Provably Personalized and Robust Federated Learning

Emerging Topics in Machine Learning

Titouan Borderies, Darius Dabert, Maxime Basse

Ecole Polytechnique

November 13, 2024

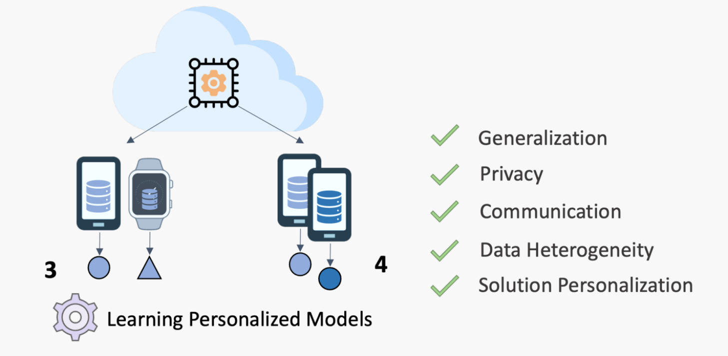
Agenda

ÉCOLE POLYTECHNIQUE

Challenges in Federated Learning

- The general Federated Learning (FL) approach encounters several fundamental challenges:
  1. **Poor Convergence on Highly Heterogeneous Data:**
     - The diversity in data distributions among clients can lead to suboptimal convergence.
  2. **Lack of Solution Personalization:**
     - FL may struggle to provide personalized solutions for individual clients.
  3. **Exposure to Byzantine attacks**

Personalized Federated Learning



[1]

[1] Source: Towards Personalized Federated Learning, Alysa Ziying Tan, IEEE

Modelling assumptions

- Clients belong to K groups that have distinct data distributions.
- The gradients from models of the same group form clusters in the gradient space.
- Gradient clusters are clearly separated between groups
- Objectives:
  1. Automatically identifying clusters of gradients at each iteration.
  2. Must be Byzantine Robust.
  3. Train one personalized model for each client

Notations

- N denotes the number of clients
- K is the number of cluster (hyperparameter)

Hypothesis

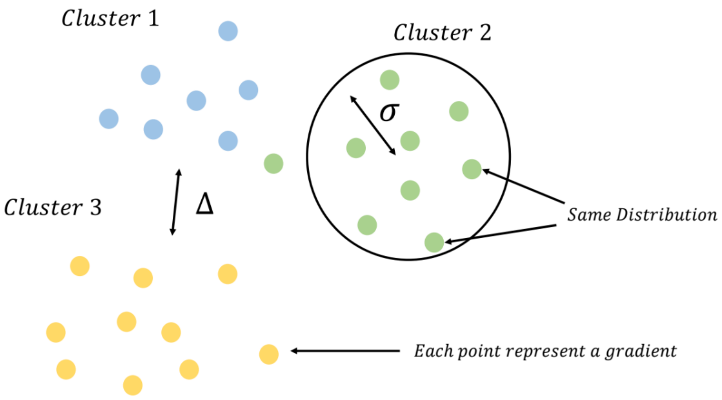
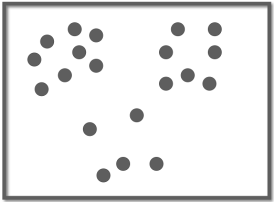


Figure: Gradients distribution

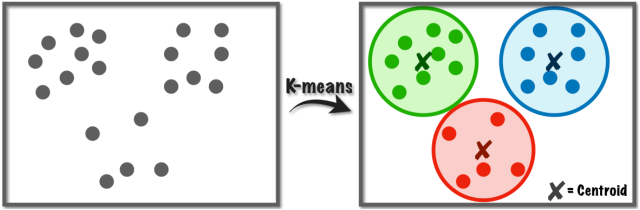
- Δ denotes the inter-cluster separation
- σ denotes the intra-cluster variance

What would you do ?



At every train step, cluster the gradient of every client models and send the cluster center to each client.

What would you do ?



At every train step, cluster the gradient of every client models and send the cluster center to each client.

- Intuitive Idea
- Communication-efficient : $\mathcal{O}(N)$

Algorithm 1: Myopic-Clustering

Input: Learning rate: η. Initial parameters: $\{x_{1,0} = \ldots = x_{N,0} = x_0\}$.

for *round* $t \in [T]$ **do**
    for *client* i *in* $[N]$ **do**
        Client i sends $g_i(x_{i,t-1})$ to **server**;
    Server clusters $\{g_i(x_{i,t-1})\}_{i\in[N]}$, generating cluster centers $\{v_{k,t}\}_{k\in[K]}$;
    for *client* i *in* $[N]$ **do**
        Server sends $v_{k_i,t}$ to client i, where k_i denotes the cluster to which client i is assigned;
        Client i computes update: $x_{i,t} = x_{i,t-1} - \eta v_{k_i,t}$;

Output: Personalized parameters: $\{x_{1,T}, \ldots, x_{N,T}\}$.

The **limits** of this naive approach:

- k-means is not Byzantine robust.
- Doesn't work well in practice because clients from different clusters can be trained in the wrong group of clients if the clustering fails at one step of the algorithm.

Intuition behind Threshold Clustering

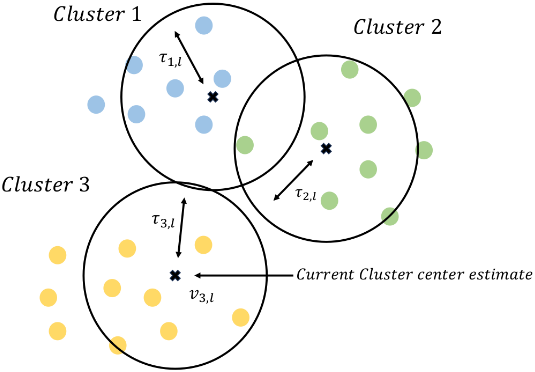


Figure: Threshold Clustering

Idea: Group data points that are close to each other within a certain threshold.

Intuition behind Threshold Clustering

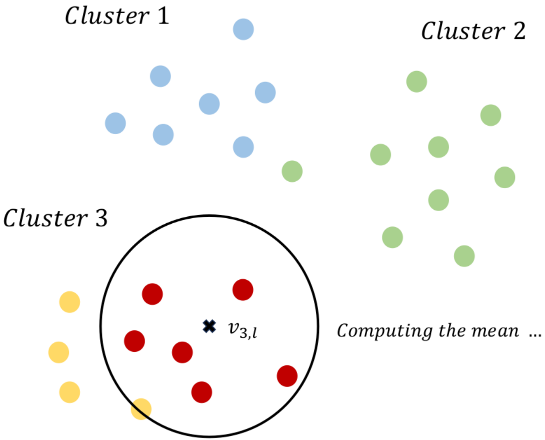


Figure: Threshold Clustering

Intuition behind Threshold Clustering

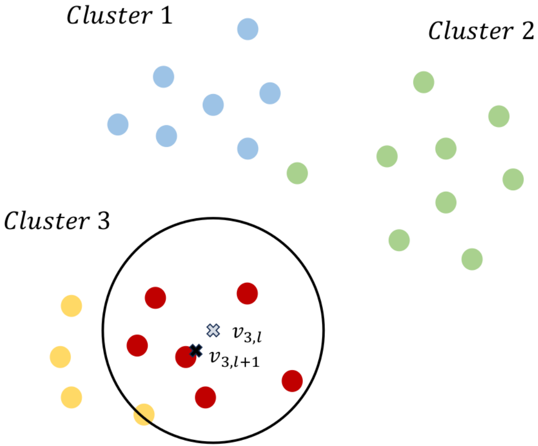


Figure: Threshold Clustering

Byzantine Robustness of Threshold Clustering

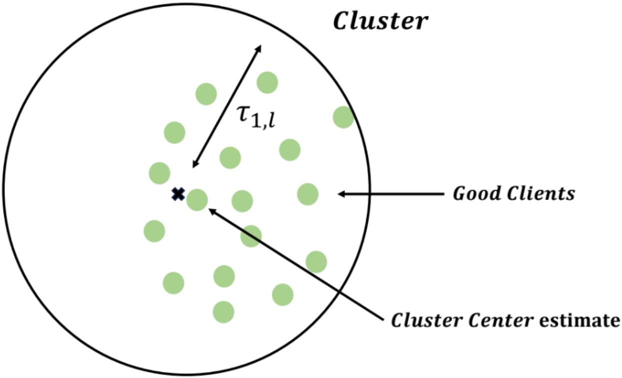


Figure: Byzantine Attack

Byzantine Robustness of Threshold Clustering

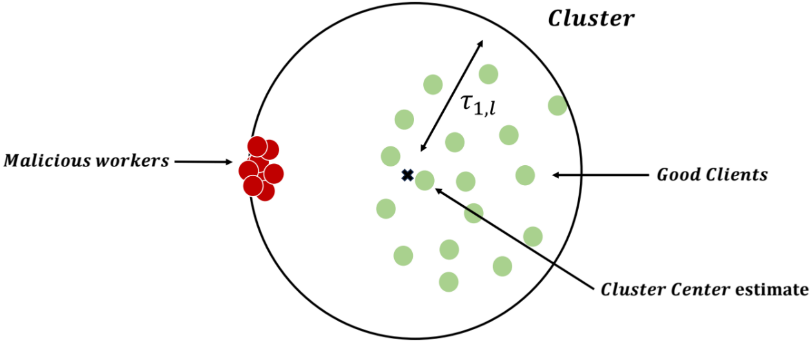


Figure: Byzantine Attack

Byzantine Robustness of Threshold Clustering

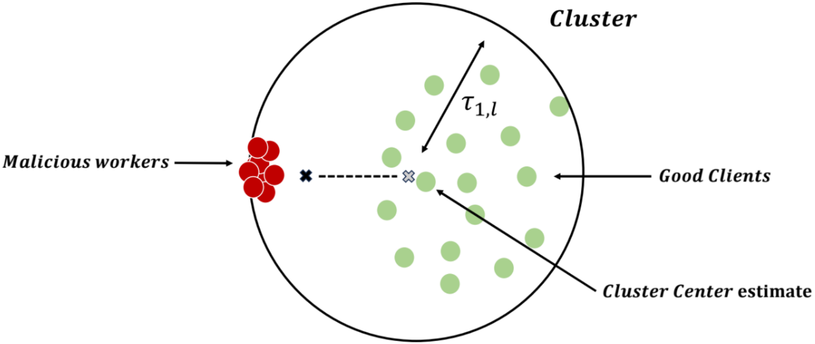


Figure: Byzantine Attack

Algorithm 3: Threshold-Clustering

Input: Points to be clustered: $\{z_1, \ldots, z_N\}$. Number of clusters: K.
Cluster-center initializations: $\{v_{1,0}, \ldots, v_{K,0}\}$.

for *round* $l \in [M]$ **do**
  for *cluster* k *in* $[K]$ **do**
    Set radius $\tau_{k,l}$;
    Update cluster-center estimate:

$$v_{k,l} = \frac{1}{N} \sum_{i=1}^{N} \left(\chi_{\{\|z_i - v_{k,l-1}\| \leq \tau_{k,l}\}} z_i + \chi_{\{\|z_i - v_{k,l-1}\| > \tau_{k,l}\}} v_{k,l-1} \right)$$

Output: Cluster-center estimates $\{v_1 = v_{1,M}, \ldots, v_K = v_{K,M}\}$.

[a]

[a] χ denotes the indicatrice function

Algorithm 2: Federated-Clustering

Input: Learning rate: η. Initial parameters for each client: $\{x_{1,0}, \dots, x_{N,0}\}$.

```
for client i ∈ [N] do
    Send x_{i,0} to all clients j ≠ i;
for round t ∈ [T] do
    for client i in [N] do
        Compute g_i(x_{j,t−1}) and send to client j for all j ≠ i ∈ [N];
    for client i in [N] do
        Compute
          v_{i,t} ← Threshold-Clustering({g_j(x_{i,t−1})}_{j∈[N]}; g_i(x_{i,t−1}));
        Update parameter: x_{i,t} ← x_{i,t−1} − η v_{i,t};
        Send x_{i,t} to all clients j ≠ i;
```

Output: Personalized parameters: $\{x_{1,T}, \dots, x_{N,T}\}$.

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\|\nabla f_i(x_{i,t-1})\|^2 \lesssim \sqrt{\frac{\max(1, A^2)(\sigma^2/n_i + \sigma^3/\Delta + \beta_i\sigma\Delta)}{T}}$$

Drawbacks

Federated Clustering suffers from several drawbacks:

- Communication overhead
- Doesn't use sufficiently the information accumulated by the previous clustering
- How to choose the threshold radius tau at each step?
- Computational inefficiency

Algorithm 4: Momentum-Clustering

Input: Learning rate: η. Initial parameters for each client: $\{x_{1,0}, \ldots, x_{N,0}\}$.

for *round* $t \in [T]$ **do**

  for *client i in* $[N]$ **do**

    Client i sends

$$m_{i,t} = \alpha g_i(x_{i,t-1}) + (1-\alpha) m_{i,t-1}$$

    to server.

  Server generates cluster centers

$$\{v_{k,t}\}_{k\in[K]} \leftarrow \text{Threshold-Clustering}(\{m_{i,t}\}_{i\in[N]}; K \text{ clusters}; \{v_{k,t-1}\}_{k\in[K]})$$

  and sends $v_{k_i,t}$ to client i, where k_i denotes the cluster to which i is assigned in this step. **for** *client i in* $[N]$ **do**

    Client i computes update: $x_{i,t} = x_{i,t-1} - \eta v_{k_i,t}$.

Output: Personalized parameters: $\{x_{1,T}, \ldots, x_{N,T}\}$.

Dataset : Each cluster has a different rotation of MNIST images

Implementation

Limitations of the momentum-clustering

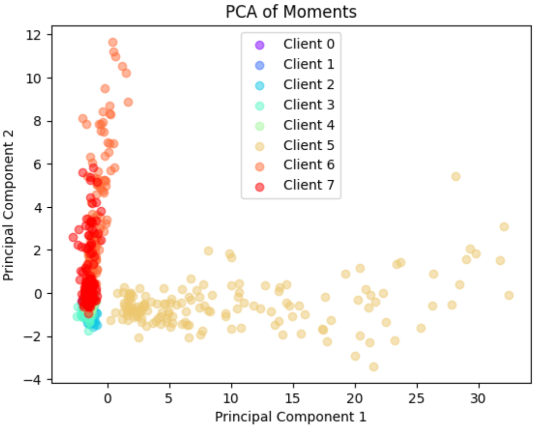


Figure: PCA of Momentums

Our contribution

We choose to focus on communication overhead to improve Federated Clustering (Algorithm 2)

Key Ideas

- Delete useless communications as you go along
- Using information from previous Threshold Clustering

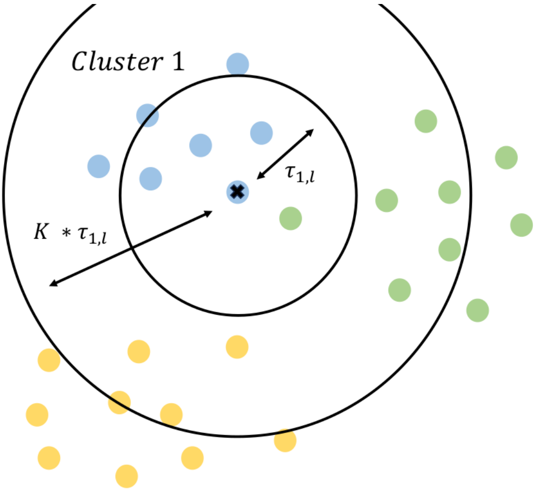


Figure: Our algorithm

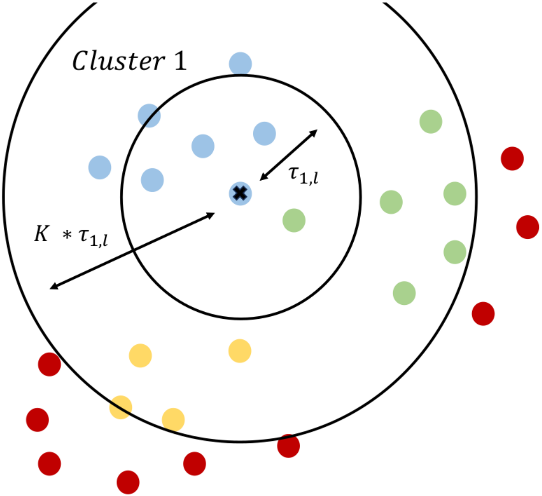


Figure: Our algorithm

Our Algorithm: Federated-Clustering++

Input: Learning rate: η. Initial parameters for each client: $\{x_{1,0}, \ldots, x_{N,0}\}$.

for *client* $i \in [N]$ **do**
    Send $x_{i,0}$ to clients j in *DiffusionList*$_i$;

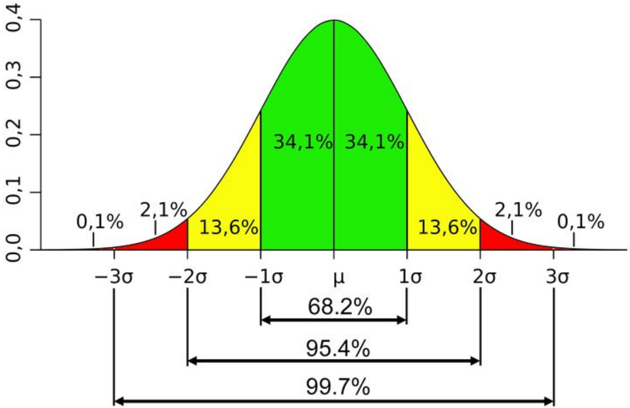
for *round* $t \in [T]$ **do**
    for *client* i *in* $[N]$ **do**
        Compute $g_i(x_{j,t-1})$ and send to client j for all j in *DiffusionList*$_i$;

    for *client* i *in* $[N]$ **do**
        Compute
        $v_{i,t} \leftarrow \text{Threshold-Clustering}(\{g_j(x_{i,t-1})\}_{j:i \in \mathit{DiffusionList}_j}; g_i(x_{i,t-1}))$;
        Update *DiffusionList*$_i$;

    Update parameter: $x_{i,t} \leftarrow x_{i,t-1} - \eta v_{i,t}$;
    Send $x_{i,t}$ to all clients j in *DiffusionList*$_i$;

Output: Personalized parameters: $\{x_{1,T}, \ldots, x_{N,T}\}$.

$$\mathbb{E}||g_i(x) - g_j(x)||^2 \leq 2\sigma^2$$
$$p = P(||g_i(x) - g_j(x)|| > R) < f(\mathbb{E}||g_i(x) - g_j(x)||^2, R)$$
$$\mathbb{E}(\hat{n}_i) = (1-p)^{Tn_i}$$



$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\|\nabla f_i(x_{i,t-1})\|^2 \lesssim \sqrt{\frac{\max(1, A^2)(\sigma^2/n_i + \sigma^3/\Delta + \beta_i\sigma\Delta)}{T}}$$

Implementation of FC++ on a personalized dataset

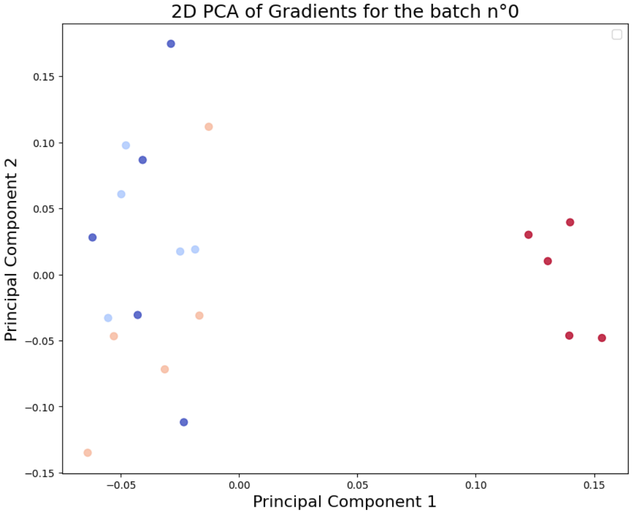


Figure: PCA of gradients for the batch n°0

Implementation of FC++ on a personalized dataset

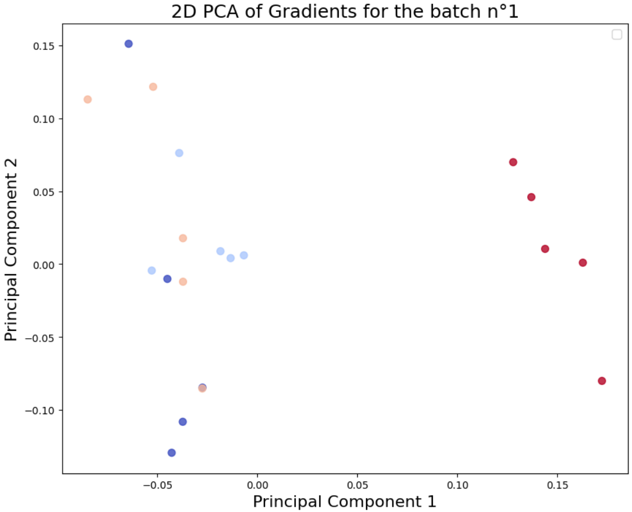


Figure: PCA of gradients for the batch n°1

Implementation of FC++ on a personalized dataset

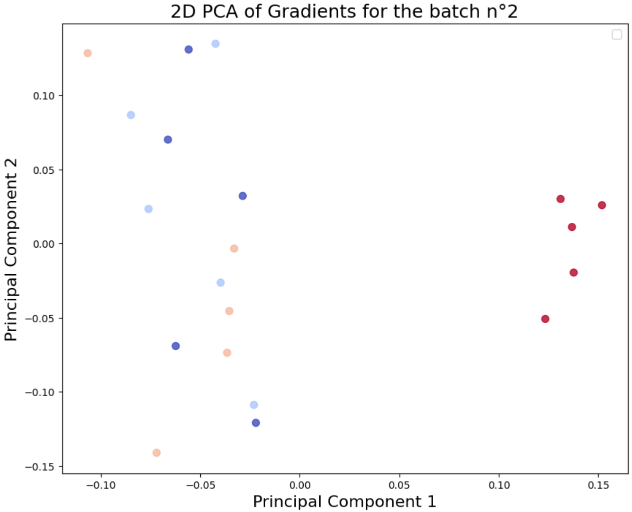


Figure: PCA of gradients for the batch n°2

Implementation of FC++ on a personalized dataset

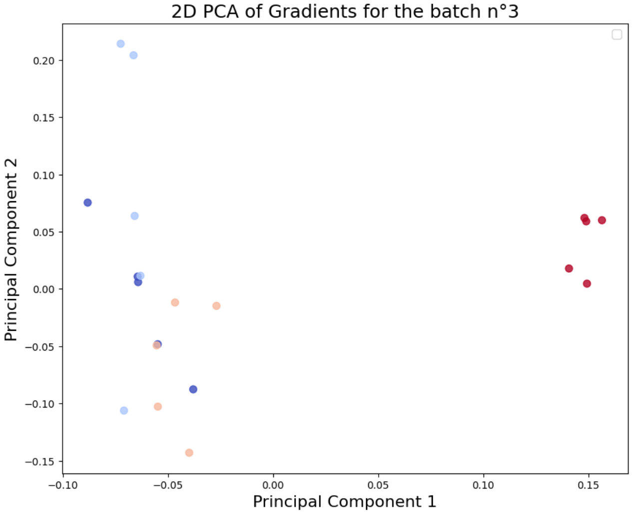


Figure: PCA of gradients for the batch n°3

Implementation of FC++ on a personalized dataset

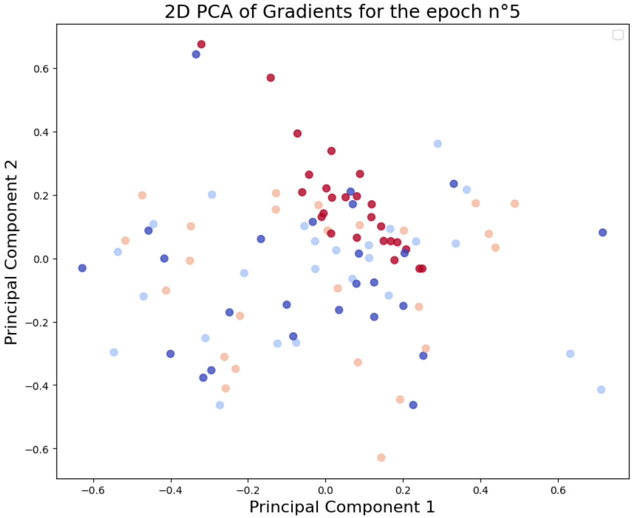


Figure: PCA of gradients for the batch n°4

Benchmarking

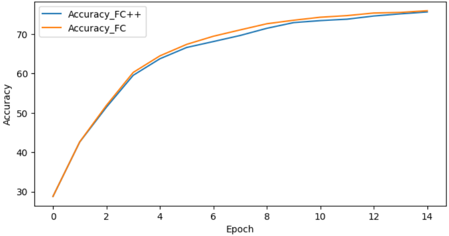


Figure: Accuracy per epoch

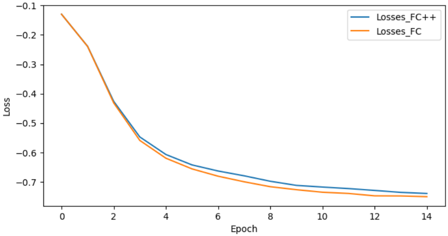


Figure: Losses per epoch

Experiments with 4 clusters, 60 clients, 15 epochs, 150 samples per client

Benchmarking

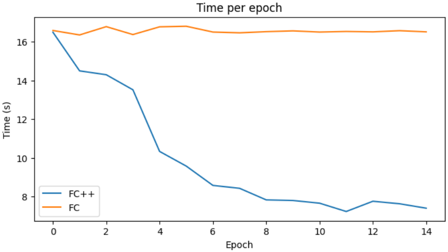


Figure: Time per epoch

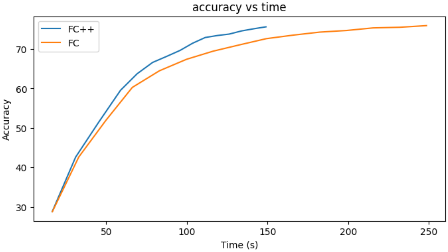


Figure: Accuracy versus time

Experiments with 4 clusters, 60 clients, 15 epochs, 150 samples per client

Conclusion

- Robustness and Personalized
- Communication improvements while remaining efficient
- Remaining challenges : Theoretical guarantees with realistic assumptions.

References

- Mariel Werner Lie He Sai Praneeth Karimireddy Michael Jordan Martin Jaggi , A. (2023). Provably Personalized and Robust Federated Learning, TLMR 2023