CSI5340 Project Proposal -

Federated Learning

GROUP 2 -
Mitchell *Chatterjee*
Ycaro *Dantas*
Alexandra *Sklokin*
Hedi *Bouali*

Topics of Discussion

1. Literary Review
    a. Introduction to Federated Learning
    b. Learning Schemes
    c. Privacy Mechanisms

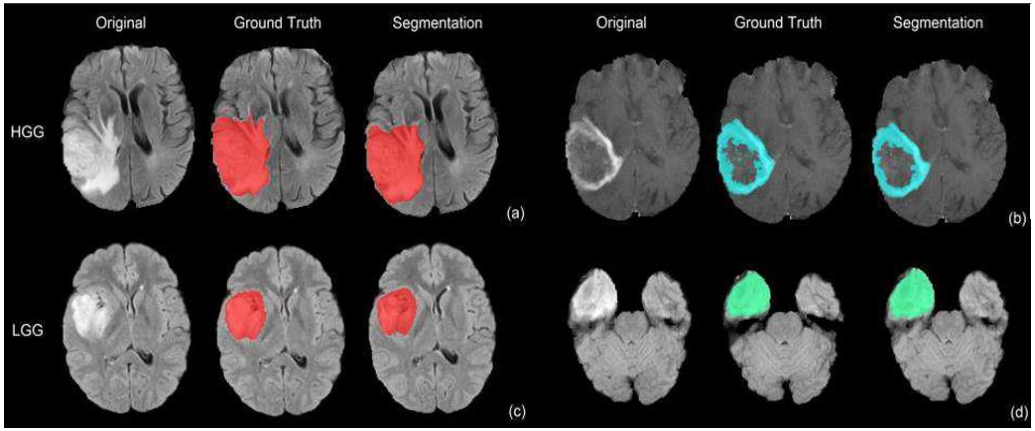
2. Project Proposal

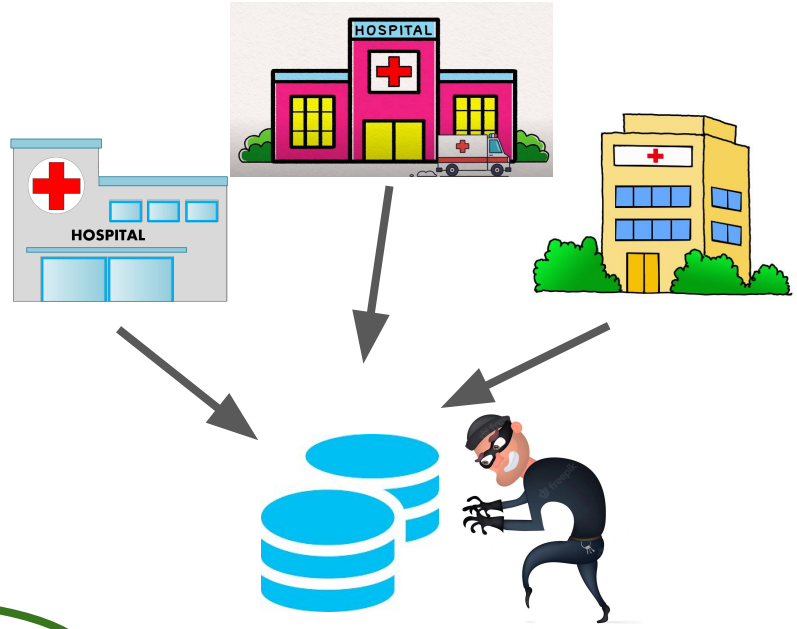
Literary Review -

Introduction to Federated Learning

Illustration of a practical application

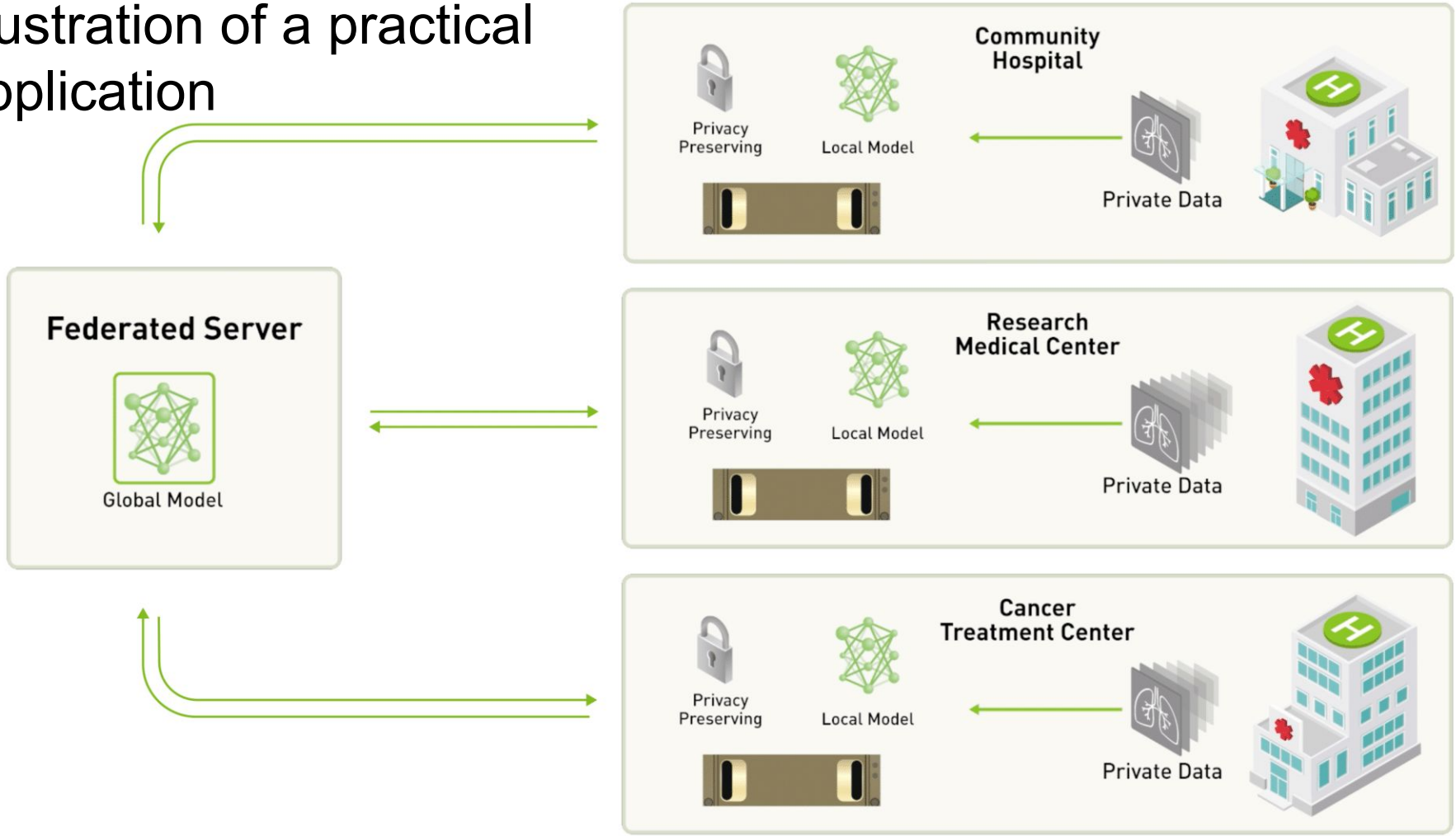
- Automatic classification of images of brain tumors





~~Centralized~~ → Federated

Illustration of a practical application



Taxonomy

3 Majors components in a federated learning systems :

- Parties (e.g., clients)
- Manager (e.g., server)
- Communication-computation framework

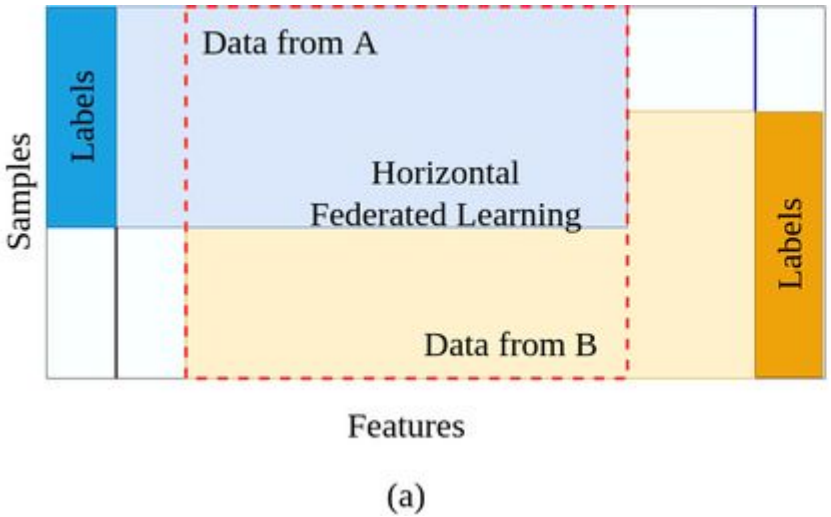
5 aspects :

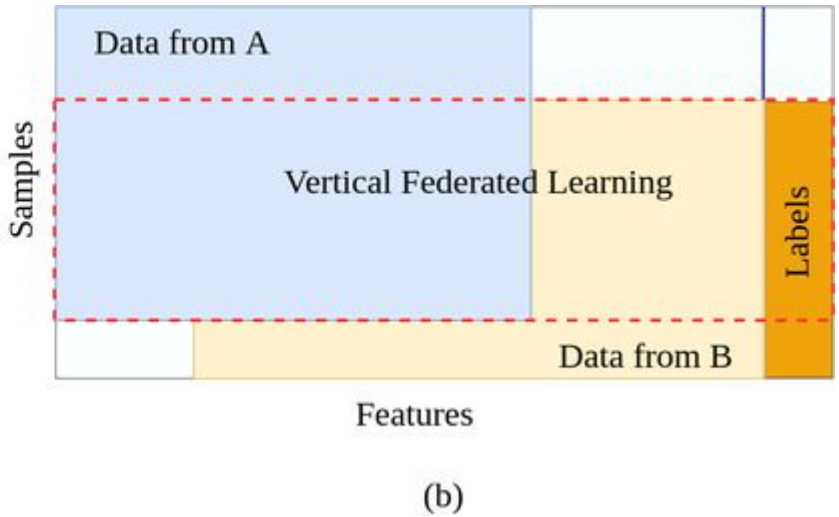
- data partitioning
  - Horizontal, vertical, and hybrid FLSs
- machine learning model
  - Neural network and decision tree
- privacy mechanism
  - Cryptographic methods and Differential privacy
- communication architecture
  - centralized design and decentralized design
- scale of federation
  - Cross-silo FLS and Cross-device FLS

Taxonomy

Data partitioning

- Horizontal, vertical, and hybrid FLSs

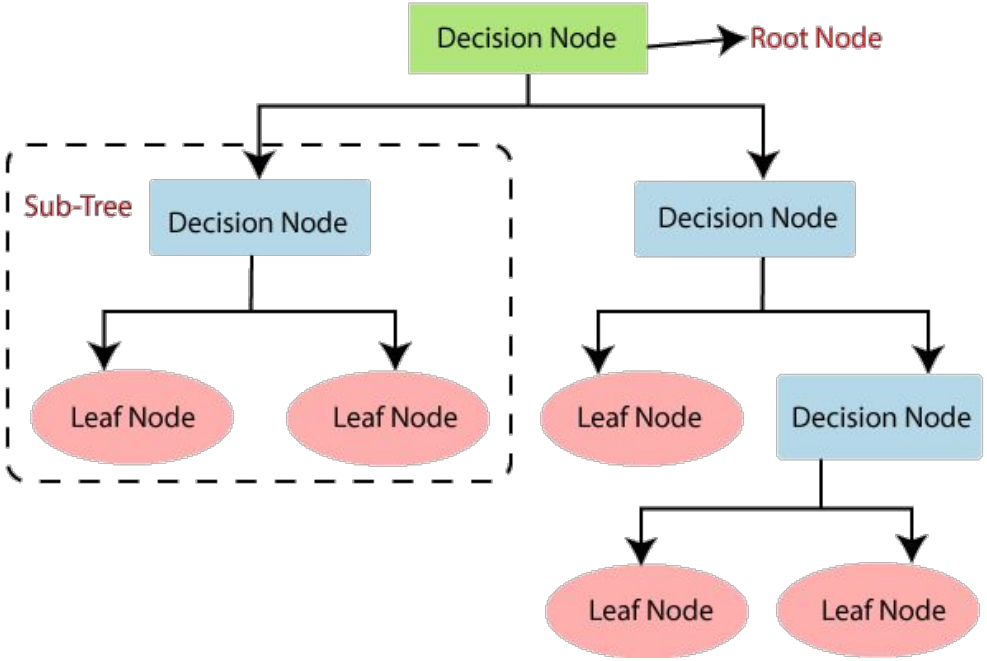




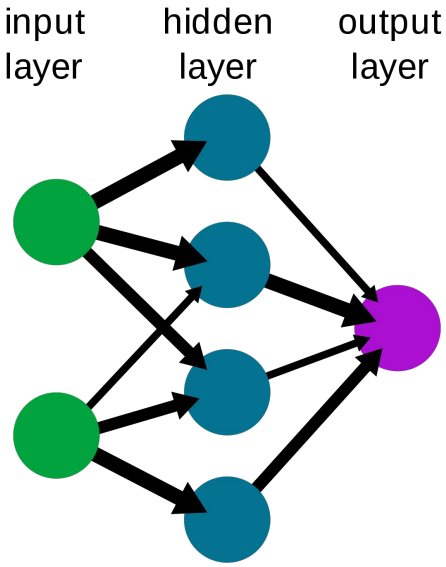
Taxonomy

Machine learning model

- Neural network and decision tree



A simple neural network

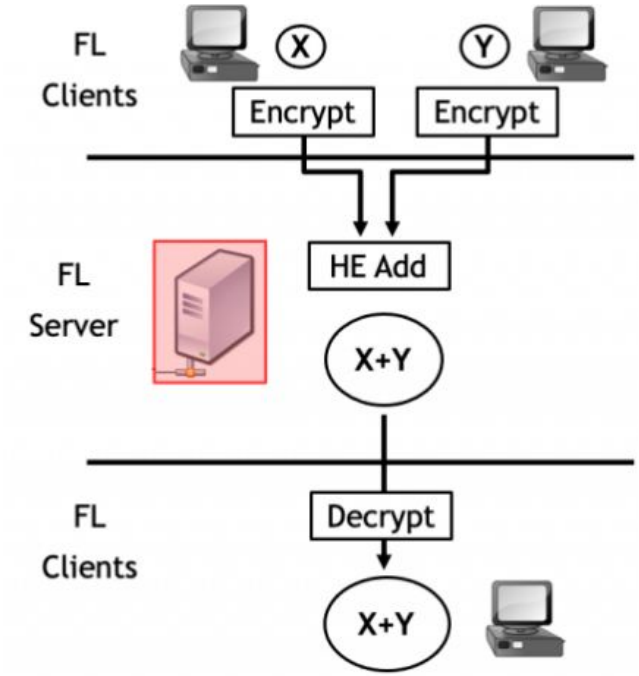


Taxonomy

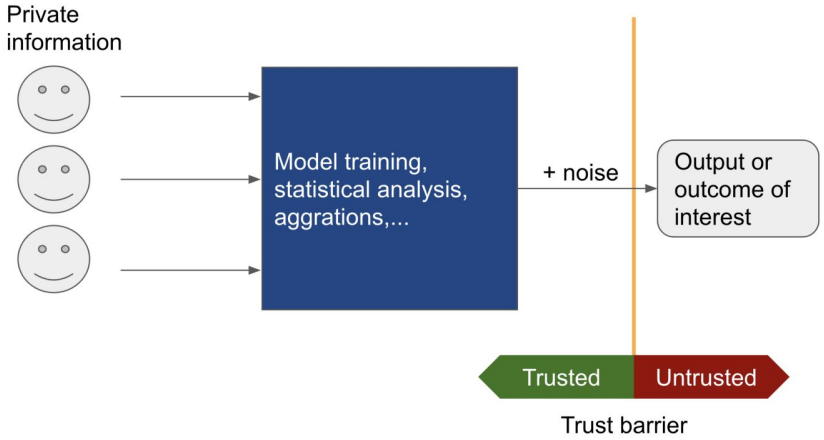
Privacy mechanism

- Cryptographic methods and

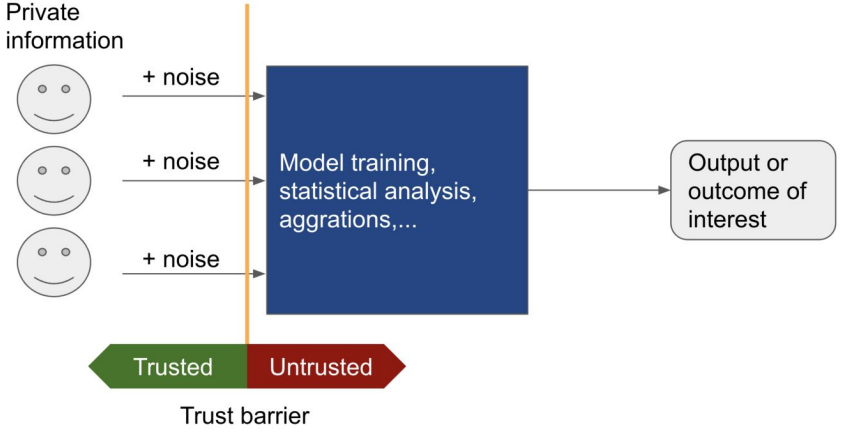
Differential privacy



Global differential privacy



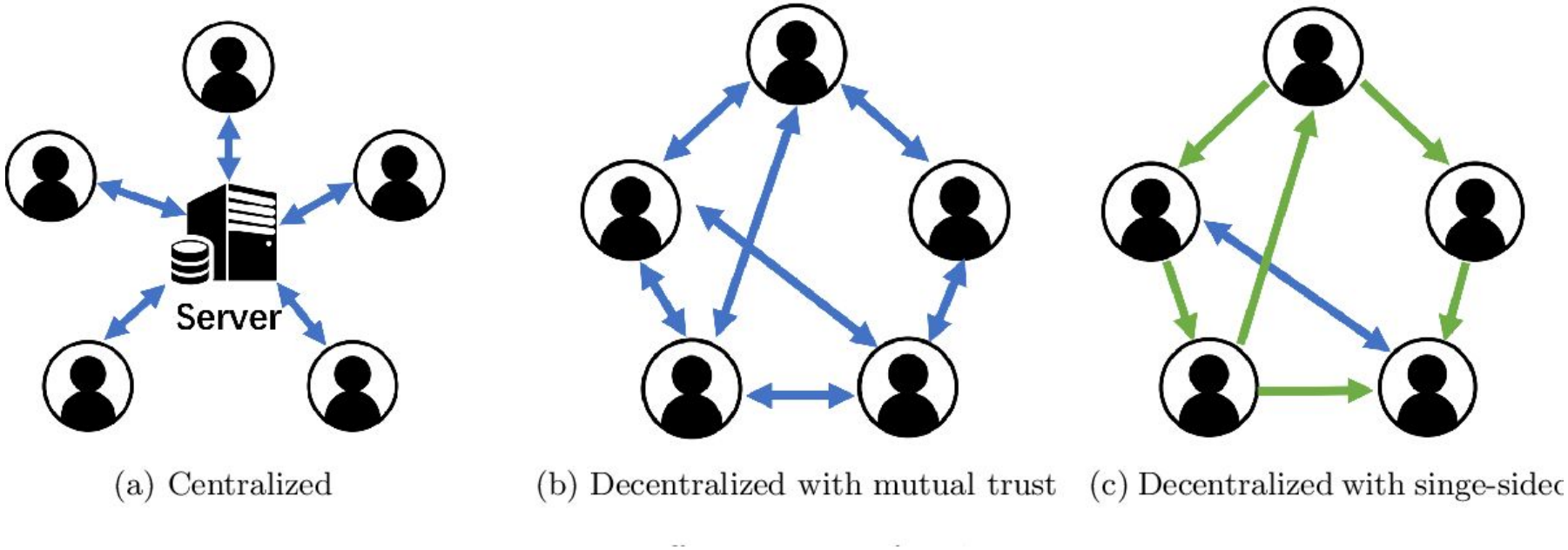
Local differential privacy



Taxonomy

Communication architecture

- Centralized design and decentralized design

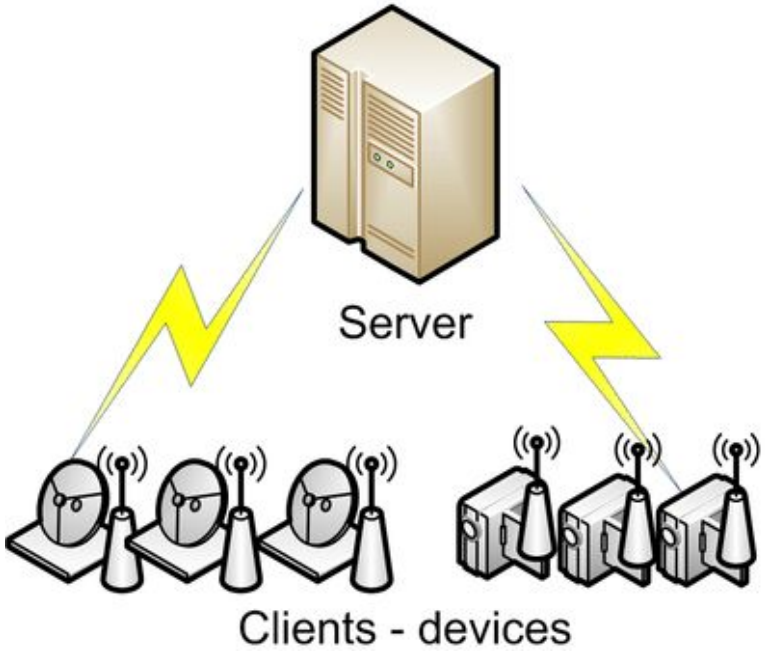


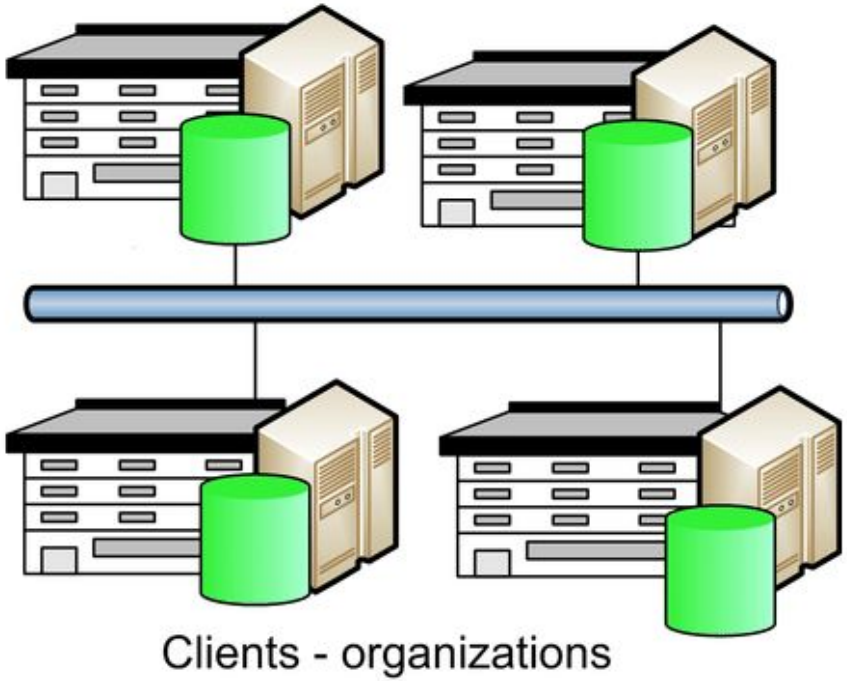
(a) Centralized (b) Decentralized with mutual trust (c) Decentralized with singe-sidec

Taxonomy

Scale of federation

- Cross-silo and Cross-device FLS





Literary Review -

Learning Schemes

FedAvg

- Federated Optimization introduces the following key properties:
  - **Non-IID data**: A local user's training data is not representative of the global dataset
  - **Unbalanced data**: Some users will have more local training data than others
  - **Massively distributed**: The number of clients is larger than the number of examples per client.
  - **Limited communication**: between devices

$$f(w) = \sum_{k=1}^{K} \frac{n_k}{n} F_k(w) \quad \text{where} \quad F_k(w) = \frac{1}{n_k} \sum_{i \in \mathcal{P}_k} f_i(w).$$

Communication-Efficient Learning of Deep Networks from Decentralized Data, McMahan et al.

FedAvg

- Key observation: Performing a series of epochs on a client prior to updating the global model significantly decreases the number of communication rounds and consequently increases the rate of convergence.

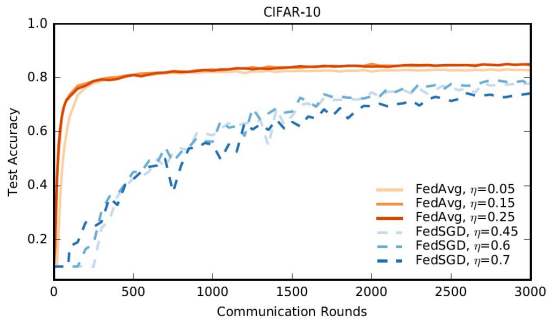
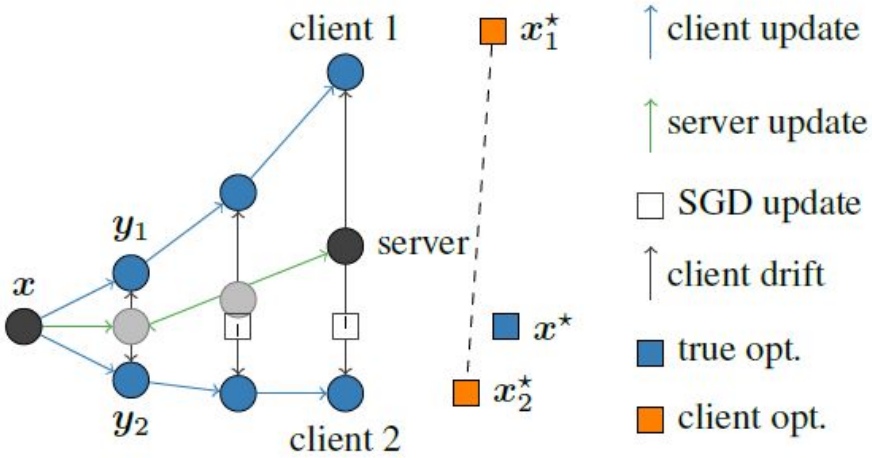


Figure 4: Test accuracy versus communication for the CIFAR10 experiments. `FedSGD` uses a learning-rate decay of 0.9934 per round; `FedAvg` uses $B = 50$, learning-rate decay of 0.99 per round, and $E = 5$.

Communication-Efficient Learning of Deep Networks from Decentralized Data, McMahan et al.

SCAFFOLD

- Heterogeneity in the local data introduces a drift in the updates of each client resulting in slow and unstable convergence of FedAvg



Stochastic Controlled Averaging for Federated Learning, Li et al.

SCAFFOLD

- Key observation: Introduce a control variate for server and client models.
  - This parameter quantifies the drift between the global and local models.
  - It then serves as a correction term.

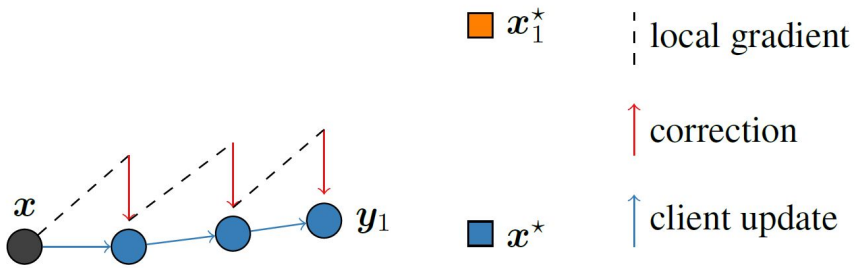


Figure 2. Update steps of SCAFFOLD on a single client. The local gradient (dashed black) points to $x_1^\star$ (orange square), but the correction term $(c - c_i)$ (in red) ensures the update moves towards the true optimum $x^\star$ (black square).

$$y_i \leftarrow y_i - \eta_l (g_i(y_i) + c - c_i) .$$

Local SGD

$$c_i^+ \leftarrow \begin{cases} \text{Option I.} & g_i(x) \text{ , or} \\ \text{Option II.} & c_i - c + \frac{1}{K\eta_l}(x - y_i) . \end{cases}$$

Local control variable update (Option II common)

$$x \leftarrow x + \frac{\eta_g}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} (y_i - x) ,$$

$$c \leftarrow c + \frac{1}{N} \sum_{i \in \mathcal{S}} (c_i^+ - c_i) .$$

Server variable updates

Stochastic Controlled Averaging for Federated Learning, Li et al.

FedProx

- Introduces a Proximal term to FedAvg that limits the impact of local updates.
  - Prefers updates that are closer to the current model.

$$\min_w h_k(w;\ w^t) = F_k(w) + \frac{\mu}{2}\|w - w^t\|^2.$$

- FedAvg then becomes a special case of FedProx with $\mu = 0$.

- Increasing the value of μ can force a divergent method to converge or increase stability.
  - Simple Heuristic: Increase μ when seeing a loss increasing and decrease μ when seeing the loss decreasing.

Federated Optimization for Heterogeneous Networks, Li et al.

FedProx

- Key observation: The authors notice that FedProx bounds the dissimilarity between the local and global models in each update. This allows them to theoretically and empirically prove convergence.
  - Can be enforced by setting μ appropriately.

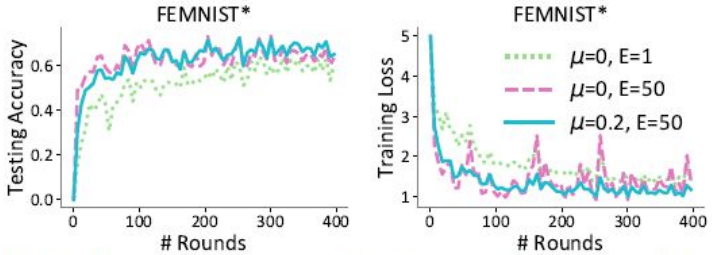


Figure 3. `FedProx` can provide faster and more stable convergence in communication-constraint environments (those requiring large E) with appropriate μ.

Federated Optimization for Heterogeneous Networks, Li et al.

MOON

- Guiding principle:
  - The global model trained on a whole dataset is able to learn a better representation than the local model trained on a skewed subset
- MOON adds a new loss term in order to bring the local models closer to the global model.

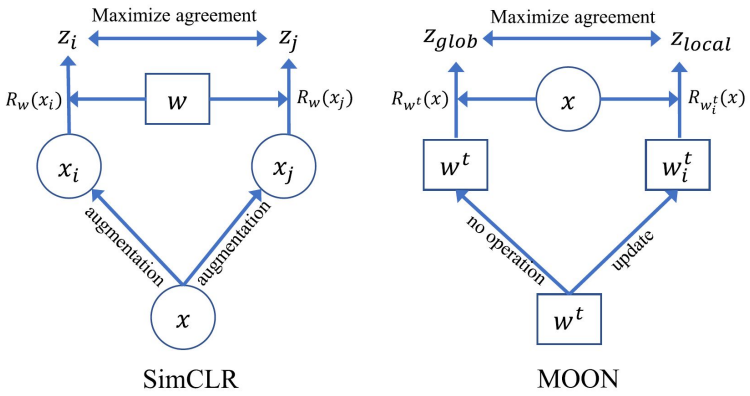
$$\ell_{con} = -\log \frac{\exp(\text{sim}(z, z_{glob})/\tau)}{\exp(\text{sim}(z, z_{glob})/\tau) + \exp(\text{sim}(z, z_{prev})/\tau)} \tag{3}$$

$$\ell = \ell_{sup}(w_i^t; (x, y)) + \mu \ell_{con}(w_i^t; w_i^{t-1}; w^t; x),$$

Model-Contrastive Federated Learning, Li et al.

MOON

- Key Observation:
  - Decrease the distance between the representation learned by the local model and the representation learned by the global model
  - Increase the distance between the representation learned by the local model and the representation learned by the previous local model



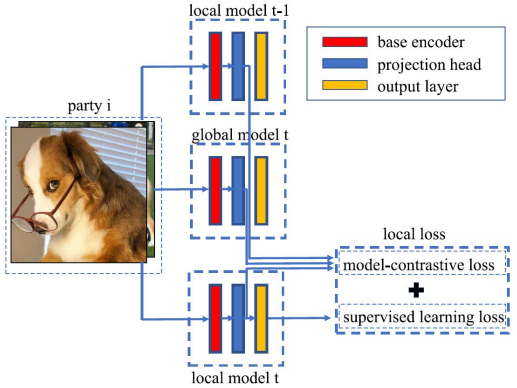


Figure 3. The local loss in MOON.

Model-Contrastive Federated Learning, Li et al.

FedMA

- Constructs the shared global model in a layer-wise manner by matching and averaging hidden elements with similar feature extraction signatures.
  - Channels for CNNs, hidden states for LSTMs and neurons for FC layers
- Permutation invariance of fully connected architectures

$$\hat{y} = \sigma\left(xW_1 \right) W_2$$

$$\hat{y} = \sigma\left(xW_1 \Pi \right) \Pi^T W_2, \text{ where } \Pi \text{ is an } L \times L \text{ permutation matrix}$$

- Accomplished through maximum bipartite matching

$$\min_{\{\pi_{li}^j\}} \sum_{i=1}^{L} \sum_{j,l} \min_{\theta_i} \pi_{li}^j c(w_{jl}, \theta_i) \text{ s.t. } \sum_i \pi_{li}^j = 1 \; \forall \; j, l; \; \sum_l \pi_{li}^j = 1 \; \forall \; i, j.$$

Federated Learning with Matched Averaging, Wang et al.

FedMA

- First data center gathers only the weights of the first layers from the clients and performs one-layer matching.
- Data center the broadcasts these weights to clients, which proceed to train all *consecutive* layers on their datasets, keeping the matched federated layers *frozen*.
- This procedure is then repeated up to the last layer.
- The last layer conducts a weighted averaging based on the class proportions of data points per client.

Federated Learning with Matched Averaging, Wang et al.

FedMA

- Key observation: The authors observe that training longer benefits FedMA.
- In contrast, longer training on FedAvg and FedProx causes the global model to diverge and eventually perform far worse (McMahan et al., 2017)

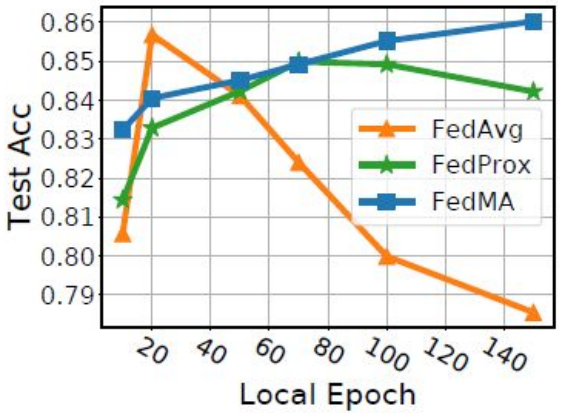
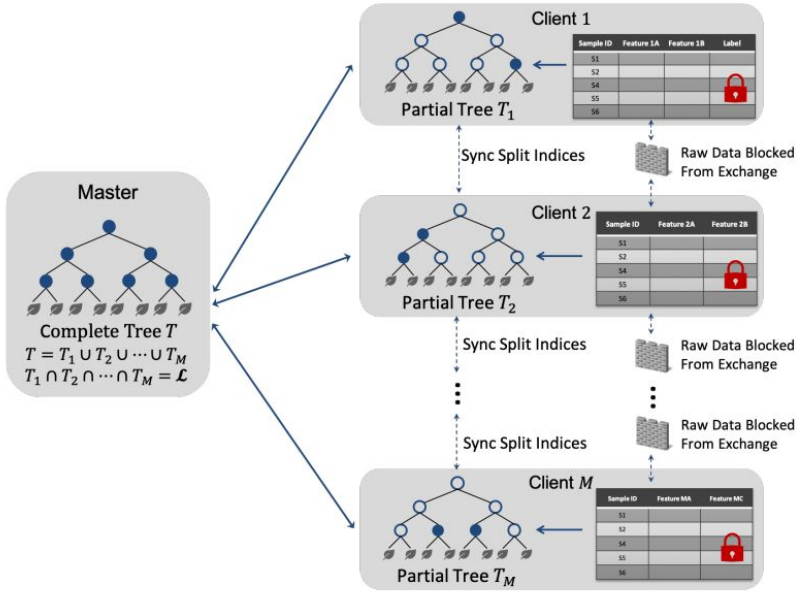


Figure 3: The effect of number of local training epochs on various methods.

Federated Learning with Matched Averaging, Wang et al.

Federated Forest

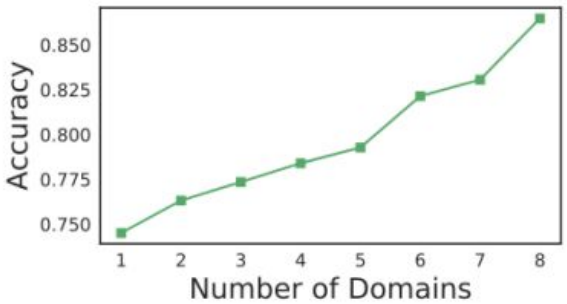
- Master
  - (1) Master uses bagging to select a subset of the features (from all data). Notify client about features and sample IDs. Perform tree pruning.
  - (4) Master chooses from local feature splits. Notify client with best split.
  - (6) Create nodes of master tree. Repeat until stopping condition. Distribute tree to clients.
- Client
  - (2) If stopping conditions are met, leaf nodes are created.
  - (3) Otherwise, SPLITTING PHASE. Each client finds optimal split feature.
  - (5) Best client sends sample ID split (left and right) back to master.

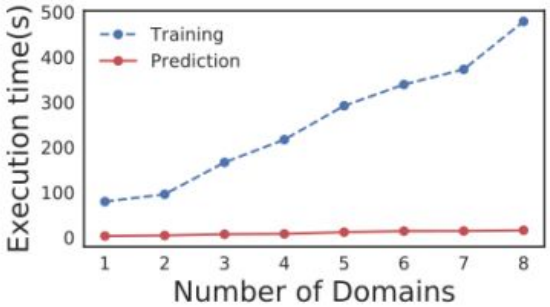


Federated Forest, Liu et al.

Federated Forest

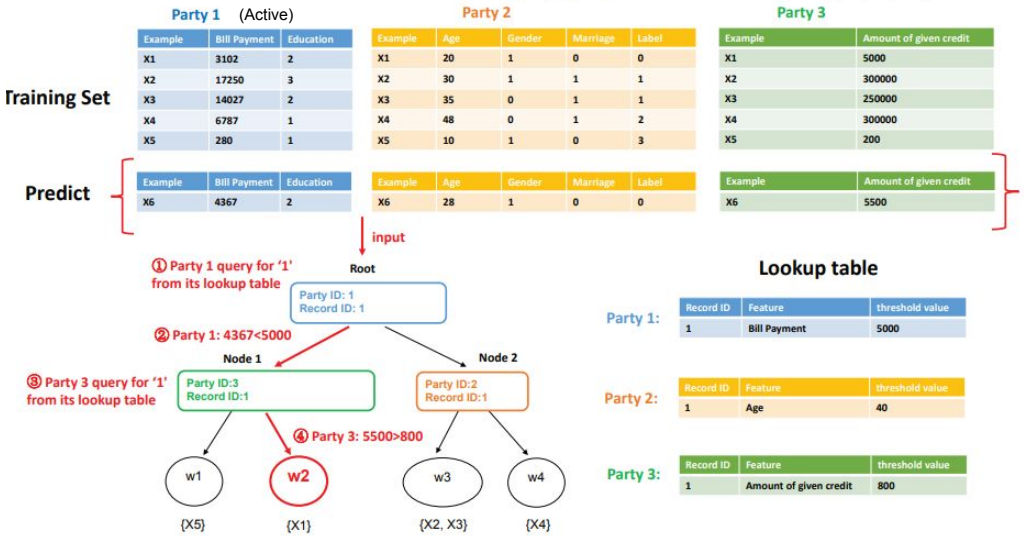
- Predictions
  - One communication
    - Client: Use local model to predict. Send predictions to master.
    - Master: Generate each global prediction by intersecting all local predictions on features.
- Key Observations:
  - More domains (features + client) contributes to better predictions. Training time increases, but prediction time remains constant.





Federated Forest, Liu et al.

SecureBoost

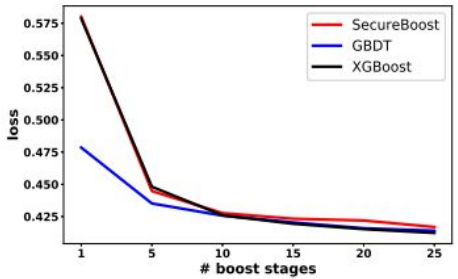


- Based on XGBoost
- Training:
  - Each client determines best feature split. Send this split using additive homomorphic encryption scheme. Each client keeps a tree.
  - Each tree node keep record of client to use for following split criteria.
  - Parties keep lookup table with spit thresholders from other clients.
- Predictions:
  - Active party (1) receives data. Uses lookup table to make predictions on unseen features.

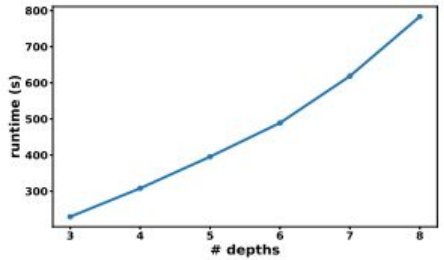
SecureBoost: A Lossless Federated Learning Framework, Cheng et al.

SecureBoost

- Key Observation:
  - (a) Similar performance to non-federated methods. Slight overperformance on testing data.
  - (b) Runtime increases linearly with the maximum depth of each client's tree.
  - Framework works relatively well of large datasets.



(a) Learning Curve

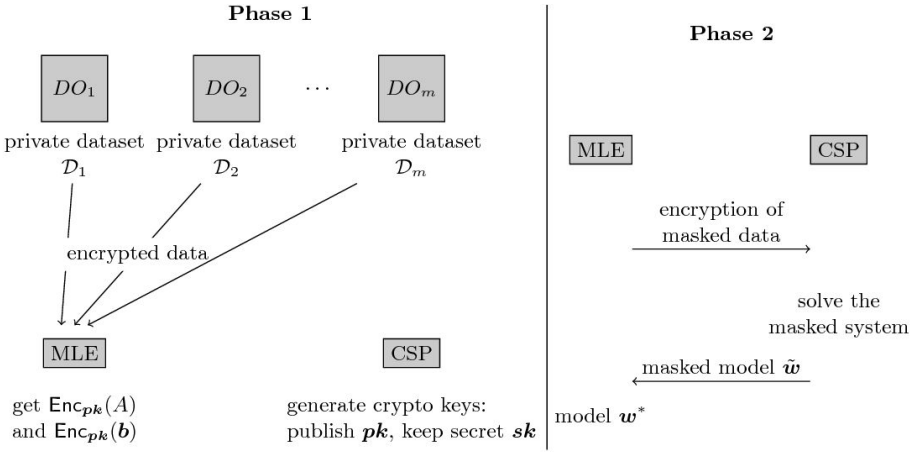


(b) Runtime w.r.t. maximum depth of individual tree

SecureBoost: A Lossless Federated Learning Framework, Cheng et al.

Privacy-Preserving Ridge Regression

- Privacy-Preserving Linear Model
- Phase 1:
  - Each client encrypts their data and sends it to centralized server, along with mechanisms for decryption.
- Phase 2:
  - Centralized server performs the linear regression/classification. Returns encrypted model parameters to clients.
- Key Observation:
  - Faster than other state-of-the-art Federated linear model approaches.



Privacy-Preserving Ridge Regression on Hundreds of Millions of Records, Nikolaenko et al.

Literary Review -

Privacy Mechanisms

Differential Privacy

User data might be reconstructed based on the gradients during the training;

Additional privacy mechanisms are required to secure information in adversarial attacks;

Noise before Aggregation Federate Learning (NbAFL) is a differential privacy mechanism based on the addition of gaussian noise to the neural network parameters.

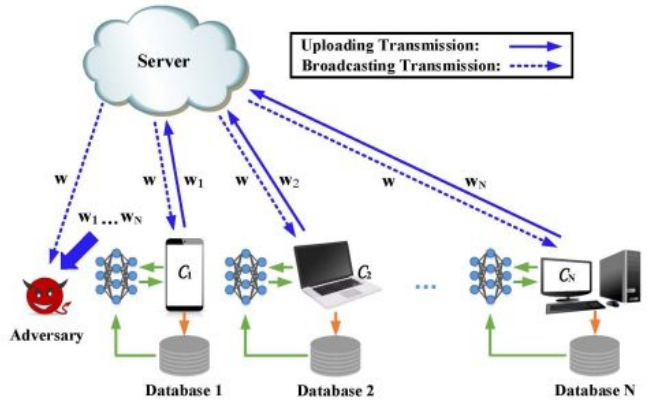


Fig. 1. A FL training model with hidden adversaries who can eavesdrop trained parameters from both the clients and the server.

Federated Learning With Differential Privacy: Algorithms and Performance Analysis, Wei et al.

Differential Privacy

The performance of the model and its aggregation time are directly affected by the noise figures added to enhance the privacy.

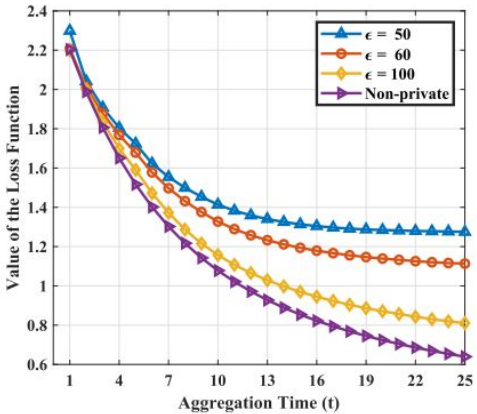


Fig. 2. The comparison of training loss with various protection levels for 50 clients using $\epsilon = 50$, $\epsilon = 60$ and $\epsilon = 100$, respectively.

Algorithm 1 Noising Before Aggregation FL

Data: T, $\mathbf{w}^{(0)}$, μ, ϵ and δ

Initialization: $t = 1$ and $\mathbf{w}_i^{(0)} = \mathbf{w}^{(0)}$, $\forall i$

while $t \leq T$ **do**

  Local training process:

  while $\mathcal{C}_i \in \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_N\}$ **do**

    Update the local parameters $\mathbf{w}_i^{(t)}$ as

    $$\mathbf{w}_i^{(t)} = \arg\min_{\mathbf{w}_i} \left(F_i(\mathbf{w}_i) + \frac{\mu}{2}\|\mathbf{w}_i - \mathbf{w}^{(t-1)}\|^2\right)$$

    Clip the local parameters

    $$\mathbf{w}_i^{(t)} = \mathbf{w}_i^{(t)} / \max\left(1, \frac{\|\mathbf{w}_i^{(t)}\|}{C}\right)$$

    Add noise and upload parameters $\widetilde{\mathbf{w}}_i^{(t)} = \mathbf{w}_i^{(t)} + \mathbf{n}_i^{(t)}$

  Model aggregating process:

  Update the global parameters $\mathbf{w}^{(t)}$ as

  $$\mathbf{w}^{(t)} = \sum_{i=1}^{N} p_i \widetilde{\mathbf{w}}_i^{(t)}$$

  The server broadcasts global noised parameters

  $$\widetilde{\mathbf{w}}^{(t)} = \mathbf{w}^{(t)} + \mathbf{n}_{\mathrm{D}}^{(t)}$$

  Local testing process:

  while $\mathcal{C}_i \in \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_N\}$ **do**

    Test the aggregating parameters $\widetilde{\mathbf{w}}^{(t)}$ using local dataset

  $t \leftarrow t + 1$

Result: $\widetilde{\mathbf{w}}^{(T)}$

Federated Learning With Differential Privacy: Algorithms and Performance Analysis, Wei et al.

Cryptographic Methods - Homomorphic Encryption

Cryptographic key mechanisms might be used to ensure the safe communication of the gradients between clients and server;

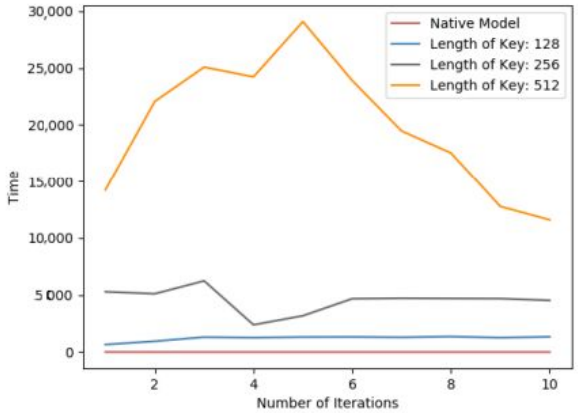


Figure 8. Per-iterations time spent on the MNIST dataset for different key lengths.

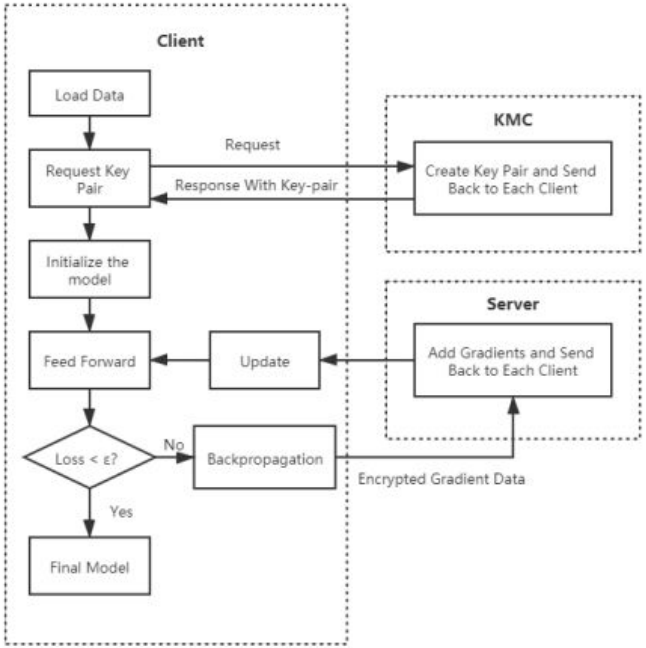


Figure 4. The architecture of a Paillier federated network.

Privacy Preserving Machine Learning with Homomorphic Encryption and Federated Learning, Fang et al.

Project Proposal

Problem Statement and Motivation

Different deployment options, learning schemes, and privacy mechanisms may be explored inside Federated Learning System;

The selection of a federated learning system configuration may vary according to the application requirements and the **impacts on model's performance introduced by each solution**.

What are the trade-offs involved?

Research Proposal

Considering an image classification task, evaluate the performance of Federated Learning Systems under multiple configuration options:

1. Assess the impact of each configuration change on each evaluation metric;
2. Provide a framework to define the FLS configuration based on the application's requirements.

Configuration Options:

- Learning Scheme
- Privacy Mechanism
- Number of Nodes
- Data Heterogeneity

Evaluation Metrics:

- Accuracy
- Training Time
- Computational Cost
- Communication Overhead

https://github.com/OpenMined/PySyft

Tools for Implementation

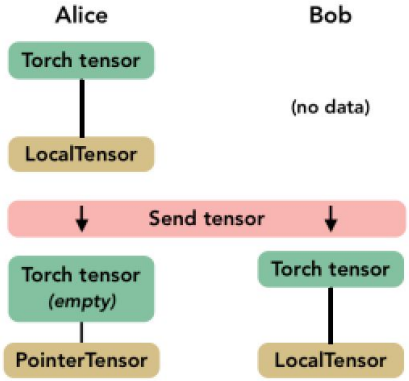
- Developed by OpenMined
  - Windows, Mac, and Linus
- Models:
  - Neural Networks and Linear Models
- Privacy Mechanisms:
  - Secure Multi-party computation: cryptographic protocol distributes computation across data centers such that no party can see other's data
  - Differential privacy: protecting privacy of individual records by only sharing patterns of data centers and injecting random noise into data and model parameters. Save from reverse engineering attacks.
- Communication:
  - Single (simulated) or multiple machines (distributed)
  - Websocket API

A generic framework for privacy preserving deep learning, Ryffel et al.

https://github.com/OpenMined/PySyft

PySyft

- SyftTensor: chain structure
  - Torch tensor: head of each SyftTensor is a Python tensor
  - LocalTensor: created automatically for every instance of SyftTensor
  - PointerTensor: created when sending tensor to remote worker
- Training Time:
  - Reasonable increase from simulated training to Websocket
  - Reasonable increase with added privacy mechanisms
  - Significant overhead from non-federated Pytorch training



Training mode	Training time (s)
PySyft (Virtual)	10.1
PySyft (Socket)	14.6
PySyft (Virtual) + DP*	15.3
Pure PyTorch	0.22

A generic framework for privacy preserving deep learning, Ryffel et al.

TensorFlow TFF

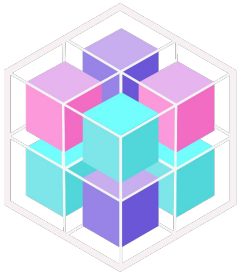
https://github.com/tensorflow/federated

FATE

https://github.com/FederatedAI/FATE

FedML

https://github.com/FedML-AI/FedML



PFL

Paddle Federated Learning

https://github.com/PaddlePaddle/PaddleFL

Project Plan

Main Tasks

- Data Setup (1 group member):
  - Data download and pre-processing; Setup the data holders (multiple number of nodes);
  - Setup the data distributions (IID, non-IID); Setup baseline centralized model.
- Implementation and evaluation of learning schemes (3 group members):
  - FedAvg, FedMA and a tree-based FL scheme.

Risk Factors

- Library (PySyft) manipulation and learning schemes implementation;
- Computation cost (long time to get results from multiple nodes).

Project Plan

Timeline and Deliverables

Oct 06-12: Theme Selection and Survey Reading

Oct 13-26: Literature Review

Oct 27-Nov 3: Project Proposal Preparation

Nov 3: Project Proposal Presentation

Nov 4-10: Data Setup and Library Exploration

Nov 10-17: Learning Schemes Implementation

TBD (Nov 14-18): Project Check-in

Nov 18-24: Results Collection and Discussions

Nov 25-Dec 1: Final Project Preparation

TBD (Nov 5-9): Final Project Presentation