
Special Topics in AI

Lecture 8: Statistical Heterogeneity in FL

University of Mazandaran

Fall 1405

Recap: Federated Learning

A decentralized learning framework in which a server and several clients collaboratively train a ML model, without sharing data

Recap: Federated Learning

A decentralized learning framework in which a server and several clients collaboratively train a ML model, without sharing data

Defining characteristics:

- ▶ Data is generated locally and remains decentralized
- ▶ Clients cannot see each other's data
- ▶ Server cannot read clients' data
- ▶ Data may not be I.I.D.

Recap: Federated Learning

A decentralized learning framework in which a server and several clients **collaboratively train a ML model, without sharing data**

Defining characteristics:

- ▶ Data is generated locally and remains decentralized
- ▶ Clients cannot see each other's data
- ▶ Server cannot read clients' data
- ▶ Data may not be I.I.D.

Key design questions:

- ▶ How to aggregate client models?
- ▶ What loss function should be used in the clients?
- ▶ What optimization algorithm should be used in the clients?
- ▶ What should be shared between clients and server?
- ▶ How much local computation should clients perform before communicating with the server?

Refers to the case where data available at each client come from different underlying distributions

- ▶ Client A (a hospital) may mostly have X-ray images of children,
- ▶ Client B may have adult patient data
- ▶ Client C may have images from a different device or resolution!

Each client's data represents a different distribution, i.e., data is non-iid

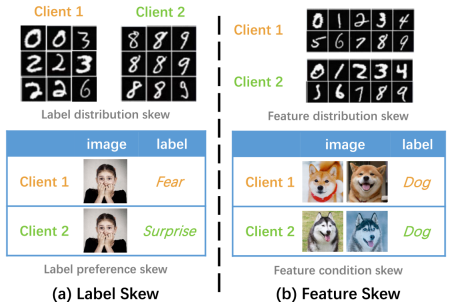
Types of Statistical Heterogeneity

► label skew



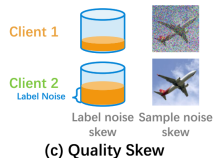
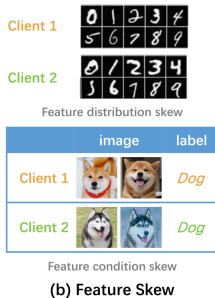
Types of Statistical Heterogeneity

- ▶ label skew
- ▶ feature skew



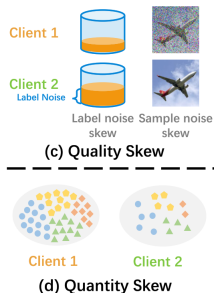
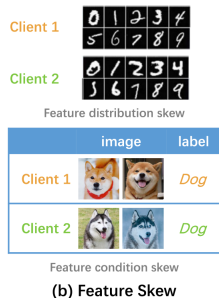
Types of Statistical Heterogeneity

- ▶ label skew
- ▶ feature skew
- ▶ quality skew



Types of Statistical Heterogeneity

- ▶ label skew
- ▶ feature skew
- ▶ quality skew
- ▶ quantity skew



Effect of Statistical Heterogeneity

- ▶ Model bias
- ▶ Slower convergence
- ▶ Reduced model accuracy
- ▶ Instability in training
- ▶ Communication inefficiency

Effect of Statistical Heterogeneity

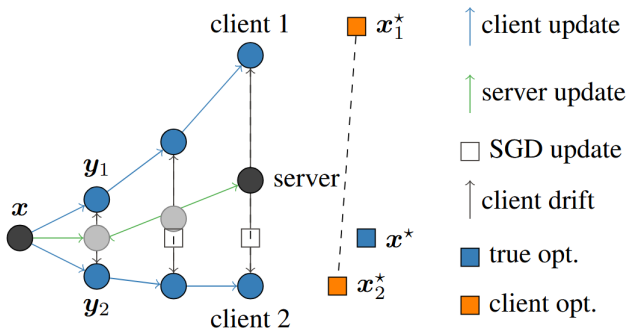


Figure 1. Client-drift in FEDAVG is illustrated for 2 clients with 3 local steps ($N = 2, K = 3$). The local updates y_i (in blue) move towards the individual client optima x_i^* (orange square). The server updates (in red) move towards $\frac{1}{N} \sum_i x_i^*$ instead of to the true optimum x^* (black square).

Karimireddy et al. "Scaffold: Stochastic controlled averaging for federated learning." In International Conference on Machine Learning (ICML), pp. 5132-5143, 2020

Effect of Statistical Heterogeneity

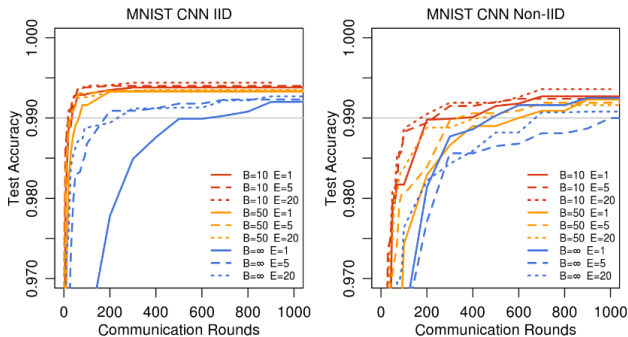
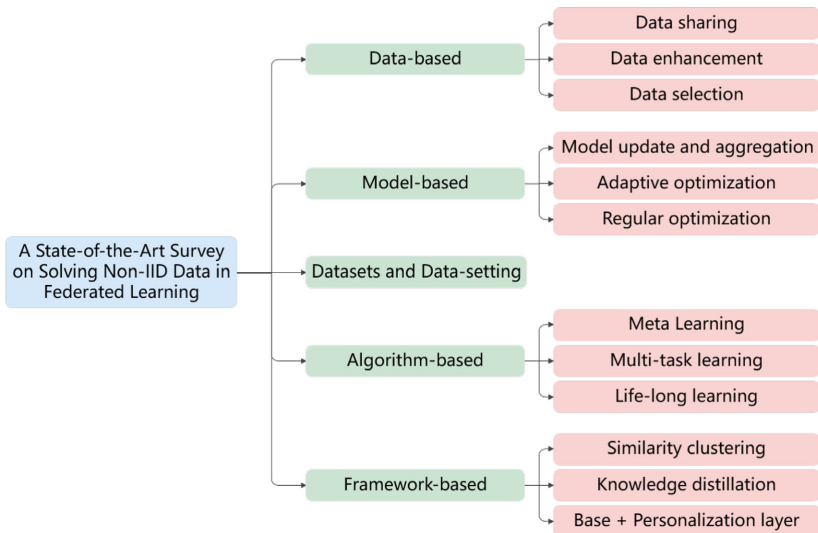


Figure 2: Test set accuracy vs. communication rounds for the MNIST CNN (IID and then pathological non-IID) with $C = 0.1$ and optimized η .

Approaches



Ma et al. "A state-of-the-art survey on solving non-iid data in federated learning." *Future Generation Computer Systems*, v. 135, pp. 244-258, 2022.

A very simple, yet effective approach

- ▶ In client drift, clients diverge from the global model
- ▶ Solution: don't let them diverge (too much!)

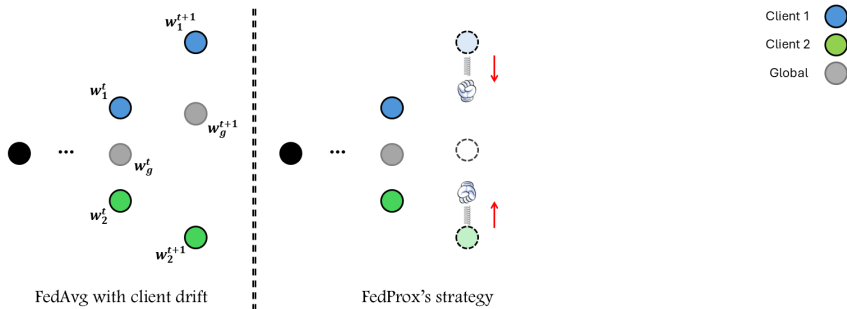
¹ Li et al. "Federated optimization in heterogeneous networks." *Proceedings of Machine Learning and Systems (MLSys)*, 2, pp. 429-450, 2020.

Intuition

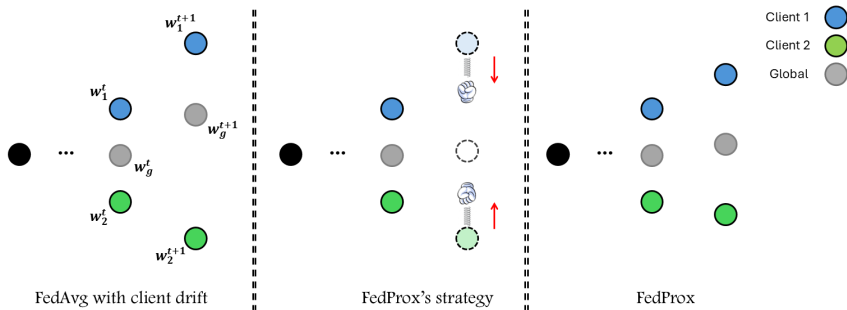


FedAvg with client drift

Intuition



Intuition



Approach

In each round $t + 1$, each client k optimizes

$$w_k^{t+1} \approx \arg \min_w F_k(w) + \frac{\mu}{2} \|w - w^t\|^2$$

Approach

In each round $t + 1$, each client k optimizes

$$w_k^{t+1} \approx \arg \min_w F_k(w) + \underbrace{\frac{\mu}{2} \|w - w^t\|^2}_{\text{proximal term}}$$

Approach

In each round $t + 1$, each client k optimizes

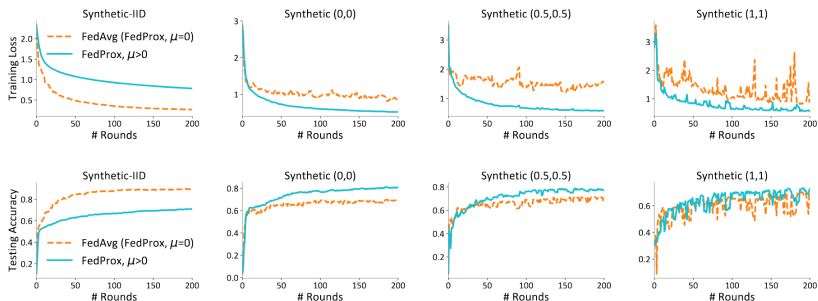
$$w_k^{t+1} \approx \arg \min_w F_k(w) + \underbrace{\frac{\mu}{2} \|w - w^t\|^2}_{\text{proximal term}}$$

- ▶ The proximal term explicitly limits the impact of heterogeneous local updates

Pseudocode

Input: $K, T, \mu, \gamma, w^0, N, p_k, k = 1, \dots, N$
for $t = 0, \dots, T - 1$ **do**
 Server selects a subset S_t of K devices at random (each device k is chosen with probability p_k)
 Server sends w^t to all chosen devices
 Each chosen device $k \in S_t$ finds a w_k^{t+1} which is a γ_k^t -inexact minimizer of: $w_k^{t+1} \approx \arg \min_w h_k(w; w^t) = F_k(w) + \frac{\mu}{2} \|w - w^t\|^2$
 Each device $k \in S_t$ sends w_k^{t+1} back to the server
 Server aggregates the w 's as $w^{t+1} = \frac{1}{K} \sum_{k \in S_t} w_k^{t+1}$
end for

Performance



- ▶ $E = 20$ is fixed
- ▶ From left to right, data become more heterogeneous

In each round of FedAvg

- ▶ All clients start their local training from the same initial model (global model)

In each round of FedAvg

- ▶ All clients start their local training from the same initial model (global model)
- ▶ The global model could perform poorly for local dataset of some clients

In each round of FedAvg

- ▶ All clients start their local training from the same initial model (global model)
- ▶ The global model could perform poorly for local dataset of some clients
- ▶ Solution: personalize the global model for each client

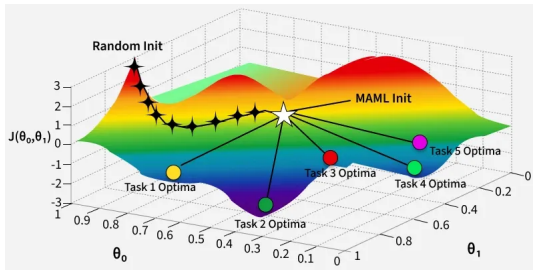
Model-Agnostic Meta-Learning (MAML)²

- ▶ Meta learning: a learning paradigm in which a model learns, from experience across multiple tasks, how to adapt to a new task effectively (learning to learn)
- ▶ MAML learns an initialization of parameters such that the model can adapt to any new task in the distribution in relatively fewer steps than random initialization

² Finn et al. "Model-agnostic meta learning for fast adaptation of deep networks." Proceedings of International Conference on Machine Learning (ICML), pp. 1126-1135, 2017.

Model-Agnostic Meta-Learning (MAML)²

- ▶ Meta learning: a learning paradigm in which a model learns, from experience across multiple tasks, how to adapt to a new task effectively (learning to learn)
- ▶ MAML learns an initialization of parameters such that the model can adapt to any new task in the distribution in relatively fewer steps than random initialization



[link](#)

² Finn et al. "Model-agnostic meta learning for fast adaptation of deep networks." Proceedings of International Conference on Machine Learning (ICML), pp. 1126-1135, 2017.

Personalization via meta learning

Personalization refers to finding an **initial shared model** between all clients which performs well **after each client updates it** w.r.t. its own loss function by performing **a few steps of gradient descent**

Personalization via meta learning¹

Let the clients start from different personalized initial model

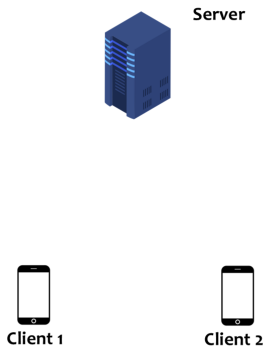
- ▶ **Step 1:** each client receives the global model from the server
- ▶ **Step 2:** each client performs a few steps of gradient descent on its own loss function using the global model
- ▶ **Step 3:** the new personalized model is used as the initial model for the loss function
- ▶ **Step 4:** each client applies gradient descent on its loss function
- ▶ **Note:** the data used in steps 2 and 4 are different

¹Fallah et al. "Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach." Advances in neural information processing systems, v. 33, pp. 3557-3568, 2020.

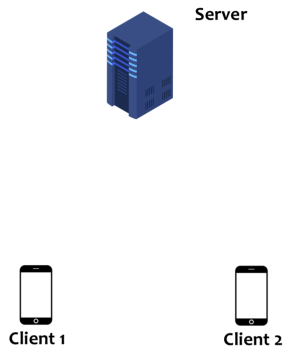
Personalized Federated Learning

Personalization via meta learning

FedAvg

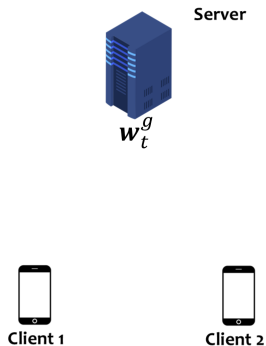


PerFedAvg

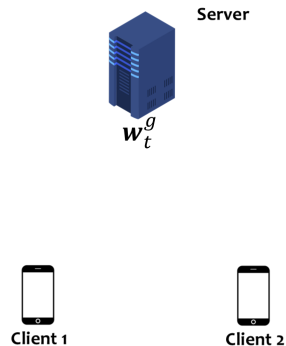


Personalization via meta learning

FedAvg

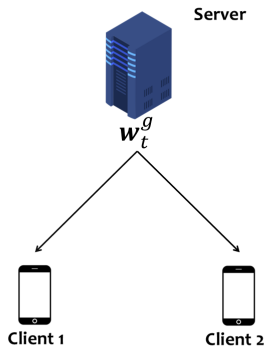


PerFedAvg

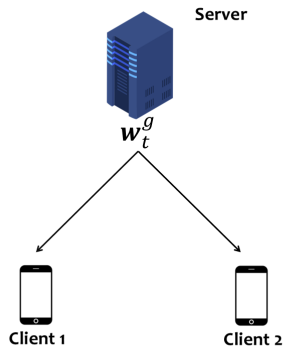


Personalization via meta learning

FedAvg



PerFedAvg



Personalized Federated Learning

Personalization via meta learning

FedAvg



Server



Client 1



Client 2

$$\mathcal{L}_1 = f_1(\mathbf{w}_t^g)$$

$$\mathcal{L}_2 = f_2(\mathbf{w}_t^g)$$

PerFedAvg



Server



Client 1



Client 2

$$\mathbf{w}_t^{g1} = \mathbf{w}_t^g - \alpha \nabla f_1(\mathbf{w}_t^g)$$

$$\mathcal{L}_1 = f_1(\mathbf{w}_t^{g1})$$

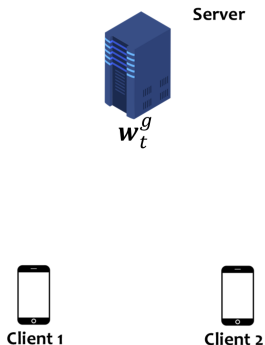
$$\mathbf{w}_t^{g2} = \mathbf{w}_t^g - \alpha \nabla f_2(\mathbf{w}_t^g)$$

$$\mathcal{L}_2 = f_2(\mathbf{w}_t^{g2})$$

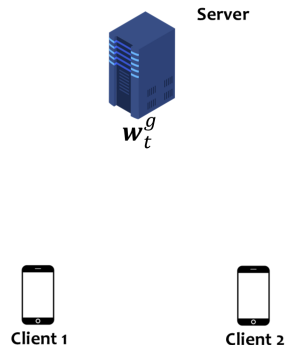
Personalized Federated Learning

Personalization via meta learning

FedAvg



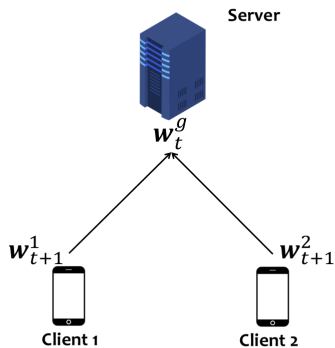
PerFedAvg



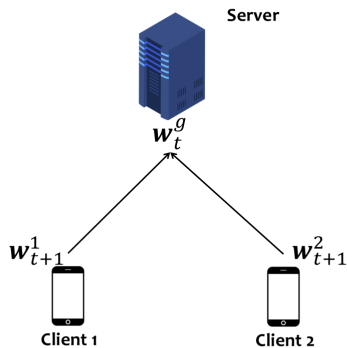
$\mathcal{L}_1 = f_1(w_t^g)$		$w_t^{g1} = w_t^g - \alpha \nabla f_1(w_t^g)$		$w_t^{g2} = w_t^g - \alpha \nabla f_2(w_t^g)$	
$\mathcal{L}_2 = f_2(w_t^g)$		$\mathcal{L}_1 = f_1(w_t^{g1})$		$\mathcal{L}_2 = f_2(w_t^{g2})$	
$w_{t+1}^1 = w_t^g - \eta \nabla f_1(w_t)$		$w_{t+1}^1 = w_t^{g1} - \beta \nabla f_1(w_t^{g1})$		$w_{t+1}^2 = w_t^{g2} - \beta \nabla f_2(w_t^{g2})$	

Personalization via meta learning

FedAvg



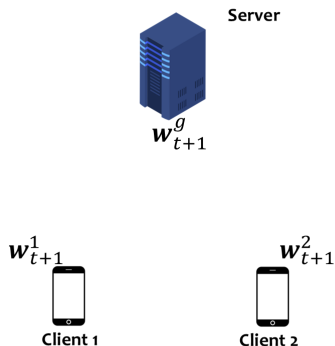
PerFedAvg



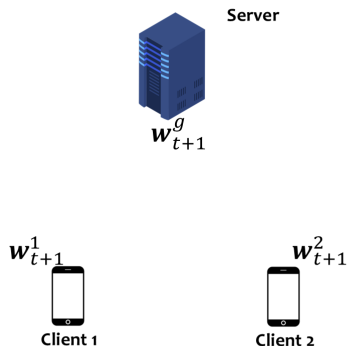
Personalized Federated Learning

Personalization via meta learning

FedAvg



PerFedAvg



Personalized Federated Learning

Table 1: Comparison of test accuracy of different algorithms given different parameters

Dataset	Parameters	Algorithms		
		FedAvg + update	Per-FedAvg (FO)	Per-FedAvg (HF)
MNIST	$\tau = 10, \alpha = 0.01$	75.96% $\pm$ 0.02%	78.00% $\pm$ 0.02%	79.85% $\pm$ 0.02%
	$\tau = 4, \alpha = 0.01$	60.18 % $\pm$ 0.02%	64.55% $\pm$ 0.02%	70.94% $\pm$ 0.03%
CIFAR-10	$\tau = 10, \alpha = 0.001$	40.49% $\pm$ 0.07%	46.98% $\pm$ 0.1%	50.44% $\pm$ 0.15%
	$\tau = 4, \alpha = 0.001$	38.38% $\pm$ 0.07%	34.04% $\pm$ 0.08%	43.73% $\pm$ 0.11%
	$\tau = 4, \alpha = 0.01$	35.97% $\pm$ 0.17%	25.32% $\pm$ 0.18%	46.32% $\pm$ 0.12%
	$\tau = 4, \alpha = 0.01$, diff. hetero.	58.59% $\pm$ 0.11%	37.71% $\pm$ 0.23%	71.25% $\pm$ 0.05%

Personalization via knowledge distillation²

Let the clients have personalized model architecture

- ▶ Each client is considered as a student
- ▶ Clients share per-class logits instead of model parameters
- ▶ Aggregated logits act as teacher

²Jeong et al. "Communication-efficient on-device machine learning: federated distillation and augmentation under non-IID private data." Proc. NeurIPS Workshop on MLPCD, 2018.

Personalized Federated Learning

Step 1. Each client trains its local model using its own dataset
(Update)

Personalized Federated Learning

- Step 1. Each client trains its local model using its own dataset (Update)
- Step 2. Each client predicts logits of its data input and calculate local average logit for each class (Prediction)

Personalized Federated Learning

- Step 1. Each client trains its local model using its own dataset (Update)
- Step 2. Each client predicts logits of its data input and calculate local average logit for each class (Prediction)

Assume the k^{th} client's dataset is $\mathcal{D}_k$ which can be divided into N set $\{\mathcal{D}_{k,c_i}\}_{i=1}^N$ where N is total number of classes. In Step 2, each client computes the following for each class:

$$t_{k,c_i} = \frac{1}{|\mathcal{D}_{k,c_i}|} \sum_{d \in \mathcal{D}_{k,c_i}} f(d)$$

If there is no sample pertaining to class c_j , then $t_{k,c_j} = 0$

Personalized Federated Learning

- Step 1. Each client trains its local model using its own dataset (Update)
- Step 2. Each client predicts logits of its data input and calculate local average logit for each class (Prediction)
- Step 3. The local average logits are uploaded from each client to the server (Upload)

Personalized Federated Learning

- Step 1. Each client trains its local model using its own dataset (Update)
- Step 2. Each client predicts logits of its data input and calculate local average logit for each class (Prediction)
- Step 3. The local average logits are uploaded from each client to the server (Upload)
- Step 4. The server aggregates per class logits to get global average logits per class (Aggregate)

Personalized Federated Learning

- Step 1. Each client trains its local model using its own dataset (Update)
- Step 2. Each client predicts logits of its data input and calculate local average logit for each class (Prediction)
- Step 3. The local average logits are uploaded from each client to the server (Upload)
- Step 4. The server aggregates per class logits to get global average logits per class (Aggregate)

$$t_{g,c_i} = \frac{1}{|K_{c_i}|} \sum_{k \in K_{c_i}} t_{k,c_i}$$

where K_{c_i} is the subset of clients having any sample pertaining to class c_i .

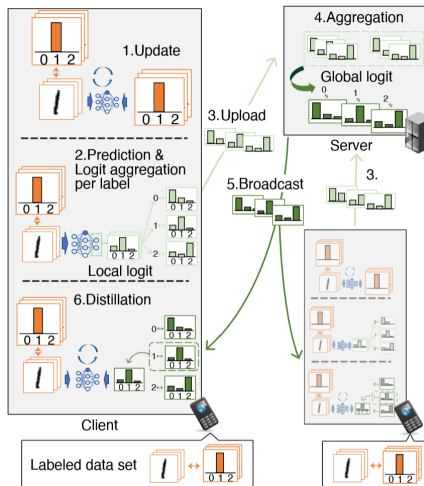
Personalized Federated Learning

- Step 1. Each client trains its local model using its own dataset (Update)
- Step 2. Each client predicts logits of its data input and calculate local average logit for each class (Prediction)
- Step 3. The local average logits are uploaded from each client to the server (Upload)
- Step 4. The server aggregates per class logits to get global average logits per class (Aggregate)
- Step 5. The server broadcasts the global average logits to all clients (Broadcast)

Personalized Federated Learning

- Step 1. Each client trains its local model using its own dataset (Update)
- Step 2. Each client predicts logits of its data input and calculate local average logit for each class (Prediction)
- Step 3. The local average logits are uploaded from each client to the server (Upload)
- Step 4. The server aggregates per class logits to get global average logits per class (Aggregate)
- Step 5. The server broadcasts the global average logits to all clients (Broadcast)
- Step 6. Each client updates the model parameters using the global average logits and the local average logits (Distillation)

Personalized Federated Learning



Itahara et al. "Distillation-based semi-supervised federated learning for communication-efficient collaborative training with non-iid private data." IEEE Transactions on Mobile Computing, 22(1), pp. 191-205, 2021