FedFed: Feature Distillation against Data Heterogeneity in Federated Learning

Zhiqin Yang et al, NeurIPS 2023[1]

Presented by:
Souvik Sarkar
Department of Computer Science and Engineering
IIT Hyderabad

Guided by
Prof. C Krishna Mohan

September 23, 2024

Content

Introduction of Federated Learning

Federated Learning is a decentralized machine learning approach where models are trained on multiple devices or servers without sharing data.

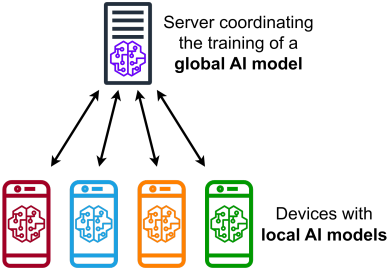


Figure 1: FL architecture

Introduction of FL

Global Objective

Minimize $L(\phi)$ over all clients' data distributions:

$$\min_{\phi} L(\phi) = \sum_{k=1}^{K} \lambda_k L_k(\phi_k)$$

Local Objective for Client k

$$L_k(\phi_k) = \mathbb{E}_{(x,y)\sim P(X_k,Y_k)} \left[\ell(\phi_k; x, y)\right]$$

Data remains local on clients; models are trained on private data, then aggregated.

Motivation - Importance of FL

- **Data Privacy Preservation**
  - Sensitive data (e.g., healthcare, financial) stays on local devices, reducing privacy risks.
- **Reducing Centralized Data Storage Costs**
  - Removes the need to centralize large datasets, reducing infrastructure and storage costs.
- **Scalability**
  - FL allows for distributed training across a large number of devices, improving scalability.
- **Security and Robustness**
  - FL can mitigate risks of a single point of failure compared to centralized models.

Motivation - Challeneges in FL

- **Data Privacy**
  - Data stays local, but updates may leak sensitive information.
- **Communication Overhead**
  - Frequent device-server communication leads to high bandwidth usage.
- **Heterogeneity**
  - **Data Heterogeneity :** Data on devices is often non-identically distributed (Non-IID), causing model bias.
  - **Device Heterogeneity :** Devices have varying computational capabilities, affecting model training speed.
- **Security Threats**
  - Vulnerable to adversarial attacks like model poisoning.

Problem Statement

Challenge:

- Federated Learning (FL) faces **data heterogeneity**, i.e., distribution shifting among clients.

Dilemma:

- **Sharing client information** helps mitigate heterogeneity, but it risks compromising privacy while aiming to improve model performance.

Key Question:

- Is it possible to share only **partial features** to address data heterogeneity while preserving privacy?

Proposed Solution

Proposed Solution: Federated Feature Distillation (FedFed)

- **Partition data** into:
  - **Performance-sensitive features:** Greatly contribute to model performance (shared globally).
  - **Performance-robust features:** Limited contribution to model performance (kept locally).
- Clients train models on both local and shared data to balance privacy and performance.

Challenges

- **Goal**: Efficiently share minimal information between clients while retaining **privacy**.
- **Challenges**:
  - How to divide data into **performance-sensitive** and **performance-robust** features.
  - Preserve local **private data** without hindering global model performance.

Methodology - FedFed Overview

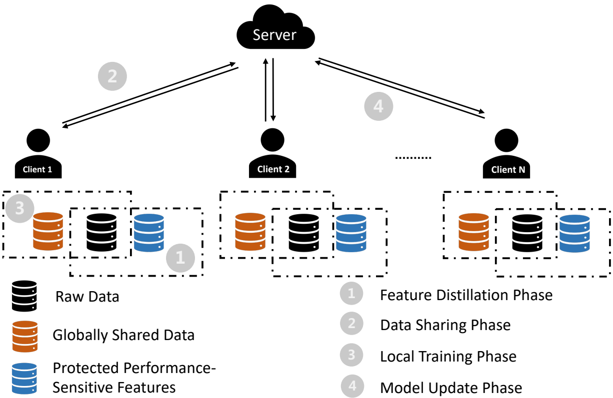
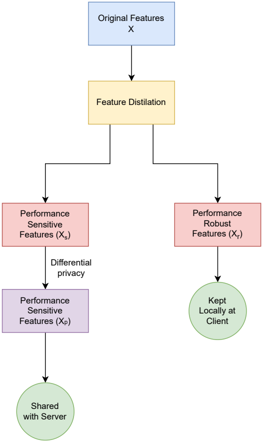


Figure 2: FedFed overview

Methodology - FedFed Overview



Methodology - Partitions

Definition : (Valid Partition)

A partition strategy is a method to partition a variable X into two parts in the same measure space such that $X = X_1 + X_2$. This strategy is valid if it satisfies:

(i) $H(X_1, X_2 \mid X) = 0$

(ii) $H(X \mid X_1, X_2) = 0$

(iii) $I(X_1; X_2) = 0$

where $H(\cdot)$ denotes the information entropy and $I(\cdot)$ denotes the mutual information.

Methodology - Partitions

Definition : (Performance-sensitive and Performance-robust Features)

Let $X = X_s + X_r$ be a valid partition strategy. We define:

- X_s as **performance-sensitive features** such that $I(X; Y \mid X_s) = 0$, where Y is the label of X.
- X_r as **performance-robust features**.

Methodology - Information Bottleneck

To divide the Features in two Partitions Authors takes inspiration from the **Information Bottleneck (IB)** Method [2].

- **Goal**: Compress input X while retaining information relevant to the output Y.

Objective :

$$L_{IB} = I(X;Y|Z), \quad \text{s.t.} \quad I(X;Z) \leq I_{IB}$$

- **Z**: Latent embedding that captures essential information.
- **IB's Goal**: Minimize mutual information between X and Y, while compressing X.

Methodology - Feature Distillation in FedFed

- **Goal**: Partition data features into:
  - **Performance-sensitive**: Essential for model performance.
  - **Performance-robust**: Retain private data information but not crucial for performance.

Objective :

$$\min_{Z} I(X;Y|Z), \text{ s.t. } I(X;X-Z|Z) \geq I_{FF}$$

  - $X - Z$: Performance-robust features.
  - Z: Performance-sensitive features for prediction.

Methodology - IB vs FedFed

- **FedFed**:
  - Aims to make the dismissed features (performance-robust features) **similar** to the original private data.
- **Information Bottleneck (IB)**:
  - Aims to make the preserved features (performance-sensitive features) as **dissimilar** as possible from the original data.
- **Information Dismissal:**
  - FedFed dismisses information directly in the **data space**.
  - IB works in the **representation space** (latent space).

Methodology - FedFed

In this context, the goal is to make the feature distillation process computationally feasible , The new objective function is derived for **client k** :

Objective :

$$\min_{\theta} -\mathbb{E}_{(x,y)\sim P(X_k,Y_k)} \log p(y|x - q(x;\theta)), \quad \text{s.t.} \quad ||x - q(x;\theta)||_2^2 \leq \rho$$

- $\boldsymbol{q(x;\theta)}$: is a generative model produces performance-robust features.
- $\boldsymbol{z(x;\theta) \equiv x - q(x;\theta)}$: represents performance-sensitive features.
- $\boldsymbol{\rho}$: is a hyperparameter that keep performance-sensitive features size bounded.

Methodology - FedFed

Objective : [3] [4]

$$\min_{\theta, w_k} -\mathbb{E}_{(x,y)\sim P(X_k,Y_k)} \ell(f(x - q(x;\theta); w_k), y), \text{ s.t. } \|z(x;\theta)\|_2^2 \leq \rho$$

- $\boldsymbol{f(\cdot; w_k)}$: is Local classifier trained on performance sensitive features to predict. labels.
- $\boldsymbol{\ell(\cdot)}$: is cross entropy loss between predicted label and the actual label.

Methodology - Differential Privacy

Motivation:

- Performance-sensitive features may contain private data.
- Differential Privacy (DP) is introduced to protect these features from privacy attacks, adversarial threats.

Noise Addition:

- Gaussian noise is added :

$$x_r + n, \quad n \sim \mathcal{N}(0, \sigma_r^2 I), \quad x_s + n, \quad n \sim \mathcal{N}(0, \sigma_s^2 I)$$

Training with DP:

- Clients train local models with private and shared data:

$$\min E(x, y) \sim P(X_k, Y_k)\ell(f(x; \varphi_k), y) +$$
$$E(x_p, y) \sim P(h(X_k), Y_k)\ell(f(x_p; \varphi_k), y)$$

- Shared features x_p protect privacy while enabling global model training.

Methodology

Algorithm 1 Feature Distillation

Server input: communication round T_d, DP noise level σ_s^2
Client k's input: local epochs of feature distillation E_d, k-th local dataset $\mathcal{D}^k$ and rescale to $[0, 1]$

Initialization: server distributes the initial model $\mathbf{w}^0, \theta^0$ to all clients
Server Executes:
for each round $t = 1, 2, \cdots, T_d$ **do**
  server samples a subset of clients $C_t \subseteq \{1, ..., K\}$,
  server **communicates** $\mathbf{w}^t, \theta^t$ to selected clients
  for each client $k \in C_t$ **in parallel do**
    $\mathbf{w}_k^{t+1}, \theta_k^{t+1} \leftarrow$ Local_FeatDis $(\mathbf{w}^t, \theta^t, \sigma_s^2)$
  end for
  $\mathbf{w}^{t+1}, \theta^{t+1} \leftarrow$ AGG $(\mathbf{w}_k^t, \theta_k^t, k \in C_t)$
end for
$\mathcal{D}^s = \{\mathcal{D}_k^s\}_{k=1}^K \leftarrow$Collecting $\mathbf{x}_p$ generated by k-th client use Eq (9), where $\mathbf{x}_p = \mathbf{x}_s + \mathbf{n}$

Local_FeatDis($\mathbf{w}^t, \theta^t, \sigma_s^2$):
for each local epoch e with $e = 1, \cdots, E_d$ **do**
  $\mathbf{w}_k^{t+1}, \theta_k^{t+1} \leftarrow$ SGD update use Eq (9).
end for
Return $\mathbf{w}_k^{t+1}, \theta_k^{t+1}$ to server

Methodology - FedFed Pipeline

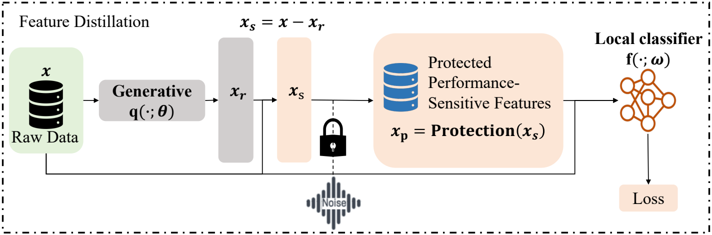


Figure 4: FedFed pipeline

Experimental setup

Models Used:

- ResNet-18: Utilized for feature distillation and classifier.
- β-VAE: Encoder-decoder structure for generating performance-robust features.

Federated Learning Algorithms:

- FedAvg, FedProx, SCAFFOLD, FedNova

Datasets:

- CIFAR-10, CIFAR-100
- Fashion-MNIST (FMNIST)
- SVHN

Experimental Setup

Data Partitioning:

- Dirichlet : Non-IID distribution ($\alpha = 0.1, 0.05$).

Hyperparameters:

- α (Dirichlet Parameter): 0.1, 0.05.
- K (Number of Clients): 10, 100.
- E_d (Local Epochs for Feature Distillation): 1, 5.
- T_d (Communication Rounds): Varied based on experiments.
- ρ (DP Strength Parameter): Adjusted for privacy vs accuracy tradeoff.
- σ_s^2 (DP Noise Level): For privacy protection.

Results

	centralized training ACC = 75.56% w/(w/o) **FedFed**							
	ACC↑	Gain↑	Round ↓	Speedup↑	ACC↑	Gain↑	Round ↓	Speedup↑
	$\alpha = 0.1, E = 1, K = 10$ (Target ACC =67%)				$\alpha = 0.05, E = 1, K = 10$ (Target ACC =61%)			
FedAvg	**69.64**(67.84)	1.8↑	**283**(495)	×**1.7** (×1.0)	**68.49**(62.01)	6.48↑	**137**(503)	×**3.7**(×1.0)
FedProx	**70.02**(65.34)	4.68 ↑	**233**(None)	×**2.1**(None)	**69.03**(61.29)	7.74↑	**141**(485)	×**3.6**(1.0)
SCAFFOLD	**70.14**(67.23)	2.91↑	**198**(769)	×**2.5**(× 0.6)	**69.32**(58.78)	10.54↑	**81**(None)	×**6.2**(None)
FedNova	**70.48**(67.98)	2.5↑	**147**(432)	×**3.4**(×1.1)	**68.92**(60.53)	8.39↑	**87**(None)	×**5.8**(None)
	$\alpha = 0.1, E = 5, K = 10$ (Target ACC =69%)				$\alpha = 0.1, E = 1, K = 100$ (Target ACC =48%)			
FedAvg	**70.96**(69.34)	1.62↑	**79**(276)	×**3.5**(×1.0)	**60.58**(48.21)	12.37↑	**448**(967)	×**2.2**(×1.0)
FedProx	**69.66**(62.32)	7.34↑	**285**(None)	×**1.0**(None)	**67.69**(48.78)	18.91↑	**200**(932)	×**4.8**(×1.0)
SCAFFOLD	**70.76**(70.23)	0.53↑	**108**(174)	×**2.6**(×1.6)	**66.67**(51.03)	15.64↑	**181**(832)	×**5.3**(×1.2)
FedNova	**69.98**(69.78)	0.2↑	**89**(290)	×**3.1**(×1.0)	**67.62**(48.03)	19.59↑	**198**(976)	×**4.9**(×1.0)

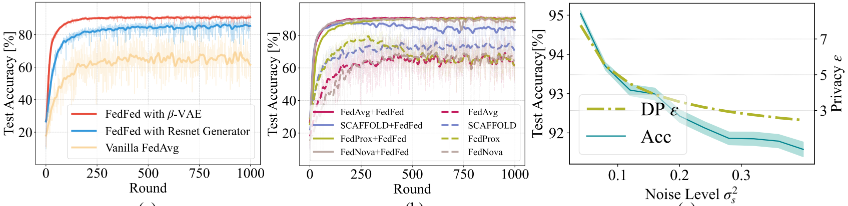
Figure 5: CIFAR-100 results

Results

	centralized training ACC = 96.56% w/(w/o) **FedFed**							
	ACC↑	Gain↑	Round ↓	Speedup↑	ACC↑	Gain↑	Round ↓	Speedup↑
	$\alpha = 0.1, E = 1, K = 10$ (Target ACC =88%)				$\alpha = 0.05, E = 1, K = 10$ (Target ACC =82%)			
FedAvg	**93.21**(88.34)	4.87↑	**105**(264)	×**2.5**(×1.0)	**93.49**(82.76)	10.73↑	**194**(365)	×**1.9**(×1.0)
FedProx	**91.80**(86.23)	5.574↑	**233**(None)	×**1.1**(None)	**93.21**(79.43)	13.78↑	**37**(None)	×**9.9**(None)
SCAFFOLD	**88.41**(80.12)	8.29↑	**357**(None)	×**0.**(None)	**90.27**(75.87)	14.4↑	**64**(None)	×**5.7**(None)
FedNova	**92.98**(89.23)	3.75↑	**113**(276)	×**2.3**(×1.0)	**93.05**(82.32)	10.73↑	**37**(731)	×**9.9**(×0.5)
	$\alpha = 0.1, E = 5, K = 10$ (Target ACC =87%)				$\alpha = 0.1, E = 1, K = 100$ (Target ACC =89%)			
FedAvg	**93.77**(87.24)	6.53↑	**105**(128)	×**1.2**(×1.0)	**91.04**(89.32)	1.72↑	**763**(623)	×0.8(×**1.0**)
FedProx	**91.15**(77.21)	13.94↑	**142**(None)	×**0.9**(None)	**91.41**(88.76)	2.65↑	**733**(645)	×0.8(×**1.0**)
SCAFFOLD	**93.78**(80.98)	12.8↑	**20**(None)	×**6.4**(None)	**92.73**(88.32)	4.41↑	**507**(687)	×**1.2**(×0.9)
FedNova	**93.66**(89.03)	4.63↑	**52**(177)	×**2.5**(×0.7)	**84.05**(81.87)	2.18↑	None(None)	None(None)

Figure 6: SVHN results

Results



(a) (b) (c)

(a) Convergence rate of different generative models compared with vanilla FedAvg.

(b) Test accuracy and convergence rate on different federated learning algorithms with or without FedFed under $\alpha = 0.1$, E = 1, K = 100.

(c) Test accuracy on FMNIST with different noise level σ_s^2.

Future work

- **Real-Time Applications:** Explore the deployment of FedFed in real-time Federated Learning (FL) applications:
  - Recommendation systems
  - Healthcare systems
    - PathMNIST, OCTMNIST, TissueMNIST etc. [5]
- **Hardware Optimization:**
  - FedFed introduces additional communication and storage overhead. Investigate hardware-friendly versions for resource-constrained environments.

References

[1] Z. Yang, Y. Zhang, Y. Zheng, *et al.*, *Fedfed: Feature distillation against data heterogeneity in federated learning*, 2023. arXiv: 2310.05077 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2310.05077.

[2] N. Tishby, F. C. Pereira, and W. Bialek, *The information bottleneck method*, 2000. arXiv: physics/0004057 [physics.data-an]. [Online]. Available: https://arxiv.org/abs/physics/0004057.

[3] R. Shwartz-Ziv and N. Tishby, *Opening the black box of deep neural networks via information*, 2017. arXiv: 1703.00810 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1703.00810.

[4] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, *Tackling the objective inconsistency problem in heterogeneous federated optimization*, 2020. arXiv: 2007.07481 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2007.07481.

[5] J. Yang, R. Shi, D. Wei, *et al.*, "Medmnist v2 - a large-scale lightweight benchmark for 2d and 3d biomedical image classification," *Scientific Data*, vol. 10, no. 1, Jan. 2023, ISSN: 2052-4463. DOI: 10.1038/s41597-022-01721-8. [Online]. Available: http://dx.doi.org/10.1038/s41597-022-01721-8.

Thank You!

Questions?