DistLearn: Federated Learning in Distributed Systems

Abhinav Raundhal
2022101089
IIIT Hyderabad

Archisha Panda
2022111019
IIIT Hyderabad

Vinit Mehta
2022111001
IIIT Hyderabad

I. Introduction

Federated Learning (FL) is a distributed machine learning approach where multiple clients collaboratively train a global model without sharing raw data. However, FL faces challenges such as privacy risks, where model updates may still leak sensitive information, security threats, including Byzantine attacks and model poisoning and communication overhead as frequent model updates can strain network bandwidth. Despite these challenges, FL offers significant advantages like preserving data privacy by keeping data local, reducing data transfer costs, enabling collaborative learning across organizations without exposing confidential data, and supporting model personalization for individual devices. Overcoming these challenges is key to realizing the full potential of privacy-preserving, decentralized AI. Our goal is to develop a Distributed Federated Learning (DFL) system that overcomes these challenges and enables scalable, secure, and communication-efficient FL training.

II. System Architecture

Our DFL framework consists of the following components:

- **Central Server:** The central server stores and manages the global model and aggregates local model updates to ensure continuous improvement.
- **Clients:** Each client has its own private dataset which never leaves the device. Clients could be mobile devices, edge devices, or enterprise servers participating in federated learning.
- **Model Distribution:** The central server sends an initial copy of the global model to all selected clients which serves as a starting point for local training at each client.
- **Local Training:** Each client trains the model independently using its own dataset and updates the model weights based on local optimization. Techniques like gradient descent, adaptive learning rates, and momentum-based optimization can be used to improve training.
- **Model Update Transmission:** After local training, each client encrypts and transmits the updated model weights to the central server.
- **Global Aggregation:** The central server collects the updated weights from all clients and uses an aggregation strategy (e.g., Federated Averaging (FedAvg), adaptive weighting) to merge the updates into the global model. Aggregation techniques ensure that relevant updates are given more weight to improve model accuracy.
- **Model Convergence:** The updated global model is redistributed to clients for the next round of training. This iterative process continues until both local and global models converge to an optimal performance level.

III. Challenges and Proposed Solutions

Federated Learning (FL) enhances privacy by keeping data decentralized, but it faces challenges like high communication costs, data leakage, and malicious attacks. Frequent model updates increase network overhead, sometimes exceeding computational costs. Additionally, FL is vulnerable to threats like poisoning attacks, where corrupted data manipulates the model, and Byzantine attacks, where malicious participants disrupt training.

A. Privacy Protection in Federated Learning

When clients send updated model weights, an attacker could intercept and reverse-engineer them to infer sensitive data. To prevent this, we propose the following methods:

- **Differential Privacy (DP)**: Adds controlled noise to weight updates before transmission, making it difficult to extract original data. While this improves privacy, it can slightly reduce model performance as updates are less precise.
- **Homomorphic Encryption & Secure Aggregation**: Homomorphic Encryption (HE) and Secure Aggregation (SA) ensure privacy by encrypting model updates before sharing. HE allows computations on encrypted data, while SA enables aggregation without revealing individual contributions. **Implementation**: Clients encrypt updates using MPC (Shamir's Secret Sharing) or HE (Paillier, BFV). The server aggregates encrypted updates and decrypts only the final result. DP can add noise for extra privacy. Tools like PySyft, Microsoft SEAL, and TensorFlow Federated support these methods, ensuring secure and efficient federated learning.
- **Data Anonymization**: Removes personally identifiable information from datasets before training. Encoding identities as anonymous labels ensures that even if some data is leaked, it cannot be linked to individuals.

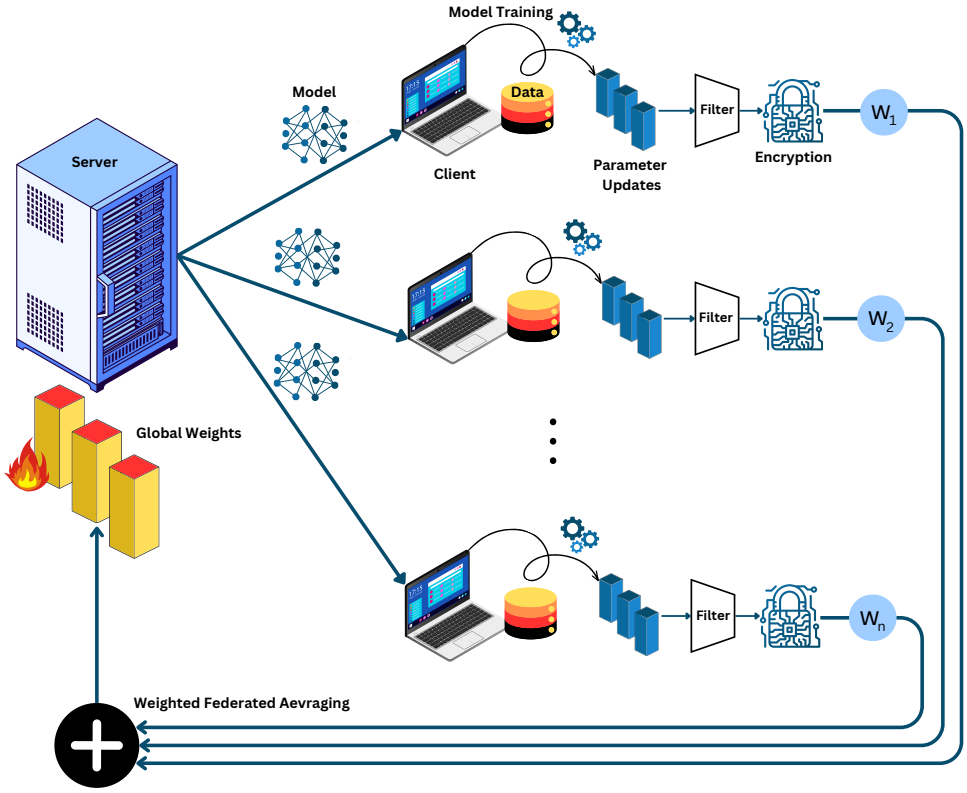

Fig. 1. Federated Learning Architecture: The figure illustrates a central server that coordinates training across multiple clients without accessing their local data. The server distributes an initial global model to clients, which train the model using their local datasets. The locally updated parameter weights are then passed through a filter and encryption process which includes KRUM, DP or HE to ensure privacy and security before being sent back to the server. The server aggregates these encrypted updates using a weighted federated averaging mechanism to update the global model weights. The process iterates until convergence, ensuring collaborative learning while preserving data privacy across distributed clients.

In our implementation, we will allow users to choose between using different configurations by declaring flags, depending on their security and performance needs.

B. Security Attacks and Mitigation Strategies

FL is vulnerable to poisoning attacks (where malicious clients send misleading updates) and Byzantine attacks (where compromised clients disrupt model training). To counter these, we propose:

- **Robust Aggregation (KRUM)**: Clustering weight updates based on distance metrics (such as Euclidean distance) helps detect outliers. Malicious or low-quality updates will deviate significantly from honest updates, allowing the system to filter them out before aggregation. This is a method of anomaly detection using clustering to identify poisoned model updates and remove them from the training process, ensuring the global model is not affected by adversarial participants.

These techniques enhance both privacy and security, making FL more resilient to attacks while maintaining model performance.

C. Communication Efficiency

Frequent transmission of model updates after every epoch leads to significant communication costs, sometimes exceeding the computational overhead. To optimize communication efficiency, we propose the following strategies:

- **Federated Averaging (FedAvg):** Instead of sending updates after every local iteration, clients perform multiple local training epochs before transmitting model updates to the central server. Another optimization could be introducing sparsification i.e sending only top-k significant gradients. These reduce the frequency of communication rounds, lowering bandwidth usage while maintaining convergence efficiency.
- **Client Selection:** Since not all clients contribute equally to the global model due to differences in computational power, network stability, and data quality, an effective client selection strategy ensures better model performance with minimal communication overhead. We prioritize clients based on the following factors:
  - **Network Stability:** Clients with stable and high-bandwidth network connections are preferred to minimize delays and transmission failures. This helps ensure timely updates and prevents bottlenecks caused by slow or unreliable connections.
  - **Data Quality and Diversity:** Clients with diverse and representative datasets improve the generalization of the global model. Selection can be based on

data distribution analysis to ensure that underrepresented data points are included in training.

– **Historical Contribution:** Clients that have previously provided valuable updates—such as those whose updates significantly improved model accuracy—can be given priority in selection. This ensures that the model benefits from high-quality contributions.

IV. Tech Stack

We will use the following technologies:

- **Programming Language:** Python
- **Frameworks:** PyTorch for deep learning
- **Communication:** gRPC for efficient communication

V. Datasets and Models

We will evaluate our framework using three datasets:

A. MNIST (Multi-class Classification - MLP/CNN)

MNIST is a widely used benchmark dataset for handwritten digit classification. We use it as a baseline to verify the correctness of our system before applying it to real-world data. Since it is a standard dataset, it helps ensure our framework functions as expected. We will use MLP and CNN models for classification.

B. Diabetes Dataset (Binary Classification - Logistic Regression/SVM/MLP)

This dataset represents a real-world medical application where hospitals have sensitive patient data but cannot share it due to privacy regulations like GDPR. Our framework enables training a global model across hospitals without sharing raw data. We will use Logistic Regression, SVM, and MLP to predict whether a patient has diabetes.

C. Taxi Trip Dataset (Regression - MLP)

Ride-sharing companies like Uber and Ola compete on pricing, and predicting optimal fares using federated learning can help them make better decisions without exposing private data. This dataset helps model real-world mobility trends using MLP for regression tasks like predicting trip duration or fare prices.

VI. Conclusion

We propose a Distributed Federated Learning (DFL) framework that enhances privacy, security, and communication efficiency. By integrating Differential Privacy, Homomorphic Encryption, and Secure Aggregation, we mitigate data leakage while preserving model performance. Robust aggregation methods like KRUM counter poisoning and Byzantine attacks, ensuring model integrity. Communication efficiency is improved through FedAvg, sparsification, and strategic client selection. We expect our distributed model to achieve accuracy comparable to a centralized model while reducing overall training time. Future work can explore using proxy data or smaller ”anchor” models to pre-screen clients, blockchain integration for decentralized trust management, and improving fairness in participation. Our proposed DFL system paves the way for scalable, privacy-preserving, and secure federated learning, enabling collaborative AI without compromising data confidentiality.