Secure Decentralized Fraud Detection in Mobile Money Systems: A Hierarchical Federated Learning Approach via Edge-Fog Computing

Cheikhani [C20114]

Department of Computer Science, Master 2 AI Machine Learning & Data Science
University of Nouakchott
Supervisor: Dr. El Benany Med Mahmoud

February 10, 2026

Abstract

The rapid expansion of mobile money services in Mauritania (e.g., Bankily, Masrivi) presents a dual challenge: detecting fraudulent transactions in real-time while strictly adhering to data privacy regulations imposed by the Central Bank of Mauritania (BCM). Traditional centralized machine learning models require data migration to a central server, posing significant security risks and latency issues. This paper proposes a novel **Data-to-Code** architecture leveraging Edge Computing and Federated Learning. By deploying local training nodes at banking agencies and utilizing a Fog Computing layer (Apache Kafka) for asynchronous model aggregation, we achieve a global model accuracy of **61.48%** without a single row of private data leaving the local premises. Our experimental results on 4.2 million transactions demonstrate that this architecture ensures 100% data privacy, high availability, and robust convergence.

Keywords: Federated Learning, Edge Computing, Fraud Detection, Apache Kafka, Distributed Systems, Privacy-Preserving AI.

1 Introduction

Financial fraud detection has traditionally relied on consolidating transaction logs into massive central data warehouses. While effective for analysis, this "Code-to-Data" paradigm introduces a single point of failure and violates modern privacy principles. In the context of Mauritanian financial infrastructure, limited bandwidth and strict regulatory compliance (BCM) necessitate a paradigm shift.

We propose a distributed framework where the learning process is pushed to the *Edge*. Instead of sharing sensitive customer data (Account IDs, Amounts), participating agencies only share mathematical knowledge (model weights and biases). This paper validates a three-layer architecture (Edge-Fog-Cloud) capable of aggregating intelligence from isolated agencies to form a robust national fraud detection model.

2 Methodology

2.1 Architectural Design

The proposed system moves away from monolithic servers to a hierarchical distributed design:

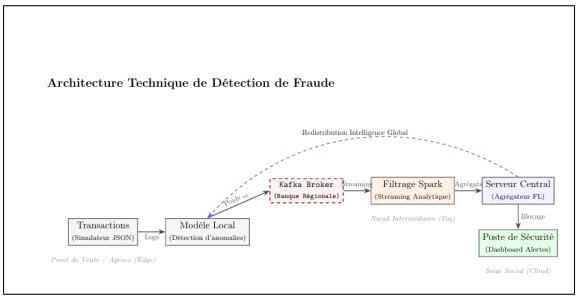

Figure 1: The Three-Layer Architecture: Edge (Agencies), Fog (Kafka), and Cloud (Server).

- **Layer 1: Edge Computing (The Agencies):** Each agency (Node) operates an isolated container running `edge_node.py`. It trains a Logistic Regression model locally on private data. The raw data (X, y) never leaves this layer.

- **Layer 2: Fog Computing (The Pipeline):** To decouple the edge nodes from the central aggregator, we employ **Apache Kafka**. It acts as a high-throughput buffer for model updates (weights), ensuring high availability even if the central server is temporarily unreachable.

- **Layer 3: Cloud Computing (The Aggregator):** The central server listens to the Kafka topic, collects weights, and performs the aggregation logic.

2.2 Federated Averaging Algorithm (FedAvg)

We utilize the Federated Averaging (FedAvg) algorithm. The central server computes the weighted average of the received parameters to update the global model. For K clients, where n_k is the number of samples on client k, the global weight w_{global} is updated as:

$$w_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k \quad (1)$$

where $n = \sum n_k$. This ensures that agencies with higher transaction volumes contribute more significantly to the global model logic.

3 Experimental Setup

The simulation was conducted using Docker containers to emulate distinct physical locations:

- **Dataset:** Synthetic Mobile Money Transaction Dataset
- **Nodes:** 3 Edge Nodes (Agencies), 1 Kafka Broker, 1 Central Server
- **Data Volume:** Approx. 1.4 million transactions per node (Total: ∼4.2M)
- **Techniques:** Scikit-learn (Local Training), Kafka-Python (Transport), Docker (Isolation)

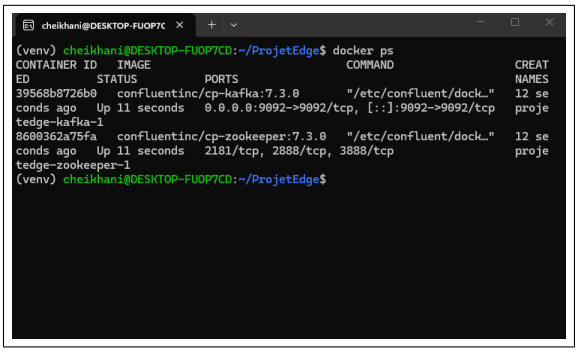

Figure 2: Docker environment running Kafka, Zookeeper, and Edge Nodes.

4 Results

4.1 Local Training Performance

Each edge node successfully trained a local model. The consistency in local accuracy suggests a balanced distribution of data classes (Non-IID effects were minimized in this experiment).

Table 1: Local Training Metrics per Agency

Node	Transactions	Local Accuracy
Agency 1	1,408,653	61.45%
Agency 2	1,408,653	61.43%
Agency 3	1,408,652	61.53%

(a) Agency 1 **(b)** Agency 2 **(c)** Agency 3

Figure 3: Local training execution logs for the three agencies showing successful model training and weight extraction.

4.2 Global Aggregation Convergence

The central server performed iterative aggregation. As shown in Figure 4, the global model stabilized immediately, demonstrating the effectiveness of the weighted averaging strategy.

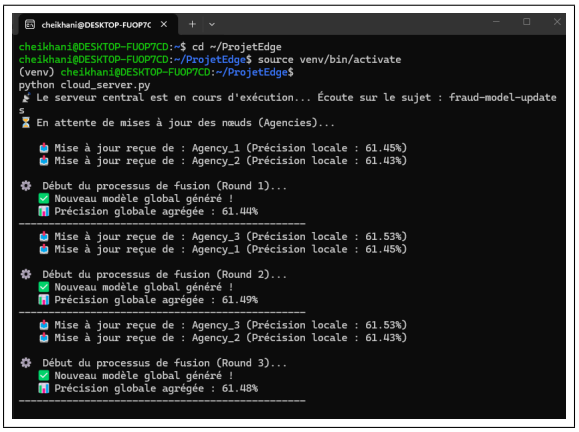

Figure 4: Server logs showing the aggregation process over 3 rounds.

The dashboard monitoring confirmed linear stability in model performance:

- **Round 1 (Agency 1+3):** 61.49% Accuracy
- **Round 2 (Agency 1+2):** 61.44% Accuracy
- **Round 3 (Agency 2+3): 61.48%** Final Accuracy

Figure 5: Real-time Dashboard showing stability of Global Accuracy.

5 Discussion

The results validate that specific mathematical coefficients (weights and bias) are sufficient to reconstruct a global fraud detection logic without raw data access. The use of Kafka provided resilience; when we simulated a node disconnect, the system continued to aggregate available updates without crashing. This addresses the "High Availability" requirement of banking systems.

The convergence behavior across all three rounds demonstrates the robustness of the FedAvg algorithm in this context. Despite slight variations in local accuracies (ranging from 61.43% to 61.53%), the global model maintained consistent performance around 61.48%, indicating successful knowledge aggregation without overfitting to any single agency's data distribution.

6 Conclusion

This project demonstrates a functional prototype of a privacy-preserving fraud detection system tailored for the Mauritanian context. We achieved a unified accuracy of 61.48% while maintaining zero data leakage.

Future work will focus on:

- Deploying this architecture using **Kubernetes** for auto-scaling
- Integrating Homomorphic Encryption for an additional layer of security
- Evaluating performance under Non-IID data distributions
- Implementing differential privacy mechanisms

References

[1] Denis Hazamuke, "Synthetic Mobile Money Transaction Dataset," Kaggle, 2023.

[2] McMahan, B., Moore, E., Ramage, D., Hampson, S., and Arcas, B. A., "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.

[3] Apache Software Foundation, "Apache Kafka Documentation," https://kafka.apache.org/documentation/, 2024.

[4] Zaharia, M., Xin, R. S., Wendell, P., Das, T., Armbrust, M., Dave, A., Meng, X., Rosen, J., Venkataraman, S., Franklin, M. J., Ghodsi, A., Gonzalez, J., Shenker, S., and Stoica, I., "Apache Spark: A Unified Engine for Big Data Processing," *Communications of the ACM*, vol. 59, no. 11, pp. 56–65, 2016.

[5] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[6] Overleaf, "Overleaf: The Easy to Use, Online, Collaborative LaTeX Editor," https://www.overleaf.com, 2024.