We show that in decentralized federated learning, even if you permanently lose a client, you can still converge to a well-performing consensus model

Check out the project website!

Mitigating Persistent Client Dropout in Asynchronous Decentralized Federated Learning

Ignacy Stępka, Nicholas Gisolfi, Kacper Trębacz, Artur Dubrawski
Carnegie Mellon University, Pittsburgh, PA, USA

Carnegie Mellon University Robotics Institute

RISS | Robotics Institute Summer Scholars

Introduction

Motivation

- Privacy: Data can't be shared directly (e.g., hospitals, regulations)
- Objective: Converge to a well-performing model on all clients
- Challenge: One client may be permanently lost during training

Problem Setting

- Data distribution: Each client has access to some unique data
- Collaboration: Clients share latest models with their neighbors
- Regularization: Clients consider models received from their neighbors in their local optimization

Proposed Approach

1. Recall the latest model shared by the destroyed client
2. Approx. training data via a gradient- or model-inversion attack
3. Deploy a new virtual agent back to the federation who will use the reconstructed synthetic dataset as its local training data

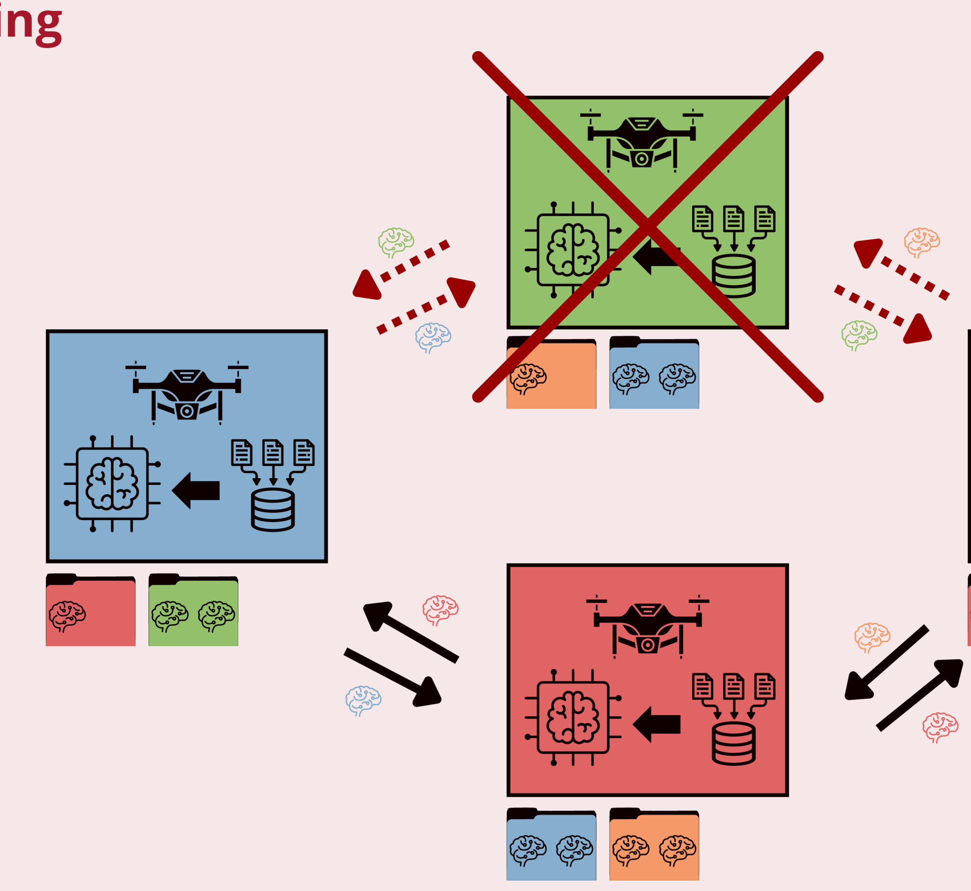
But how?

Decentralized Federated Learning

Algorithm 1 Asynchronous Decentralized Federated Learning framework

Require: No of clients m, initial models $\{\theta_i\}_{i=1}^m$, comm. graph G

```
1: Initialize each client i with model θ_i
2: while not converged do
3:   for all clients i in parallel do
4:     Sample E_i ~ U{5, 10}
5:     for e = 1 to E_i do
6:       θ_i ← arg min_{θ_i} L_i(θ_i, X_i, Y_i)
7:     end for
8:   end for
9:   Randomly select k pairs (i, j) such that g_ij ≠ 0
10:  for all selected pairs (i, j) do
11:    Clients i and j exchange and update their models
12:  end for
13: end while
14: return {θ_i}_{i=1}^m
```



Global objective

$$\underset{\theta_1,\dots,\theta_m}{\text{minimize}} \quad \frac{1}{m}\sum_{i=1}^{m}\mathcal{L}_i(\theta_i, X_i, Y_i) \quad \text{s.t.} \quad \mathcal{R}(\theta_1,\dots,\theta_m) = 0$$

Algorithm #1: DJAM (local optimization based) [2]

$$\mathcal{L}_{\text{DJAM}} = \mathcal{L}_d + \|\theta_i^t - \theta_i^{t-1}\|_2 + \frac{1}{2}\sum_{j=1}^{N} g_{ij}\|\theta_i - \theta_{ij}^t\|_2$$

$$\underset{\theta_i}{\text{minimize}} \quad \mathcal{L}_i(\theta_i, X_i, Y_i)$$

Algorithm #2: Function Space Regularization (local optimization based) [4]

$$\mathcal{L}_{\text{FSR}} = \mathcal{L}_d + \frac{1-\omega}{\omega}\|f_i^t - f_i^{t-1}\|_2 + \lambda\frac{1}{N}\sum_{j=1}^{N} g_{ij}\|f_i - f_{ij}^t\|_2$$

$$\underset{\theta_i}{\text{minimize}} \quad \mathcal{L}_i(\theta_i, X_i, Y_i)$$

Algorithm #3: Decentralized Federated Averaging (Fed-avg based) [3]

$$\theta_i^{t+1} = \sum_{j=1}^{N} g_{ij}\theta_j^t$$

Mitigation strategies

Simple strategies

No action

Serves as a sanity check for what would have happened if we don't act after noticing that one of the clients has been lost

Drop

Another sanity check. What if all other clients decide to simply not incorporate lost client's model in their local optimization?

Adaptive strategies

Model inversion

$$X' \sim \mathcal{U}[0,1]^d$$

$$Y' \sim \mathcal{U}\{1, C\}$$

$$\mathcal{L}_{MI} = \nabla_\theta \mathcal{L}_d(\theta, X, Y)$$

$$X'_{t+1} = X'_t - \eta\nabla_{X'_t}\mathcal{L}(\theta, X', Y')$$

Gradient inversion

$$X' \sim \mathcal{U}[0,1]^d$$

$$Y' \sim \mathcal{U}\{1, C\}$$

$$\mathcal{L}_{GI} = d(\nabla W' - \nabla W)^2 + \lambda\mathcal{L}_{\text{prior}}$$

∇W is the observed gradient

$$\nabla W' = \nabla_\theta \mathcal{L}_d(\theta, X', Y)$$

$$X'_{t+1} = X'_t - \eta\nabla_{X'_t}\mathcal{L}(\theta, X', Y')$$

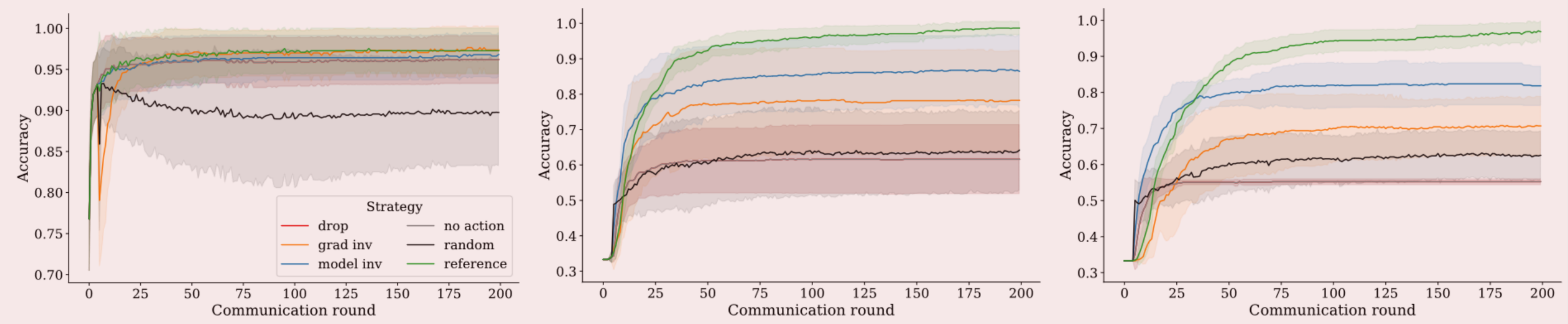
Random

A sanity check for adaptive strategies. Are model/gradient inversion attacks necessary? What if we simply create a new client with random local training data?

Holistic Scheme

1. Generate random data
2. Pick your attack's loss term
3. Run data optimization (reconstruction)
4. Get the reconstructed data, give it to a new virtual client
5. Continue your federated learning algorithm

Convergence plots

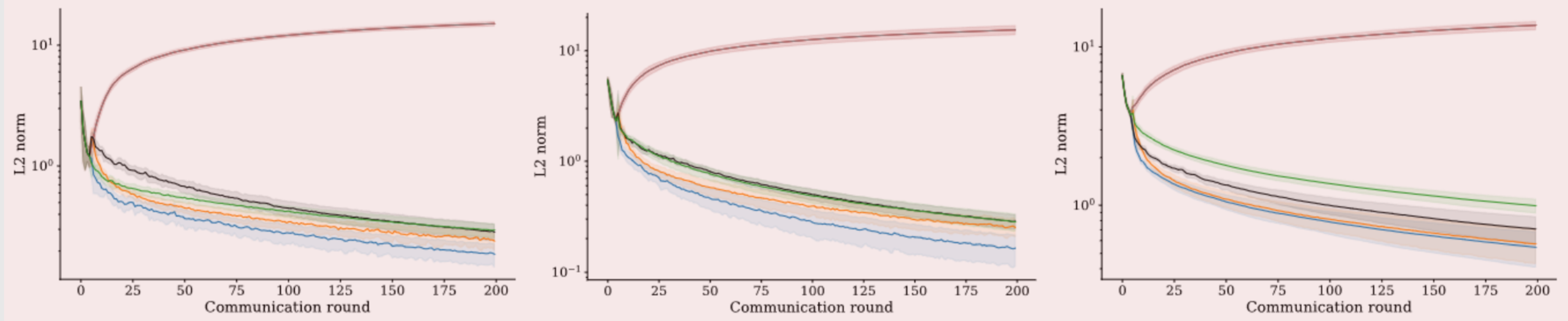


(a) iid | (b) non-iid (clusters) | (c) non-iid (classes)

Final accuracy

Dataset	Distribution	No action	Forget	Random	Grad inv	Model inv	Reference
wine	iid	0.96 ± 0.03	0.96 ± 0.03	0.90 ± 0.06	0.97 ± 0.03	0.97 ± 0.03	0.97 ± 0.03
	non-iid (clusters)	0.62 ± 0.10	0.62 ± 0.10	0.64 ± 0.11	0.78 ± 0.14	**0.86 ± 0.10**	0.99 ± 0.02
	non-iid (class)	0.55 ± 0.01	0.55 ± 0.01	0.63 ± 0.07	0.71 ± 0.08	**0.82 ± 0.05**	0.97 ± 0.03
iris	iid	0.90 ± 0.04	0.90 ± 0.04	0.89 ± 0.09	0.92 ± 0.09	**0.95 ±0.04**	0.97 ± 0.04
	non-iid (clusters)	0.64 ± 0.11	0.64 ± 0.11	0.70 ± 0.17	0.79 ± 0.17	**0.87 ± 0.12**	0.94 ± 0.05
	non-iid (class)	0.57 ± 0.04	0.57 ± 0.04	0.57 ± 0.13	0.62 ± 0.10	**0.73 ± 0.08**	0.84 ± 0.04
digits	iid	0.94 ± 0.01	0.94 ± 0.01	0.94 ± 0.01	0.95 ± 0.02	0.94 ± 0.02	0.95 ± 0.01
	non-iid (clusters)	0.75 ± 0.04	0.75 ± 0.04	0.76 ± 0.04	0.84 ± 0.06	**0.86 ± 0.04**	0.95 ± 0.02
	non-iid (class)	0.55 ± 0.02	0.55 ± 0.02	0.63 ± 0.05	0.69 ± 0.04	**0.75 ± 0.04**	0.93 ± 0.02

Similarity between client models



(g) DFedAvgM - iid | (h) DFedAvgM - non-iid (clusters) | (i) DFedAvgM - non-iid (classes)

Main takeaways

- On average, adaptive strategies based on data reconstruction outperform baselines and the random adaptive strategy.
- The final performance gain is most pronounced in non-iid scenarios, high lighting the importance of recovering client-specific information in heterogeneous federations
- Results are consistent across different model types - logistic regression, neural network architectures (see appendix)
- Results are consistent across different federated learning algorithms - DJAM, Function Space Regularization, DFedAvgM (see appendix)
- More research into what makes these adaptive strategies succesfull is needed, e.g., how noisy can the reconstructed data be? does it scale to large models and datasets?

Reconstructed images (digits)

(a) After 1 epoch | (b) After 800 epochs

References

[1] Ovi et al. 2023 "A Comprehensive Study of Gradient Inversion Attacks in Federated Learning and Baseline Defense Strategies"
[2] Almeida et al. 2018 "Distributed Jacobi Asynchronous Method for Learning Personal Models"
[3] Tsun et al. 2021 "Decentralized Federated Averaging"
[4] Good 2024 "Trustworthy Learning using Uncertain Interpretation of Data"
[5] Zhu et al. 2019 "Deep Leakage from Gradients"

Prior term (optional)

$$\mathcal{L}_{prior} = \sum_{i=1}^{d}\text{ReLU}(x-1) + \text{ReLU}(-x)$$

Gradient from update history

$$\nabla W = \frac{\theta_t - \theta_{t-1}}{\eta}$$

This work was partially supported by the NSF (awards 2406231 and 2427948), NIH (awards R01NS124642 and R01NR013912), DARPA (HR00112420329), and the US Army (W911NF-20-D0002).