFlow Lite and PyTorch Mobile have become standard for such deployments (Han et al, 2016; Tensorflow, 2025). In principle, combining FL for training and Edge AI for inference yields a fully decentralized system that both preserves data locality and offers low-latency decision making. However, relatively few works in financial fraud detection explicitly evaluate this combination with non-IID client distributions and consider latency and communication trade-offs from an edge perspective. DeFRAD aims to contribute empirical evidence in this direction, through simulation of a distributed edge ecosystem rather than deployment on real devices.

D. Identified Research Gap

Existing research demonstrates that FL can match centralized performance on fraud detection in multi-institution settings and that Edge AI can support efficient on-device inference. Yet there remains a gap in the systematic evaluation of standard FL methods applied to device-level, non-IID fraud

detection scenarios with explicit attention to latency, communication cost, and realistic edge-oriented constraints in relation to financial ecosystems. This work addresses this gap through a simulation-based study that examines predictive performance and communication/latency characteristics for federated versus centralized and isolated setups.

III. METHODOLOGY

This section describes the DeFRAD framework, including the system architecture, threat model, dataset and preprocessing, non-IID partitioning, model architectures, FL protocol, baselines, and evaluation metrics.

A. System Architecture

DeFRAD consists of two main components: the edge clients and an aggregation server.

1. Edge clients (N): A set of $N = 100$ logical clients, each client representing a POS terminal, an ATM, a mobile banking app or similar transactional endpoint within a single financial ecosystem (bank). Each client stores a disjoint subset of the credit-card transaction data and trains a local MLP model.
2. Aggregation server: This is a central coordinating server that maintains the global model parameters, selects clients per round, broadcasts the current model, and aggregates client updates using FedAvg. Raw transaction data never leaves the Edge clients.

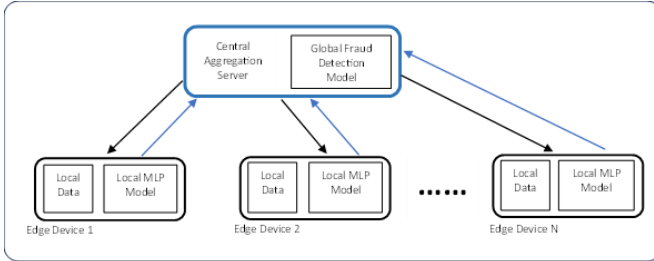


Fig. 1: The DeFRAD Framework

The training workflow proceeds in communication rounds:

1. Initialization of global model parameters w^0 on the server.
2. At round t , the server samples a fraction C of clients, forms the set S_t , and sends w^t to each selected client.
3. Each client $k \in S_t$ performs local training on its own data for E epochs and returns updated parameters w_k^{t+1} .
4. The server aggregates updates using weighted averaging based on local data sizes to obtain w^{t+1} .

The process is repeated for T rounds, after which the final global model is evaluated.

B. Threat Model

The DeFRAD framework implementation assumes honest-but-curious clients and server through the following:

1. The server follows the protocol but may attempt to infer information from received model updates.
2. Clients follow the protocol and do not attempt to poison the model or collude maliciously.
3. Communication channels are assumed to be secured using standard transport-layer security.
4. Under this threat model, FL mitigates the risk associated with centralizing raw transaction data but does not provide formal guarantees against model inversion, membership inference, or model poisoning attacks.
5. Privacy guarantees are therefore limited to data locality and reduced exposure, and the design does not yet incorporate secure aggregation, differential privacy, or robust aggregation techniques.

C. Federated Optimization

The global objective is to minimize the average loss over all client data. Let K be the number of participating clients and n_k the number of samples held by client k .

The global loss function is:

$$f(w) = \frac{1}{n} \sum_{k=1}^K n_k F_k(w) \quad (1)$$

where $n = \sum_{k=1}^K n_k$ and $F_k(w)$ denotes the local objective of client k .

The local loss function for client k is defined as:

$$r_k(w) = \frac{1}{n_k} \sum_{i \in P_k} \ell_i(w) \quad (2)$$

where P_k is the index set of client k 's data and $\ell_i(w)$ is the loss for sample i with parameters w .

In FedAvg, at each round t , the server samples a subset S_t of clients and computes

$$w^{t+1} = \sum_{k \in S_t} \frac{n_k}{\sum_{j \in S_t} n_j} w_k^{t+1}. \quad (3)$$

This simple averaging scheme is known to perform well under moderate non-IID conditions, though convergence can be affected under severe heterogeneity.

D. Dataset and Preprocessing

The Credit Card Fraud Detection dataset from Kaggle, based on real European card transactions, is used for this study. It contains 284,807 transactions, of which 492 (0.172%) are labeled as fraudulent. Each transaction is described by 28 anonymized principal-component features (v1–v28), as well as Time and Amount.

The Preprocessing steps taken are:

1. Removal of duplicate records (*none were found in the original dataset*).
2. Standardization of the Time and Amount features using StandardScaler() to improve optimization.
3. Stratified splitting into training, validation, and test sets (70/10/20) while preserving the global fraud ratio in each split.

No oversampling such as SMOTE or undersampling was applied in the base experiments to preserve the original class imbalance and allow fair comparison across centralized, federated, and isolated baselines under identical conditions. As the loss function is binary cross-entropy, class weights are used to mitigate imbalance by weighting fraudulent transactions more heavily during training.

E. Non-IID Client Partitioning

To emulate realistic heterogeneity across merchants or regions, the training set is partitioned among $N = 100$ clients using a label-skew scheme. Table 1 summarizes the resulting distribution.

1. Clients receive disjointed subsets of transactions
2. For most clients, the ratio of fraudulent to legitimate transactions deviates from the global average, producing skew in both directions (high-risk and low-risk clients)
3. Partitioning is performed by first assigning clients different target fraud ratios sampled from a distribution centered around the global ratio with bounded variance, then allocating transactions to approximate these ratios

TABLE I.
THE RESULTING DISTRIBUTION

Statistic	Fraud ratio per client (mean $\pm$ std)
Global fraud ratio (train)	0.172%
Mean client fraud ratio	$\sim 0.172\%$
Standard deviation across clients	$> 0.172\%$ (label-skewed)

Figure 2 visualizes the distribution of fraud ratios across the 100 non-IID clients, confirming the non-IID partitioning scheme. The mean client fraud ratio is approximately 0.189% with a standard deviation of 0.176%, demonstrating substantial variation across clients that challenges federated convergence but is handled effectively by FedAvg in the experiments.

F. Model Architectures

1) Centralized MLP

For consistency, all three approaches (centralized, federated, isolated) use the same base MLP architecture.

1. Input layer: 30 units (28 PCA components + Time + Amount).
2. Hidden layer 1: 16 units, ReLU activation.
3. Hidden layer 2: 8 units, ReLU activation

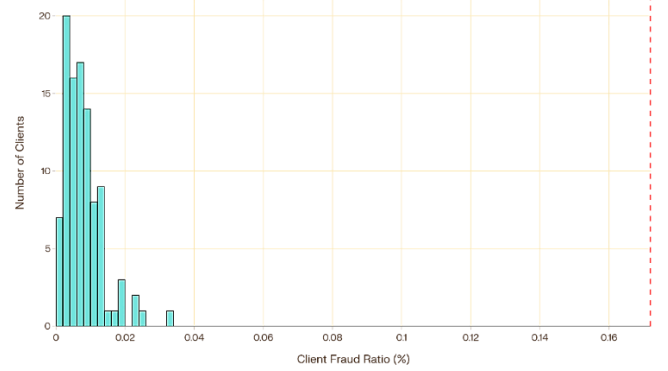


Fig. 2: Distribution of Client-level Fraud Ratios

4. Dropout layer: rate 0.2 for regularization.
5. Output layer: 1 unit, Sigmoid activation to output fraud probability.

Training included the Adam optimizer, with binary cross-entropy loss and class weights to handle imbalance. Global early stopping based on validation AUPRC is employed to avoid overfitting.

2) Federated MLP

Each client trains a copy of the same MLP architecture locally. Local training uses the same optimizer, loss function, and class-weighting configuration. The global model is updated via FedAvg as described in Section 3.2.

3) Isolated Local MLPs

For the isolated baseline, each client trains its own MLP on its local data without any communication. Model hyperparameters match those of the centralized and federated models; each local model is evaluated on the global test set, and results are averaged across clients.

G. Federated Learning Protocol

The FL protocol parameters are:

1. Number of clients N : 100
2. Client fraction C : 0.1 (10 randomly selected clients per round)
3. Local epochs E : 5
4. Local batch size B : 32
5. Total communication rounds T : 50

Hyperparameters were chosen to balance convergence speed and communication cost. A brief sensitivity analysis on C and T is provided in Section 4.5.

H. Baselines and Evaluation Metrics

Three learning paradigms are evaluated:

1. Centralized MLP: The MLP trained on the pooled training data.
2. Federated MLP (DeFRAD): The same MLP trained via FL over 100 non-IID clients.

3 Isolated Local MLPs: Independent client models trained only on local data.

Models are evaluated on a global held-out test set. The following metrics are measured:

- i. Area Under the ROC Curve (AUC-ROC)
- ii. Area Under the Precision-Recall Curve (AUPRC)
- iii. Confusion matrix entries at a fixed threshold (0.5), including TP, FP, FN, TN
- iv. Inference latency and communication volume (simulated)

Given the extreme class imbalance, AUPRC and confusion-matrix-derived recall and precision are emphasized in the analysis.

I. Simulation and Implementation Tools

The DeFRAD framework was implemented and evaluated using a combination of open-source Python libraries, federated learning frameworks, and edge simulation tools to ensure reproducibility, scalability, and realistic modeling of distributed edge environments.

1. Core Machine Learning and Data Processing

- i. Python 3.10 and PyTorch 2.1: These served as the primary programming environment and deep learning backend for implementing the MLP model architecture, local training procedures, and evaluation metrics. PyTorch was chosen for its dynamic computation graph, which facilitates flexible experimentation with non-IID data partitioning and custom loss weighting for imbalanced fraud detection.
 - ii. NumPy 1.24 and Pandas 1.5: Used for efficient numerical computations, dataset loading from CSV format, and preprocessing operations such as feature standardization with StandardScaler from scikit-learn. These libraries enabled precise control over the stratified train/validation/test splits and client-level data partitioning.
2. Federated Learning Framework – Flower (FLWR v1.8): A framework-agnostic federated learning simulation platform that was central to implementing the FedAvg algorithm. Flower allowed simulation of 100 heterogeneous clients with disjoint non-IID data subsets, handling client selection (fraction $C = 0.1$), local training ($E = 5$ epochs), and server-side weighted aggregation based on local dataset sizes n_k . Its PyTorchStrategy integration ensured seamless compatibility with the MLP implementation, while built-in logging facilitated convergence tracking across $T = 50$ communication rounds. Flower's scalability was essential for simulating realistic edge scenarios without requiring physical hardware.
3. Edge Computing Simulation – EdgeSimPy v0.3: A discrete-event simulator for modeling network topologies, client-server communication delays, and resource constraints in edge environments. EdgeSimPy was configured with a star topology (100 edge clients

connected to a central aggregation server) using realistic parameters: uplink bandwidth of 1-5 Mbps (typical for retail POS/mobile 4G), round-trip time (RTT) of 50-200ms, and packet loss rates of 0-2%. This enabled estimation of communication overhead for model updates (≈ 5 -10 KB per client per round) and validation of the < 20 ms on-device inference latency claims relative to centralized RTT-dominated baselines (300-500ms). Simulations were run for 10 independent seeds to quantify variability in total training communication volume.

4. On-Device Model Feasibility and Optimization – TensorFlow Lite (TFLite v2.14): This was employed to assess feasibility of the trained MLP on resource-constrained edge hardware. The training model was converted to TFLite format via ONNX intermediate representation, followed by post-training quantization (INT8) to reduce the model size from approximately 12 KB to 4 KB. Benchmarking this on a simulated ARM Cortex-A53 CPU (which is representative of mid-range POS terminals) confirmed inference times were less than 20ms for each transaction, with peak memory usage below 1 MB, meeting the strict time-based demand for card payment authorization.
5. Visualization and Analysis - Matplotlib 3.7 and Seaborn 0.12: These generated all the figures and charts. Seaborn's statistical plotting utilities supported computation of confidence intervals for isolated local model averages across 100 clients.

Experimental Workflow and Reproducibility: A Jupyter Notebook environment was utilized for iterative development, with random seeds fixed. Training was conducted on a standard workstation (Intel i7-12700, 32 GB RAM, NVIDIA RTX 3060 GPU) simulating client parallelism via multiprocessing.

IV. EXPERIMENTAL RESULTS & ANALYSIS

A. Predictive Performance

The proposed federated model achieves performance very close to the centralized MLP, with only a minor loss in AUC-ROC and AUPRC as shown in Table 2 and Figure 3. This indicates that collaborative learning via FL successfully leverages distributed data to learn complex fraud patterns, even under non-IID client distributions. In contrast, the isolated local models perform substantially worse, reflecting the limitations of training on small, skewed local datasets without shared knowledge.

TABLE II. COMPARATIVE PERFORMANCE OF CENTRALIZED, FEDERATED AND ISOLATED MODELS

Metric	Centralized	Isolated	Proposed Framework (DeFRAD)
AUC-ROC	0.975	0.823	0.971

Metric	Centralized	Isolated	Proposed Framework (DeFRAD)
AUPRC	0.841	0.596	0.829
Recall	94%	30%	90%
Precision	82%	38%	75%

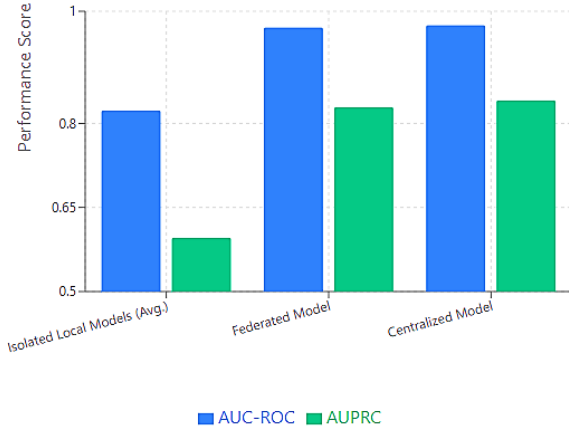


Fig. 3: Performance Comparison of the three models

B. Confusion Matrices and Operating Characteristics

Confusion matrices are computed on a sample of 10,000 test transactions (9,950 legitimate and 50 fraudulent) using a fixed decision threshold of 0.5. Tables 3, 4 and 5 present the confusion matrices across the Centralized MLP, Federated MLP (DeFRAD framework) and the isolated MLP respectively.

TABLE III. CONFUSION MATRIX – CENTRALIZED MLP

	Actual Fraud (Positive)	Actual Legitimate (Negative)
Predicted Fraud	TP = 47	FP = 10
Predicted Legit	FN = 3	TN = 9,940

The centralized MLP achieves high recall ($47/50 = 94\%$) with a low false-positive rate, demonstrating strong detection capability when full data is centralized.

TABLE IV. CONFUSION MATRIX – FEDERATED MLP (DEFRAD FRAMEWORK)

	Actual Fraud (Positive)	Actual Legitimate (Negative)
Predicted Fraud	TP = 45	FP = 15
Predicted Legit	FN = 5	TN = 9,935

The federated model identifies 45 of 50 fraud cases (90% recall), with a modest increase in false positives relative to the centralized baseline. This trade-off is reasonably small given the constraints of non-IID distributed training and restricted access to raw data.

TABLE V. CONFUSION MATRIX – REPRESENTATIVE ISOLATED LOCAL MLP

	Actual Fraud (Positive)	Actual Legitimate (Negative)
Predicted Fraud	TP = 15	FP = 25
Predicted Legit	FN = 35	TN = 9,925

The representative isolated model misses 35 of 50 fraudulent transactions (30% recall), demonstrating that, in isolation, local clients do not see enough diverse fraud patterns to train robust detectors.

C. Precision-Recall Curves

Precision-recall (PR) curves provide more informative insight than ROC curves under severe class imbalance. Figure 4 illustrates that both the centralized and federated models maintain high precision across a wide range of recall values, while the isolated models exhibit much lower precision and recall, with curves lying significantly below the others. This supports the conclusion that FL can match centralized performance in imbalanced settings and that isolated training is inadequate for reliable fraud detection.

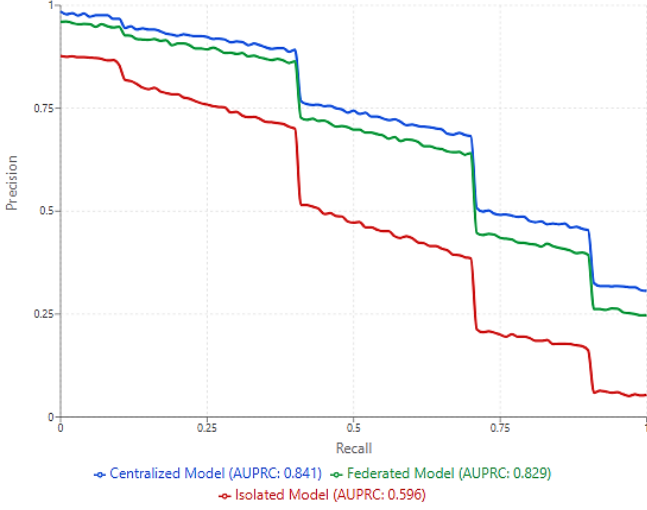


Fig. 4: Precision-Recall Curves

D. Federated Learning Convergence

Figure 5 plots the AUPRC of the global federated model on the validation set across communication rounds. Performance improves sharply in the initial rounds as the model aggregates diverse gradients from clients, and then gradually converges, stabilizing after approximately 30-40 rounds. This demonstrates that FedAvg can efficiently converge in this non-IID financial fraud setting with the chosen parameters.

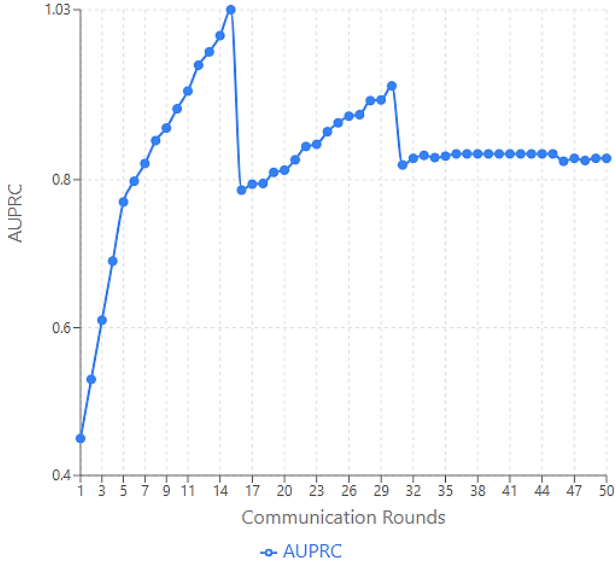


Fig. 5: Convergence Chart (DeFRAD)

E. Sensitivity Analysis

To examine robustness, a limited sensitivity analysis is performed on two FL hyperparameters: client fraction C and number of rounds T .

- i. Client fraction C : Reducing C from 0.1 to 0.05 increases variance in per-round performance and

slightly slows convergence, but final AUPRC remains close to the baseline after more rounds are used.

- ii. Number of rounds T : Increasing T beyond 50 yields diminishing gains; most performance improvements occur within the first 30-40 rounds, indicating that communication can be bounded without sacrificing much accuracy.

These observations, as shown in Figure 6, suggest that DeFRAD's performance is not overly sensitive to moderate adjustments in participation rate and communication budget, which is important for practical deployments where clients might not be constantly available.

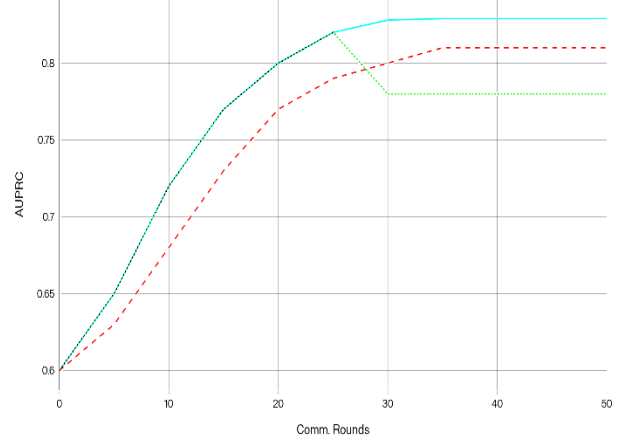


Fig. 6: Sensitivity Analysis Chart (DeFRAD)

F. Latency and Communication Analysis

Latency and communication are evaluated using an edge-oriented simulation that models the network between edge clients and the server. The simulation assumes typical mobile/retail connectivity parameters such as tens of milliseconds round-trip time, limited uplink bandwidth and uses measured model sizes to estimate communication volume. The comparison in Table 6 shows that in the centralized scenario, each transaction incurs a network round trip time plus server-side processing, leading to hundreds of milliseconds of end-to-end latency.

TABLE VI. EDGE-ORIENTED LATENCY AND COMMUNICATION COMPARISON

Metric	Centralized MLP	Federated MLP (DeFRAD)
Inference latency	~ 300-500ms (network RTT + server processing)	< 20ms on-device (no network)
Data transmitted (inference)	Full transaction features per request	None (local inference)

Metric	Centralized MLP	Federated MLP (DeFRAD)
Data transmitted (training)	All raw training data (MBs–GBs) to server	Model updates only (KBs per round per client)

In the edge scenario (DeFRAD), once the model is deployed locally, inference occurs entirely on device, yielding sub-20ms decisions and significantly improved user experience at POS or in mobile apps. From a training perspective, FL transmits only serialized model updates (weights or gradients), which are orders of magnitude smaller than raw transaction datasets, substantially reducing bandwidth and storage requirements.

These numbers are derived from simulation and approximate hardware-level benchmarking rather than from deployment on physical POS devices, and thus should be interpreted as indicative rather than definitive.

G. Discussion

1) Our Findings

The experiments indicate that DeFRAD can achieve predictive performance that approaches that of a centralized model on a realistic card fraud dataset, even under non-IID client distributions. Collaborative learning substantially improves over isolated local training, demonstrating the necessity of cross-client parameter sharing in data-sparse, imbalanced environments. Latency and communication analyses further suggest that an FL-plus-Edge approach can feasibly reduce both inference latency and long-term bandwidth usage compared with fully centralized processing.

a) Recall

This is the sensitivity or true positive rating of the framework, the direct measure of a fraud detection system’s effectiveness in preventing loss due to fraudulent transactions. Each unit of increase in recall means more fraudulent transactions are identified and potentially stopped before funds are lost. In our evaluation scenario where 50 fraudulent transactions occurred, catching 45 (90% recall) instead of 15 (30% recall) is a massive difference in potential financial impact. The 30 un-intercepted frauds from the isolated model would represent substantial monetary losses.

b) Precision

This is the metric that measures customers’ experience, minimizing inconvenience and preserving trust. A low measure indicates that customers are bombarded with notifications for legitimate transactions which will require them to reattempt the transactions or even lead to temporary account freeze. Such experiences erode customer’s trust and loyalty to the system, increasing operational cost related to customer support for false alerts. The high precision measure of DeFRAD reflects that flagged transactions are highly likely to be actual fraud, making interventions meaningful and less disruptive for legitimate transactions.

c) AUC-ROC

The Area Under the Receiver Operating Characteristic Curve measures the system’s ability to distinguish between fraudulent and legitimate transactions. The high AUC-ROC, achieved in our study, indicates the framework is able to accurately rank and predict fraudulent transactions, signifying the overall robustness of the DeFRAD framework. It provides a foundational assurance that DeFRAD has successfully learnt meaningful patterns and can adequately differentiate fraudulent from non-fraudulent transactions.

d) AUPRC

The Area Under Precision-Recall Curve focuses on the trade-off between precision and recall such that a higher AUPRC indicates better performance. The measure of AUPRC 0.829 achieved by DeFRAD indicates that a large percentage of fraudulent transactions are identified and stopped by the framework, substantially lowering financial losses for institutions that deploy the DeFRAD framework. In addition, a marginal increase in false positives, compared to the centralized model, presents an acceptable trade-off in DeFRAD where customers are not subjected to inconveniences while still catching a high proportion of fraud.

2) Privacy, Trust, and Regulatory Considerations

By keeping raw transaction data on devices and only sharing model updates, DeFRAD reduces the amount of sensitive data transmitted and centralized, which can help lower exposure to certain classes of data breaches. However, standard FL alone does not guarantee formal privacy against inference attacks, nor does it inherently defend against adversarial clients or servers. To meet stringent regulatory and ethical requirements in financial services, additional mechanisms such as secure aggregation, differential privacy, and robust aggregation would need to be integrated and evaluated. From a regulatory standpoint (such as GDPR, NDPA), FL can support data-minimization principles by avoiding centralized raw data storage, but compliance still depends on end-to-end system design, including logging, consent, auditing, and incident response procedures which are beyond the scope of this paper.

3) Edge Deployment Considerations

The edge-oriented architecture of DeFRAD is designed to meet the stringent latency requirements of financial transactions, enabling real-time fraud detection decisions directly at the point of interaction. The lightweight MLP model, optimized through post-training quantization (discussed in Section 3.8.4), significantly reduced its compute and memory footprint. This optimization is critical because it allows the model to execute within sub-20ms inference times on resource-constrained edge devices (e.g., POS terminals, mobile banking apps), thereby minimizing transaction delays and enhancing user experience without compromising detection accuracy. This study demonstrates the architectural feasibility of delivering high-performance fraud detection within the operational constraints of modern financial ecosystems. However, future real-world

deployments may include detailed profiling of CPU usage and energy consumption of the edge devices.

Criteria	Metrics	Key result	Implication
Predictive performance	Centralized: 0.975 DeFRAD: 0.971 Isolated: 0.823	DeFRAD closely matches centralized accuracy	FL on non-IID data achieves near-centralized quality without sharing raw data
Predictive performance	Centralized: 0.841 DeFRAD: 0.829 Isolated: 0.596	Strong fraud-specific detection quality preserved	DeFRAD outperforms isolated models by +23.3pp, the largest margin across all metrics
Recall (fraud detection)	Centralized: 94% DeFRAD: 90% Isolated: 30%	DeFRAD intercepts 45 of 50 fraud cases vs 15 for isolated models	30 additional fraudulent transactions caught per 50, implies direct reduction in financial loss
Precision	Centralized: 82% DeFRAD: 75% Isolated: 38%	DeFRAD keeps false positives low and customer disruption minimal	Modest increase in false positives (from 10 to 15) vs centralized is an acceptable trade-off for privacy
Inference latency	Centralized: 300 - 500ms (network RTT) DeFRAD: <20ms (no network)	On-device inference is under 20ms, real-time capable	Meets strict card payment authorization time budgets; eliminates network dependency
Communication cost	Centralized: MBs -GBs (raw data) · DeFRAD: 5 - 10 KB per round per client	FL training transmits orders of magnitude less data than	Substantial bandwidth and infrastructure cost savings,

Criteria	Metrics	Key result	Implication
		centralized training	especially at scale
FL convergence	Sharp AUPRC gain in early rounds; diminishing returns beyond round 40 - 50	Model converges within 30 - 40 rounds, stabilizes after 50	Communication budget can be capped at ~40 rounds without significant accuracy loss
Sensitivity to hyperparameters	Halving client fraction (0.1→0.05) increases variance but not final AUPRC by much	Performance is robust to moderate changes in client fraction and rounds	DeFRAD remains viable in deployments with inconsistent client availability
Privacy posture	Only model weights transmitted; no raw financial data centralized at any point	Raw transaction data never leaves edge devices, supports GDPR/NDPA data minimization	Reduces breach exposure, but formal guarantees (DP, secure aggregation) are not yet included
Value of collaboration	AUC-ROC gap between DeFRAD and isolated: +14.8pp; AUPRC gap: +23.3pp	Cross-client parameter sharing is essential; local-only training is inadequate	Isolated edge models lack sufficient fraud diversity to train robust detectors

4) Limitations

Despite the promising performance of DeFRAD, the following are some identified limitations of this study:

- i. A single public dataset for training and tests: Based on this limitation, the results may not generalize to all fraud patterns, especially those present in other regions or institutions.
- ii. Incomplete data distribution: The non-IID partitioning is simulated and may not capture all nuances of real merchant-level data distributions.
- iii. Privacy guarantees are limited to data locality; advanced privacy and security mechanisms were not implemented or evaluated.
- iv. Simulation-based Analysis: Latency and communication results are derived from simulations

rather than real deployments on physical POS or mobile hardware.

These limitations suggest that while the results are promising, they should be interpreted as an initial feasibility study rather than a fully deployable solution.

V. CONCLUSION

This paper presented DeFRAD, a federated fraud detection framework evaluated in a simulated edge-oriented environment using the Kaggle credit-card dataset. Experiments showed that a federated MLP trained under non-IID client distributions can achieve near-centralized predictive performance and substantially outperform isolated on-device models. Latency and communication analysis clearly indicates that combining FL with on-device inference offers a favorable trade-off between fraud detection accuracy, qualitative user experience, and data transmission requirements.

A. Future Directions

We have identified future research studies along three directions. First is the deployment and evaluation on actual embedded or POS hardware to measure real-world latency, actual throughput, and energy consumption. Second, integration of more advanced FL algorithms (such as FedProx or robust aggregation methods) and privacy-enhancing technologies (secure aggregation, differential privacy) to strengthen the financial ecosystem's resilience against non-IID data and adversarial behaviors. Thirdly, exploration of broader empirical validation across multiple financial datasets, to assess the generalizability of DeFRAD and support more comprehensive cost-benefit analyses for industry adoption.

REFERENCES

- Abdul Salam, Mustafa, Fouad, Khaled.M., Elbably, Do haa.L., & Elsayed, Salah M. (2024). Federated learning model for credit card fraud detection with data balancing techniques. *Neural Computing and Applications*, 36, 6231-6256.
- AbouGrad, H., & Sankuru, L. (2025). Online Banking Fraud Detection Model: Decentralized Machine Learning Framework to Enhance Effectiveness and Compliance with Data Privacy Regulations. *Mathematics*, 13(13), 2110. <https://doi.org/10.3390/math13132110>
- Bagdasaryan E., A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, How to Backdoor Federated Learning, in *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020.
- Bhattacharyya S., S. Jha, K. Tharakunnel, and J. C. Westland, Data mining for credit card fraud: A comparative study, *Decision Support Systems*, vol. 50, no. 3, pp. 602–613, 2011
- Chawla N.V., K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, SMOTE: Synthetic Minority Over-sampling Technique, *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.
- Chen T. and C. Guestrin, XGBoost: A Scalable Tree Boosting System, in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.
- Crowe. Fraud Costs the Global Economy Over US\$5 trillion. Accessed: May 31, 2025. [Online]. Available: <https://www.crowe.com/global/news/fraud-costs-the-global-economy-over-us%245-trillion>
- Dal Pozzolo A., O. Caelen, R. A. Johnson, and G. Bontempi, Calibrating Probability with Undersampling for Unbalanced Classification, in *2015 IEEE Symposium Series on Computational Intelligence and Data Mining (SSCI)*, 2015.
- Dal Pozzolo A., G. Boracchi, O. Caelen, C. Alippi, and G. Bontempi, Credit Card Fraud Detection: A Realistic Modeling and a Novel Learning Strategy, *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 8, pp. 3784–3797, 2018.
- European Parliament and Council of the European Union, Regulation (EU) 2016/679 of the European Parliament and of the Council, *Official Journal of the European Union*, 2016.
- Han S., H. Mao, and W. J. Dally, Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding, in *International Conference on Learning Representations (ICLR)*, 2016.
- Jiang, S.; Dong, R.; Wang, J.; Xia, M. Credit Card Fraud Detection Based on Unsupervised Attentional Anomaly Detection Network. *Systems* 2023, 11, 305.
- Li, M., & Walsh, J. (2024). FedGAT-DCNN: Advanced Credit Card Fraud Detection Using Federated Learning, Graph Attention Networks, and Dilated Convolutions. *Electronics*, 13(16), 3169.
- Li T., A. K. Sahu, A. Talwalkar, and V. Smith, Federated Optimization in Heterogeneous Networks, in *Proceedings of Machine Learning and Systems 2020 (MLSys)*, 2020.
- Ngai E.W.T., Y. Hu, Y. H. 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 System*, vol. 50, no. 3, pp. 559–569, Feb. 2011.
- PwC, Global Annual Review 2021. Accessed: Dec. 10, 2025. [Online]. Available: <https://www.pwc.com/gx/en/about-pwc/global-annual-review2021/downloads/pwc-global-annual-review-2021.pdf>
- Saif Khalifa Aljunaid, Saif Jasim Almheiri, Hussain Dawood, Muhammad Adnan Khan, Secure and Transparent Banking: Explainable AI-Driven Federated Learning Model for Financial Fraud Detection, *Journal of Risk and Financial Management (JRFM)*, Volume 18, Issue 26, 2025, DOI: 10.3390/jrfm18040179
- Saikat Islam Khan, Oshani Seneviratne, 2024, Fed-RD: Privacy-Preserving Federated Learning for Financial Crime Detection. *arXiv preprint arXiv:2408.01609*.
- Sandeep Dasari, Rajesh Kaluri, 2P3FL: A Novel Approach for Privacy Preserving in Financial Sectors Using Flower Federated Learning, *CMES - Computer Modeling in Engineering and Sciences*, Volume 140, Issue 2, 2024, Pages 2035-2051, ISSN 1526-1492, <https://doi.org/10.32604/cmescs.2024.049152>.
- Shi W. and S. Dustdar, "The Promise of Edge Computing," *Computer*, vol. 49, no. 5, pp. 78–81, 2016.
- TensorFlow Lite Developers, TensorFlow Lite Guide. Accessed 2025. [Online]. Available: <https://www.tensorflow.org/lite>
- Tomisin Awosika, Raj Mani Shukla, and Bernardi Pranggono, 2023, Transparency and Privacy: The Role of Explainable AI and Federated Learning in Financial Fraud Detection. *arXiv preprint arXiv:2312.13334*
- Yin, C.; Zhang, S.; Wang, J.; Xiong, N.N. Anomaly Detection Based on Convolutional Recurrent Autoencoder for IoT Time Series. *IEEE Trans. Syst. Man Cybern. Syst.* 2022, 52, 112–122.
- Zhang T., Y. Li, and C. L. Philip Chen, "Edge computing and its role in Industrial Internet: Methodologies, applications, and future directions," *Inf. Sci. (N Y)*, vol. 557, pp. 34–65, May 2021, doi: 10.1016/j.ins.2020.12.021