Computer Modeling in Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2024.049152

ARTICLE

2P3FL: A Novel Approach for Privacy Preserving in Financial Sectors Using Flower Federated Learning

Sandeep Dasari and Rajesh Kaluri*

School of Computer Science Engineering and Information Systems, Vellore Institute of Technology, Vellore, 632014, India

*Corresponding Author: Rajesh Kaluri. Email: rajesh.kaluri@vit.ac.in

Received: 29 December 2023 Accepted: 06 March 2024 Published: 20 May 2024

ABSTRACT

The increasing data pool in finance sectors forces machine learning (ML) to step into new complications. Banking data has significant financial implications and is confidential. Combining users data from several organizations for various banking services may result in various intrusions and privacy leakages. As a result, this study employs federated learning (FL) using a flower paradigm to preserve each organization's privacy while collaborating to build a robust shared global model. However, diverse data distributions in the collaborative training process might result in inadequate model learning and a lack of privacy. To address this issue, the present paper proposes the implementation of Federated Averaging (FedAvg) and Federated Proximal (FedProx) methods in the flower framework, which take advantage of the data locality while training and guaranteeing global convergence. Resultantly improves the privacy of the local models. This analysis used the credit card and Canadian Institute for Cybersecurity Intrusion Detection Evaluation (CICIDS) datasets. Precision, recall, and accuracy as performance indicators to show the efficacy of the proposed strategy using FedAvg and FedProx. The experimental findings suggest that the proposed approach helps to safely use banking data from diverse sources to enhance customer banking services by obtaining accuracy of 99.55% and 83.72% for FedAvg and 99.57%, and 84.63% for FedProx.

KEYWORDS

Federated learning; FedAvg; FedProx; flower framework; privacy preservation; financial sectors

1 Introduction

ML is rapidly gaining popularity in data-influenced environments. It provides the foundation for many innovations, which have been widely embraced and utilized in our daily lives. The COVID-19 epidemic has drawn attention to the essential importance of digitization in almost all sectors, where data privacy and security are critical in sensitive areas like banking and the Defence Research and Development Organisation (DRDO). When data is dispersed across several parties, it would be adventurous to consolidate it into one location to collect the necessary data to construct high-quality models. Banking sectors develop hundreds of gigabytes of financial information daily, implying that data transmission and storage virtually lead to various data risks. Also, it contains privacy information, which raises questions about the transmission and storage of data on a central server, and due to increasing privacy and security concerns, data integration might be challenging.

FL is a possible solution for all the above challenges, facilitating many stakeholders to train a model while preserving privacy effectively. FL is an ML paradigm that enables participants to fine-tune ML models with data distributed over multiple mobile devices rather than consolidating data in one location. In addition, FL has become prevalent over the past couple of years. The reason for getting a wide range of acceptance in various fields is its ability to overcome the privacy issues raised by standard ML methodologies. Additionally, it utilizes multiple strategies to aggregate the global model [1]. FedAvg [2] is one of the most often used algorithms in the FL process. It is employed widely due to its ease of use and inexpensive communication costs. A random selection of clients, the aggregation of local model updates, and the formation of a global model constitute the general operation of FedAvg [3].

Several modifications to the FedAvg algorithm will provide numerous alternative algorithms, federated Proximal (FedProx) and quantized Federated Averaging (QFedAvg), use a quantization process to compress the model updates and reduce communication overhead between clients and servers. It requires additional computation to quantify the model updates. Federated Averaging momentum (FedAvgM) uses momentum to accelerate the algorithms convergence. It converges faster than FedAvg. Fault-Tolerant Federated Averaging (FaultTolerant FedAvg). It is resilient to client failures without affecting the training process. Federated secure aggregation (FedSecAgg) provides privacy guarantees for the clients. Federated curvature (FedCurv) improves the model updates. Federated dynamic weighting (FedDyn) applies a dynamic weighting process to adjust the clients contribution. It can handle non-IID data distributions [4]. Federated stochastic gradient (FedSGD) requires a large number of clients to achieve good performance. Federated differential privacy (FedDP) provides strong privacy guarantees for clients. The advancements offered by FedAvg are beneficial to several server-side optimization algorithms, including FedAdagrad, FedYogi, and FedAdam [5]. Table 1 provides the related works done using various FedAvg variation algorithms. To address system heterogeneity, FedProx [6] was proposed as a generalization and reparameterization of FedAvg. It is another widely employed algorithm. It consists of an additional proximal term that enhances system stability by reducing the deviation of the averaged model from global optima.

There are multiple barriers to implementing FL in the banking sector. One major issue is the need to strike a compromise between data privacy and model accuracy. Financial data is sensitive, and it will be challenging to maintain ideal model training while adhering to rules. It is also difficult to handle the variability of data among many institutions. The diverse data forms and formats shown by financial firms provide a challenge in developing a unified FL methodology. Delivering robust security protocols to thwart any intrusions on the FL mechanism is essential. Upholding the FL system's integrity and confidentiality is crucial as financial data is a popular target for criminal activity.

Furthermore, it might be challenging to collaborate and communicate effectively between various financial organizations without losing the confidentiality of data. Collaborating on FL requires establishing explicit standards and building trust. Another major challenge is regulatory compliance. Financial organizations must navigate a multifaceted regulatory environment to guarantee that their FL implementations comply with industry standards and legal obligations.

Data centralization is one of the traditional approaches used in financial sectors, which provides practical analytical benefits but also causes significant risks with profound privacy implications on the transparency of customer details. The present work addresses this issue through FedAvg and FedProx, using inherent in standard machine learning algorithms by storing the bulk of the data on individual devices throughout the training phase. This decentralization has various benefits in terms of privacy.

FedAvg and FedProx help to address the issue by:

i. Reduced attack surface: With data shared among several devices, there is no single point of vulnerability or central repository to target. This makes it far more difficult for malicious parties to steal or influence customer details.

ii. Improved data privacy: Users maintain control over their data since only localized model changes are forwarded to the server for aggregate. This reveals less sensitive information than transmitting raw data to a central server.

To the best of our knowledge, this is the first research work to use the FedAvg and FedProx algorithms in financial applications using the Flower framework.

This paper is structured as follows: Section 2 provides a flower framework, while Section 3 discusses the various related works on FL. Section 4 presents the different specifications for implementing FL algorithms; the proposed methodology is discussed in Section 5. Section 6 presents the paper's conclusions and provides insights into future works.

1.1 Major Distinguishing Features of Federated Learning with Other Technologies

i. Participants with diverse devices and distinct computing abilities: Participants (devices) linked to the learning network. These devices can be heterogeneous and have varying computational abilities. FL connects various computing devices into a decentralized network, enabling individual data collection devices to contribute to model training. Local ML models are trained on local heterogeneous datasets during training. This improves data protection and cybersecurity.

ii. Federated learning prioritizes privacy and security: FL prioritizes privacy and security by keeping the data local to the devices that collect it, reducing the risk of data breaches and cyber-attacks [7].

2 Flower Framework

In FL, the flower framework facilitates research with both system-related and algorithmic difficulties. Flower was developed with AI research in mind since it was produced as a result of research at the University of Oxford. Many components have grown and overridden to create new cutting-edge systems. It is a framework for developing FL systems. It provides a stable framework for independent synthesis of core FL system components and higher-level interfaces, which enable researchers to investigate and implement cutting-edge concepts on top of a reliable stack. It also allows the rapid transformation of existing ML model training pipelines to an FL design to analyze their evolving characteristics and training outcomes in a federal environment. Flower significantly supports expanding FL implementations to remote and virtual clients with various computing, storage, and networking capabilities [8]. It is a revolutionary end-to-end FL platform that allows a more seamless transition of simulation-based and exploratory research to system study on a large cohort of real-world edge devices. Flower has unique capabilities in both simulation and real-world applications. The opportunity for experiment implementations to move between the two extremes is required throughout research and development. Due to system-level restrictions such as network bandwidth in mobile devices, memory and computing capabilities are significant obstacles. Flower includes built-in features to solve many of these difficult scenarios in a cloud-based environment, allowing for real-world scenarios of FL algorithms. Finally, the flower is built for scalability, enabling large-cohort research using numerous connected clients and training. The ability to do FL on larger scales will allow for new paths of investigation since the results of small-scale experiments sometimes translate poorly to large-scale problems. Fig. 1 gives the entities involved in the flower framework.

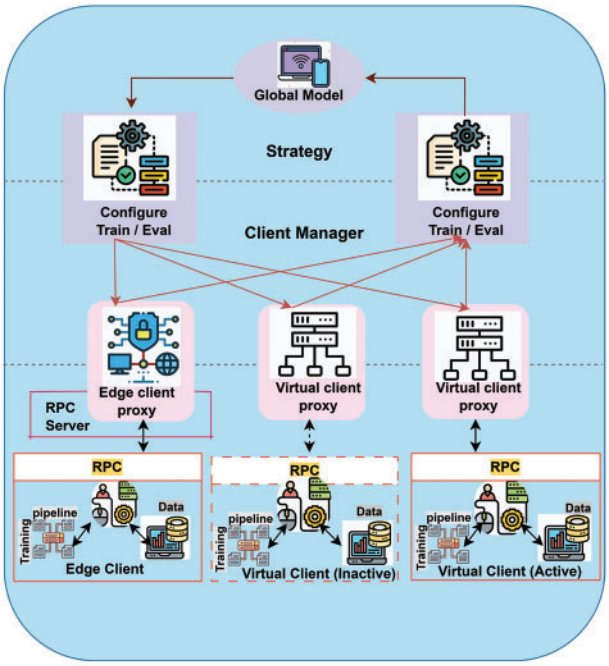

Figure 1: Federated flower framework

2.1 Governing Principles of Flower FL

The adaptability of the flower FL architecture is one of its significant potentials; other interesting features are:

i. Configurable components: Users can customize various options like selecting the training, updating fusion, and client communication techniques. Determining which clients should divide the data across cycles and federated analytics.

ii. Flexible framework: Customise the activities of the server, the compressed data, and the client training. Develop new communication protocols or adapt existing ones.

iii. Indifferent to different machine learning frameworks, such as PyTorch, TensorFlow, etc. Floweras flexible FL architecture and preferred tools may be used with its seamless integration with other programs.

iv. Creating communities: Analysing algorithms developed and learning from Floweras active community assist in customizing and growing the Flower framework. Messages between the server and the client. The FL loop is crucial to the FL process since it controls the whole learning process.

v. Dependability: Flower's distributed and fault-tolerant architecture, along with device heartbeat monitoring and straggler management features, improves handling device failures and network difficulties. It imposes stringent security measures, such as safe aggregation, encryption, and differential privacy for data protection. Error recovery is ensured by implementing methods for error handling and node failure recovery. It can resume unsuccessful processes across several devices, ensuring overall dependability.

vi. Scalability: Flower's client-server architecture supports horizontally scalable properties, which add additional devices or servers to meet rising data and computing needs. It uses a distributed strategy

in which devices train models locally and send changes to a central server for aggregation. It also helps to improve server-side speed by increasing server resources such as CPU and memory. For device-side scalability, evaluates devices with higher computing capability. Flower implements methods like dynamic device selection and adaptive batching to balance workloads among devices, resulting in more effective utilization of resources.

2.2 The Main Contributions of Current Work Include

i. A novel framework for preserving privacy as 2P3FL was proposed for a distributed network for maintaining privacy by leveraging FL. Furthermore, we study the impact of two FL methods, FedProx and FedAvg, on the model performance in privacy preservation.

ii. Federated flower environments integrated with FedAvg and FedProx were applied to the features extracted from the credit card and CICIDS dataset.

iii. Conduct extensive evaluations of 2P3FL using the aforementioned datasets, using performance evaluation metrics such as accuracy, precision, and recall.

3 Related Works

This section highlights the major works on Fedavg and FedProx under various circumstances, like in non-iid environments, where multiple strategies and metrics are applied to protect data privacy. The following works are considered for the analysis of the current work.

Deng et al. [9] performed experimental evaluations on several criteria in the experimental study. The proposed method beats basic FL techniques, including FedAvg and Paillier-encryption-based PPDL, on the MNIST dataset while demonstrating strong convergence properties [10]. Nilsson et al. [11] presented that FedAvg obtains the maximum accuracy among fedAvg algorithms, according to tests carried out on FedAvg and FedProx. FedProx concerns variances in computing capacity and other aspects of devices partaking in the FL training iterations. FedProx additionally includes a proximal term to address non-uniformity in local updates. Xie et al. [12] worked on asynchronous FedAvg does not need iid data. The restrictions it imposes do not ensure convergence to the local minimum value. Nilsson et al. Li et al. [13] provided an empirical assessment of the behavior of FedAvg in common non-IID scenarios.

Sahu et al. [6] presented FedProx, an optimization framework in FL for dealing with statistical differences, and given convergence assurances in non-iid environments. Additionally, it augments each local target with a proximal term. FedProx declines in comparison to FedAvg when these proximal elements are removed. Sattler et al. [14] investigated the non-iid environment, which includes convergence rate. FL, and hierarchical clustering were presented as a combination to enhance the learning rate on non-IID data [15]. This method combines comparable individuals' local models in order to reduce divergence. FedAvg is used for the initial aggregation, and then the AHC method is employed to choose clusters of comparable local models.

FedAvg was tested on a range of audio, video, and text datasets to simulate self-reported emotional experience and perceptual labels [16]. Identified two learning paradigms typically encountered in emotional computing tasks: modeling self-reports (user-as-client) and modeling perceptual judgments (rater-as-client), such as labeling sentiment of online comments. FedAvg was used to simulate those activities, and it presents the FedRater method, which learns client-specific label distributions in federated environments. The outcomes reveal that FedRater outperforms FedAvg in terms of global classification performance and gives results to determine proxies of inter-rater agreement in distributed

environments. Liu et al. [17] examined the performance of two techniques within the Federated Learning (FL) paradigm: FedAvg a Per-FedAvg using Non-IID data. Rubaie et al. [18] addressed the important problems involved with ensuring data privacy via FL. It was tackled utilizing practical attack techniques, and the related answers to the relevant attack are noted. Several research elements are also highlighted, as well as prospective possibilities and applications with FL. Li et al. [19] proposed the paper to illustrate the feasibility of the approach for practical data science by conducting evaluations with the proposed MPC protocols for feature selection in a commonly used machine-learning-as-a-service configuration with semi-honest and malicious adversaries. The authors demonstrate that secure feature decision-making using the suggested protocols enhances classifier accuracy on real-world data sets without revealing information about feature values.

Collins et al. in [20] applied FedAvg generalizability was examined in the context of multi-task linear modeling. Outcomes show that FedAvg output's generalizability is due to its capacity to identify the standard information representations across the client's jobs by taking advantage of the variety among client data distributions via local updates. Xing et al. [21] proposed an FL methodology as N-FedAvg based on FedAvg. It chooses clients in a series before every session, decreasing randomization in client selection and allowing all clients' data to be involved in federated learning but preventing data locality from specific clients from partaking in collaboration with less chance of occurring. Optimize the objective function using gradient descent to achieve the global optimum of the function's loss value. The rate of learning was reduced to get the model to be as near to this point as feasible, and cosine processing can be attained and generates favorable outcomes by the cosine function. The sparsity technique is a model compression method that not only communicates a limited number of parameters but also decreases network traffic between the server and clients and may avoid global model parameter leaking. On the CIFAR-10 dataset, the N-FedAvg method suggested work was found to be 1.34% more reliable than the standard FedAvg algorithm, and the loss function value was found to be 2.77% lower. The authors demonstrate that the amount of data heterogeneity, as represented by a Dirichlet distribution, has a considerable impact on the effectiveness of both techniques, with Per-FedAvg outperforming FedAvg under settings of high variability. Some impact works on FedAvg and FedProx are presented in Table 2.

Table 1: Recent evaluation works on various FL algorithms

Ref.	Objective	Metrics	Proposed/applied method	Outcomes
[22]	Secure aggregation and privacy preservation	Privacy, model accuracy	Cryptographic techniques, FedAvg	Privacy protection and accuracy affected by cryptographic overhead

(Continued)

Table 1 (continued)

Ref.	Objective	Metrics	Proposed/applied method	Outcomes
[23]	Adaptive personalization for user-specific models	Model accuracy, personalization level	Adaptive regularization, FedAvg	Higher personalization, Potential privacy concerns with more data shared
[24]	Improve convergence speed and stability	Model accuracy, Convergence speed	Federated averaging with momentum	Faster convergence, Improved accuracy in some cases
[25]	Reduce the communication cost while maintaining model accuracy	Communication cost, model accuracy	Double gradient descent, quantized averaging	Reduced communication cost, Balanced accuracy loss
[26]	Faster convergence with reduced communication cost	Convergence speed	Communication-efficient FedSGD	Reduced communication cost
[27]	Robustness against malicious participants with efficient aggregation	Model accuracy, byzantine fault tolerance	Partially byzantine tolerant (PBT) aggregation, FedAvg	Strong fault tolerance, lower communication overhead
[28]	Ensure secure aggregation against malicious participants	Privacy, byzantine fault tolerance	Secure multi-party computation (MPC), FedAvg	Privacy preservation, Byzantine fault tolerance, Aggregation overhead
[29]	Improve generalizability and efficiency	Model accuracy, communication cost	Model curvature estimation, FedAvg	Reduced communication cost

Table 2: Major works in FL using FedAvg and FedProx

Ref.	Objective	Metrics	Proposed/applied method	Outcomes
[30]	Propose a privacy-preserving method for detecting sensitive data exposure called data leak detection (DLD)	Detection rate, false positive, negative rate	DLD algorithm	Detecting sensitive data exposure while preserving user privacy

(Continued)

Table 2 (continued)

Ref.	Objective	Metrics	Proposed/applied method	Outcomes
[31]	Privacy preservation intrusion detection (PPID) with correlation coefficient and expectation maximization(EM)	Detection rate, false positive rate	Correlation coefficient and EM clustering mechanisms	Select important portions of data and recognize intrusive events
[32]	Optimized federated learning for energy constrained devices	Energy consumption, model accuracy	FDMA	Reduced energy consumption while maintaining accuracy
[33]	Propose a privacy-preserving intrusion detection system based on federated learning for IoT networks.	Accuracy, privacy loss	Federated learning	Achieve high accuracy while preserving data privacy
[34]	Minimize global loss, balance local model updates	Accuracy, loss	FedAvg and FedProx	Proposed a non-parametric view of FedAvg and FedProx by finite-sum optimization
[35]	Improve patient data privacy	Accuracy	Auto-FedAvg	Proposed approach validated by current FL methods
[36]	Optimization of compute-communication and data importance-aware resource-monitoring systems, as well as evaluation of performance of training	Mean accuracy	FedAvg, FedMeta	Implemented models for individual clients
[37]	Heterogeneity in IoT gives chance to attackers	Accuracy measured for different epochs	PAG-FL	When training over a non-IID data set in an asynchronous FL, privacy is ensured
[38]	Reduce the training cost of models	Accuracy and loss	FedAwo optimization algorithm	Proposed the FedAwo optimization algorithm that combines adaptive learning with federated learning

(Continued)

Table 2 (continued)

Ref.	Objective	Metrics	Proposed/applied method	Outcomes
[39]	Users automatically alter the local iterations in every global interval and will not lose out due to lack of resources	Loss, and accuracy	FedProx	Reduced communication cost and fast convergence

4 Federated Learning Algorithm Specifications

The following are major advantages of federated learning algorithms:

i. Security and Privacy Data privacy and security are crucial features of FL, required for federated gradient updates and aggregation processes through encryption and other ways, and they can also be reflected in stand-alone optimizations.

ii. Collaboration Effectiveness The FL algorithms consider the data holders network diversity, enhance communication efficiency, and decrease communication impairment without sacrificing reliability.

iii. Support Nonindependent and Identically Decentralised Data This is a fundamental aspect of federated learning algorithms. The FL method must function effectively with non-independent and identically distributed data. In practice, the data quality and dispersion of the data are uncontrolled. The FL methods handles non-independent and identically distributed data because the data holders data cannot be compelled to fulfill independent and identical distribution.

iv. Support for Complex Users Complex users refer to many users and an unbalanced or variation in user data. This is achievable in the actual application of FL, and the federated optimization algorithm must have a good scalability impact in this circumstance.

v. Rapid Convergence In global modeling, achieving model convergence while increasing convergence speed.

4.1 FedAvg

FedAvg is an ML approach that allows training a model across a large number of decentralized devices without transferring the data to a central server. This method is helpful in the banking sector, where data privacy is crucial. Additionally, it allows multiple devices to develop a collective, reliable ML model without exchanging data. The following Algorithm 1 shows the working of the FedAvg:

Algorithm 1: FedAvg

1: **Input:** D, I_t, L_R, e, V_0, N, P_D, $D = 1, \ldots, N$
2: **for** $t = 0, \ldots, I_t - 1$ **do**
3: Server selects a subset S_t of D devices at random (each device D is chosen with probability P_D)
4: Server sends V_t to all chosen devices $\{k \in S_t\}$
5: Each device $D \in S_t$ updates V_t for e epochs of SGD on F_D with step-size L_R to obtain V_D^{t+1}

(Continued)

Algorithm 1 (continued)

6: Each device $D \in S_t$ sends V_D^{t+1} back to the server

7: Server aggregates the V's as $V_{t+1} = \frac{1}{D} \sum_{D \in S_t} V_D^{t+1}$

8: **end for**

The notations used in the algorithm are D is the total number of devices, I_t is the number of iterations, μ is the step size or learning rate, e is the number of epochs, and V_0 is the initial value of the optimization variable. N is the number of functions to be minimized. P_D is the probability of choosing device k. S_t: The subset of D devices chosen at random by the server. w_t is the current value of the optimization variable at iteration t. $F_D(V)$ is the function that device D can minimize. V_D^{t+1} is the updated value of the optimization variable for device k at iteration t+1, SGD is the stochastic gradient descent. μ is the step size or learning rate for SGD [40]. V_{t+1}: The updated value of the optimization variable at iteration t+1, which is the average of the updated values of the optimization variables of all the devices in S_t.

The algorithm works as follows: At each iteration t, the server selects a subset S_t of D devices randomly, with each device k chosen with probability P_D. The server then sends V_t to all chosen devices. Each chosen device $D \in S_t$ updates V_t for e epochs of SGD on F_D with step-size L_R to obtain V_D^{t+1}. Finally, each device $k \in S_t$ sends V_D^{t+1} back to the server, and the server aggregates the V values as $V_{t+1} = (1/D) * \sum_D \in S_t\ V_{t+1}^D$.

4.2 FedProx

FedProx is an extension of FedAvg, which can be used in distributed environments to increase the accuracy of models. It is designed to address the issues raised by heterogeneity in the system variables on each device correlated to the network. The Algorithm 2 specifies FedProx working.

Algorithm 2: FedProx

1: **Input:** D, I_t, μ, γ, w_0, N, P_D, $D = 1, \ldots, N$

2: **for** $t = 0, \ldots, I_t - 1$ **do**

3: Server selects a subset S_t of D devices at random (each device D is chosen with probability P_D)

4: Server sends V_t to all chosen devices. Each chosen device D $\in S_t$ finds a V_D^{t+1}

5: which is a γ_D^{t+1}-inexact minimizer of: $V_D^{t+1} \approx \arg min_w h_D(V; V_t) = F_D(V) + \frac{\mu}{2}||V - V_t D||^2$

6: Each device D $\in S_t$ sends V_{t+1}^D back to the server. Server aggregates the V values as

7: $V^{t+1} = \frac{1}{D} \sum_{D \in S_t} V_D^{t+1}$

8: **end for**

The notations used in the algorithm are D, the total number of devices. I_t is The number of iterations. μ is step size or learning rate. γ is the inexactness parameter. V_0 is the initial value of the optimization variable. N is the number of functions to be minimized. P_D is the probability of choosing device V. S_t is the subset of D devices chosen randomly by the server. V_t is the current value of the optimization variable at iteration t. $F_D(V)$: The function to be minimized by device D. γ_D^{t+1} is the inexactness parameter for device D at iteration t. V_{t+1}^D is the updated value of the optimization variable for device D at iteration $t + 1$. $hD(V; V_t)$ is the objective function for device D at iteration t. is the squared Euclidean distance between V and $V_t D$. $V_t + 1$ is the updated value of the optimization variable at iteration $t + 1$, which is the average of the updated values of the optimization variables of

all the devices in S_t. The algorithm works as follows: At each iteration t, the server randomly selects a subset St of D devices, with each D chosen with probability P_D. The server then sends V_t to all chosen devices. Each chosen device $D \in S_t$ finds a V_D^{t+1} which is a γ_D^{t+1} inexact minimizer of the function $h_D(V; V_t) = F_D(V) + \frac{\mu}{2}||V - V_t D||^2$. Finally, each device D $\in S_t$ sends V_{t+1}^D back to the server, and the server aggregates the V value as $V^{t+1} = \frac{1}{D} \sum D \in S_t V_D^{t+1}$.

$||V - V_t D||^2$

5 Proposed Methodology

To conduct the experiment on the proposed approach, adopted Flower, the new framework for FL developed by the University of Oxford. Flower is a Python 3 library for FL that is Deep Learning framework-agnostic. Training of statistical models may be done with any deep learning framework, such as TensorFlow or PyTorch, via a plugin mechanism. The quality of training data is a crucial factor in the accuracy and performance of machine learning models. Poor quality data can lead to inaccurate models and wasted time. In the case of financial transaction datasets, poor data quality can arise due to various reasons, such as missing data, incorrect data, or data that is not representative of the population. These issues can lead to biased models that do not generalize well to new data. Therefore, it is essential to ensure that the data is clean, complete, and representative of the population before using it to train ML models. Inaccurate, incomplete, or noisy data might negatively impact model performance on unseen data. This may result in erroneous predictions, decreased accuracy, and difficulties generalizing to new scenarios. It hinders FL algorithms from achieving optimum results. The training process may get trapped in suboptimal local optima or fluctuate indefinitely without finding a stable equilibrium. It may lead to privacy implications in FL. Increased noise or biases in the data may leak private information about individual participants from model updates sent during training. PCA (Principal Component Analysis) helps in this case. It is a technique used to reduce the dimensionality of large datasets while retaining as much of the original information as possible. It reduces the number of features by identifying the most informative ones, leading to a smaller and easier-to-manage dataset. This improves the efficiency of subsequent data processing and analysis, especially for tasks like visualization, clustering, and machine learning. Removing redundant or irrelevant features helps reduce overfitting in machine learning models. PCA focuses on capturing the underlying structure of the data, which can help to filter out random noise and outliers. This can lead to more accurate and reliable results in tasks like trend analysis and anomaly detection. PCA Improves Interpretability by identifying the principal components that explain most of the variance in the data. PCA can provide insights into the key features and relationships within the data. This can be helpful for understanding the data better and making informed decisions. PCA improves Computational Efficiency by reducing the dimensionality of the data, significantly improving computation time for various tasks.

All the experiments were computed in a distributed environment encompassing N = 10 collaborators. Each collaborator is run in a lab environment. Initially, the data collection is performed. The next step involves preprocessing operations with PCA to enhance data quality and make it more suitable for applying operations on datasets. After this, it performs the primary operation of the proposed work, which includes applying FedAvg and FedProx methods on the flower framework. The results are aggregated in this global model stage. The final stage involves the evaluation of performance using precision, recall, and accuracy measures. Fig. 2 represents the proposed framework. The operations can be done as follows:

i. Initialization of the global model: Initially, the centralized server RPC (Remote Procedure Call) activates the learning process by establishing the global model weights required for training, such as learning rates, momentum, and other values, FedAvg and FedProx. After startup, the server distributes the obtained model to the users specified in the first round.

ii. Training the models locally: The model is trained using local data on each client device. This training approach might include several iterations or epochs to improve the modelas performance. We assume clients that collaborate to train a global model for preserving privacy. The clients are responsible for preparing data locally using RPC, flower client python (FCP), training pipeline tensorflow (TPT), local data base (LDB).

iii. Model aggregation: After local training, the updated models from each client are delivered back to the central server (RPC).

iv. Averaging of model parameters: The server at the center collects the models received from clients by averaging the model parameters. This approach guarantees that the central global model gets the benefits of the information gained from multiple clients while maintaining privacy. These are evaluated based on performance parameters in the next phase.

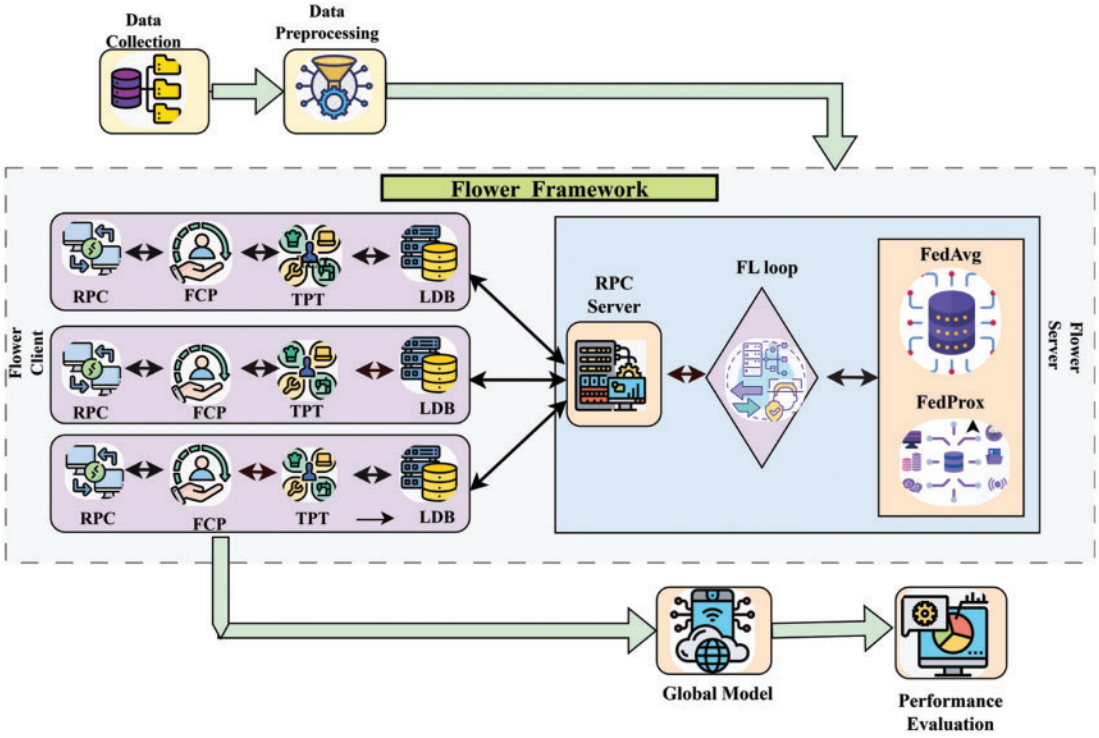

Figure 2: Proposed framework for preserving privacy in financial sectors

5.1 Performance Evaluation

Dataset Description: The dataset contains banking transaction details made by European cardholders in September 2013 [41]. This dataset contains 492 fraudulent activities out of 284,807 transactions that happened over the course of two days. It only has numeric input variables, which are the outcome of a PCA transformation. Another dataset considered is the CICIDS dataset [42],

which is representative of real-world data and includes the most recent and benign prevalent intrusions, according to information from the Canadian Institute for Cybersecurity. The dataset has more intricate characteristics and is accessible as a benchmark. There are a lot of traffic records and data in the form of values for a wide range of attributes in every record.

The proposed approach implemented with flwr, numpy, and tensorflow libraries was used to construct federated learning systems; tensorflow and numpy were utilized for ML and numerical computing. FL Client Implementation generates and configures clients for FL in the Flower framework. It accepts a client ID (CID) as input and returns a NumpyFlowerClient object. The function first creates a sequential model architecture for each client that includes many tightly linked layers and sigmoid activation functions. The number of output neurons corresponds to the total number of classes. Then, the given CID extracts the client's individual training data partition and labels. Finally, it generates and returns a NumpyFlowerClient object containing the client ID, trained model, local training data, and labels, allowing the client to participate in the FL process. By using this feature, the FL framework quickly establishes and maintains unique clients, each contributing to global model training with their local data and cooperatively developing common knowledge without compromising sensitive information. The Flower FL framework was implemented utilizing FedAvg and FedProx methods, and the same parameter values were considered for both algorithms. The classes flatten, dense, and activation from tensorflow.keras.layers. Sequential and flatten were used. One can generate a linear stack of layers by utilizing the Sequential class. The Flatten class is used to flatten the information that has been supplied. A fully connected layer was built using a dense class. The Activation class was used to apply an activation function to a layer's output. After every cycle, the evaluate function updates the model with the most current parameters. A sequential model with three dense layers, the *server_model* was created using the adam optimizer with sparse categorical cross-entropy loss. The Dense layers form a neural network with 256, 128, and 18 neurons in each layer. These layers used the sigmoid activation function to control output and decision-making. The results are compared with work presented in [43]. In this work, Zhang implemented a hybrid FL methodology with an accuracy of 99.05%. Current work improves the accuracy of the result with 2%.

Table 3 shows keen observation values obtained by using both FedAvg and FedProx. Metrics considered for performance analysis are precision, recall, and accuracy. Performance values for FedAvg were obtained as 99.38%, 99.64%, and 99.55%, and for FedProx as 99.58%, 99.73%, and 99.57%, respectively, for the Creditcard dataset. The CICIDS dataset obtained performance values as 94.51%, 83.32%, 83.72% for FedAvg, 94.62%, 84.45%, and 84.63% for Fedprox (all values considered in percentages). Figs. 3 and 4 give the graph visualization of performance metrics for both datasets, and the results demonstrate that FedProx is producing better results. Regarding effectiveness, FedAvg and FedProx successfully train the model in a distributed setting, achieving convergence and reasonable performance. Additionally, centralized training outperforms FedProx and FedAvg, but the difference is relatively small. This suggests that both are working effectively, but FedProx performance is a little higher than FedAvg.

Table 3: FedAvg and FedProx evaluation using credit card and CICIDS datasets

Metrics	Creditcard dataset		CICIDS dataset	
	FedAvg	FedProx	FedAvg	FedProx
Precision	99.38	99.58	94.51	94.62
Recall	99.64	99.73	83.32	84.45
Accuracy	99.55	99.57	83.72	84.63

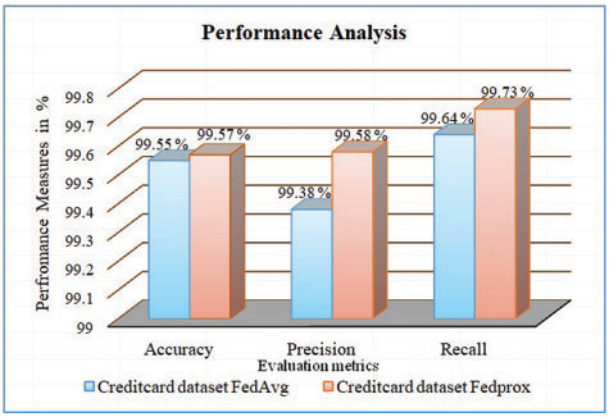

Figure 3: Performance analysis of creditcard dataset with proposed FL methods

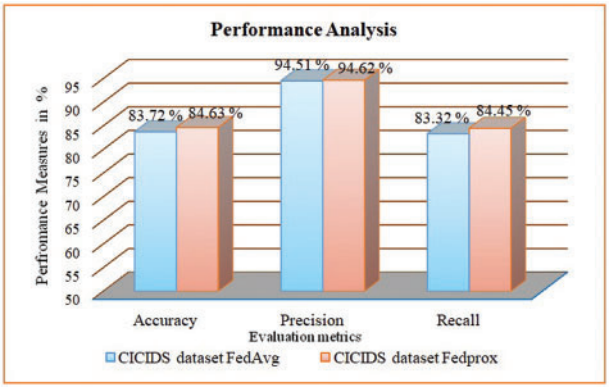

Figure 4: Performance analysis of CICIDS dataset with Proposed FL methods

6 Conclusion

The federated learning system is an efficient way of training a machine learning model that is more generic and widely applicable than one developed using data points from a single source. The conventional technique customizes the model to the organization's needs and provides the data. To protect sensitive data, current work presents a privacy-preserving FL technique based on FedAvg

and FedProx methods with the help of FlowerFL. It is effective in various FL scenarios, an effective breakthrough in bridging the gap between FL real-world systems and research. After empirical investigation on the FedAvg and FedProx algorithms with FlowerFL, the level of privacy between communication parties increased. Results show that FedProx and FedAvg both are effectively playing their roles in providing privacy with an accuracy of 99.55% and 99.57% for the credit card dataset and 83.72% and 84.63% for the CICIDS dataset, respectively. With the built-in extra functionalities, FedProx performs better than the FedAvg. Future works include comparing various FedAvg methods and server-side optimizations to enhance privacy in the financial sectors.

Acknowledgement: The authors are thankful to the anonymous reviewers for improving this article.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: The authors confirm their contribution to the paper as follows: study conception and design, data collection, analysis, and interpretation of results: Sandeep Dasari and Rajesh Kaluri, draft manuscript preparation, and editing: Sandeep Dasari, validation: Rajesh Kaluri. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: Based upon reasonable request, data can collect from the corresponding author.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Li Q, Wen Z, Wu Z, Hu S, Wang N, et al. A survey on federated learning systems: vision, hype and reality for data privacy and protection. IEEE Transacti. Knowledg. Data Eng. 2021;35(4):3347–66. doi:10.1109/TKDE.2021.3124599.
2. McMahan B, Moore E, Ramage D, Hampson S, Arcas BA. Communication-efficient learning of deep networks from decentralized data. In: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, 2017 Apr 10; Fort Lauderdale, USA, PMLR. p. 1273–82
3. Kim G, Kim J, Han B. Communication-efficient federated learning with acceleration of global momentum. arxiv preprint arxiv:2201.03172. 2022 Jan 10.
4. Li X, Huang K, Yang W, Wang S, Zhang Z. On the convergence of fedavg on non-IID data. arxiv preprint arxiv:1907.02189. 2019 Jul 4.
5. Reddi S, Charles Z, Zaheer M, Garrett Z, Rush K, Garrett Z, Konečný J, et al. Adaptive federated optimization. arxiv preprint arxiv:2003.00295. 2020 Feb 29.
6. Li T, Sahu AK, Zaheer M, Sanjabi M, Talwalkar A, et al. Federated optimization in heterogeneous networks. In: Proceedings of Machine Learning and Systems 2 (MLSys 2020), 2020 Mar 15. p. 429–50.
7. Dutta A, Doan TT, Reed JH. Resilient federated learning under byzantine attack in distributed nonconvex optimization with 2-f redundancy. In: 2023 62nd IEEE Conference on Decision and Control (CDC), 2023 Dec 13; IEEE.
8. Ntantiso L, Bagula A, Ajayi O, Kahenga-Ngongo F. A review of federated learning: algorithms, frameworks and applications. In: Towards new e-Infrastructure and e-Services for developing countries. Springer, Cham; 2022 Dec 5. p. 341–57.
9. Deng L. The mnist database of handwritten digit images for machine learning research [best of the web]. IEEE Signal Process Mag. 2012;29(6):141–2. doi:10.1109/MSP.2012.2211477.

10. Acar DAE, Zhao Y, Navarro RM, Mattina M, Whatmough PN, et al. Federated learning based on dynamic regularization. arxiv preprint arxiv:2111.04263. 2021 Nov 8.
11. Nilsson A, Smith S, Ulm G, Gustavsson E, Jirstrand M. A performance evaluation of federated learning algorithms. Proc Second Workshop Distrib Infrastruct for Deep Learn. Rennes, France; 2018 Dec 10; p. 1–8.
12. Xie C, Koyejo S, Gupta I. Asynchronous federated optimization. arxiv preprint arxiv:1903.0393. 2019 Mar 10.
13. Li M, Sahu AK, Zaheer M, Sanjabi M, Talwalkar A. Federated averaging: a new optimization method for distributed deep learning and its application to image classification. arxiv preprint arxiv:2107.06917. 2019, 2021 Jul. 1.
14. Sattler F, Wiedemann S, Müller KR, Samek W. Robust and communication-efficient federated learning from non-i.i.d. data. IEEE Trans Neural Netw Learn Syst. 2019;31(9):3400–13.
15. Briggs C, Fan Z, Andras P. Federated learning with hierarchical clustering of local updates to improve training on non-iid data. In: 2020 International Joint Conference on Neural Networks (IJCNN), 2020 Jul 19; Glasgow, UK. p. 1–9.
16. Somandepalli K, Qi H, Eoff B, Cowen A, Audhkhasi K, Belanich J, et al. Federated learning for affective computing tasks. In: 2022 10th International Conference on Affective Computing and Intelligent Interaction (ACII), 2022; Nara, Japan.
17. Reguieg H, El Hanjri M, El Kamili M, Kobbane A. A comparative evaluation of fedavg and per-fedavg algorithms for dirichlet distributed heterogeneous data. In: 2023 10th International Conference on Wireless Networks and Mobile Communications (WINCOM), 2023 Oct 26; Istanbul, Turkiye. p. 1–6.
18. Li Z, Sharma V, Mohanty SP. Preserving data privacy via federated learning: challenges and solutions. IEEE Consum Electron Mag. 2020;9(3):8–16. doi:10.1109/MCE.5962380.
19. Li X, Dowsley R, de Cock M. Privacy-preserving feature selection with secure multiparty computation. Int Conf Mach Learn. PMLR; 2021 Jul 1; p. 6326–36.
20. Collins L, Hassani H, Mokhtari A, Shakkottai S. Fedavg with fine tuning: Local updates lead to representation learning. Adv Neural Inf Process Syst. 2022;35:10572–86.
21. Xing S, Ning Z, Zhou J, Liao X, Xu J, Zou W. N-fedavg: Novel federated average algorithm based on fedavg. In: 2022 14th International Conference on Communication Software and Networks (ICCSN), 2022 Jun 10; Chongqing, China. p. 187–96.
22. Bonawitz K, Ivanov V, Kreuter B, Marcedone A, McMahan HB, et al. Practical secure aggregation for federated learning on user-held data. arxiv preprint arxiv:1611.04482. 2016.
23. Deng Y, Kamani MM, Mahdavi M. Adaptive personalized federated learning. arxiv preprint arxiv:2003.13461. 2020 Mar 30.
24. Liu W, Chen L, Chen Y, Zhang W. Accelerating federated learning via momentum gradient descent. IEEE TPDS. 2020;31(8):1754–66.
25. Qu L, Song S, Tsui CY. FedDQ: communication-efficient federated learning with descending quantization. In: GLOBECOM 2022–2022 IEEE Global Communications Conference, 2022 Dec 4; Rio de Janeiro, Brazil. p. 281–6.
26. Yang HH, Liu Z, Fu Y, Quek TQ, Poor HV. Federated stochastic gradient descent begets self-induced momentum. In: CASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2022 May 23; Singapore. p. 9027–31.
27. Tolomei G, Gabrielli E, Belli D, Miori V. A byzantine-resilient aggregation scheme for federated learning via matrix autoregression on client updates. arxiv preprint arxiv:2303.16668. 2023.
28. Jahani-Nezhad T, Maddah-Ali MA, Caire G. Byzantine-resistant secure aggregation for federated learning based on coded computing and vector commitment. arxiv preprint arxiv:2302. 2023 Feb.

29. Jothimurugesan E, Hsieh K, Wang J, Joshi G, Gibbons PB. Federated learning under distributed concept drift. In: International Conference on Artificial Intelligence and Statistics, 2023 Apr 11; Valencia, Spain, PMLR.
30. Shu X, Yao D, Bertino E. Privacy-preserving detection of sensitive data exposure. IEEE Trans Inf Forensics Secur. 2015;10(5):1092–103. doi:10.1109/TIFS.2015.2398363.
31. Keshk M, Moustafa N, Sitnikova E, Creech G. Privacy preservation intrusion detection technique for scada systems. In: 2017 Military Communications and Information Systems Conference (MilCIS), 2017 Nov 14; Canberra, Australia, IEEE. p. 1–6.
32. Yang Z, Chen M, Saad W, Hong CS, Shikh-Bahaei M. Energy efficient federated learning over wireless communication networks. IEEE Wirel Commun. 2020;20(3):1935–49.
33. Ruzafa-Alcázar P, Fernández-Saura P, Mármol-Campos E, González-Vidal A, Hernández-Ramos JL, Bernal-Bernabe J, et al. Intrusion detection based on privacy-preserving federated learning for the industrial IoT. IEEE Trans Ind Inf. 2021;19(2):1145–54.
34. Su L, Xu J, Yang P. A non-parametric view of FedAvg and FedProx: beyond stationary points. arxiv preprint arxiv:2106.15216. 2021 Jun 29.
35. Xia Y, Yang D, Li W, Myronenko A, Xu D, et al. Auto-FedAvg: learnable federated averaging for multi-institutional medical image segmentation. arxiv preprint arxiv:2104.10195. 2021 Apr 20.
36. Balakrishnan R, Akdeniz M, Dhakal S, Anand A, Himayat N. Resource management and model personalization for federated learning over wireless edge networks. J Sens Actuator Netw. 2021;10(1):17. doi:10.3390/jsan10010017.
37. Zhang T, Song A, Dong X, Shen Y, Ma J. Privacy-preserving asynchronous grouped federated learning for iot. IEEE Internet Things J. 2021;9(7):5511–23.
38. Yu X, Li L, He X, Chen S, Jiang L. Federated learning optimization algorithm for automatic weight optimal. Comput Intell Neurosci. 2022 Nov 9;2022.
39. He J, Guo S, Qiao D, Yi L. HeteFL: network-aware federated learning optimization in heterogeneous mec-enabled internet of things. IEEE Internet Things J. 2022;9(15):14073–86. doi:10.1109/JIOT.2022.3145360.
40. Ye M, Abbe E. Communication-computation efficient gradient coding. In: Proceedings of the 35th International Conference on Machine Learning, 2018 Jul 3; Stockholm, Sweden: PMLR. p. 5610–19.
41. Mlg-ulb. Credit card fraud detection dataset. Available from: https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud. [Accessed 2021].
42. Cicids2017. Intrusion detection dataset. Available from: https://www.kaggle.com/datasets/cicdataset/cicids2017. [Accessed 2020].
43. Zhang H, Hong J, Dong F, Drew S, Xue L, Zhou J. A privacy-preserving hybrid federated learning framework for financial crime detection. arxiv preprint arxiv:2302.03654. 2023 Feb 7.