Personalized Federated Learning under Local Supervision

Qiqi Liu[1,2*], Jiaqiang Li[3*], Yuchen Liu[4], Yaochu Jin[1]†, Lingjuan Lyu[5], Xiaohu Wu[6], Han Yu[7]
[1] Westlake University, Hangzhou, China
[2] Westlake Institute for Advanced Study, Hangzhou, China
[3] East China University of Science and Technology, Shanghai, China
[4] Zhejiang University, Hangzhou, China
[5] Sony AI
[6] Beijing University of Posts and Telecommunications, Beijing, China
[7] Nanyang Technological University, Singapore
{qiqi6770304@gmail.com, jiaqiangli1626@gmail.com, jinyaochu@westlake.edu.cn}

Abstract

A crucial issue in federated learning is the heterogeneity of data across clients, which may lead to model divergence, eventually deteriorating the model performance. Personalized federated learning (pFL) has been shown to be an effective approach to addressing data heterogeneity in federated learning. However, many existing pFL studies rely on directly using the global model for local training without fully assessing its impact on the performance of the local model, resulting in a potential conflict between personalization and generalization. To address this issue, we propose a parallel structure of a local supervisor and an inter-learning model for the local model and introduce a novel pFL method called federated learning by considering data similarity across clients assisted by a local supervisor (FedSimSup). Specifically, FedSimSup maintains an inter-learning model for each client and refines the inter-learning model using a local supervisor for each client. The local supervisor monitors the aggregated global information and ensures that the inter-learning model aligns with the local heterogeneous data to enhance local model performance. Additionally, the similarity between clients is measured based on differences in local data distributions, and this similarity is used to adjust the weights of the inter-learning models. Experimental results show that FedSimSup outperforms eight state-of-the-art federated learning methods in handling heterogeneous data. Additionally, it supports different model architectures across clients, providing greater flexibility when computational resources vary among them. Our code can be found at https://github.com/jqLi1626/FedSimSup.

1. Introduction

With the increasing importance of data privacy in the digital era, federated learning (FL) has emerged as a response to the growing need for data in artificial intelligence [9]. FL aims to maximize the use of local data while maintaining privacy and minimizing communication costs by training a global machine learning model. Despite its widespread use, the traditional federated learning method FedAvg [33] often suffers from performance degradation and slow convergence due to data heterogeneity [18, 34]. This issue is prevalent in real-world applications, where data gathered from different sources, such as users, devices, and organizations, typically exhibit distributional shifts (Non-IID data).

Personalized federated learning (pFL) [43] has emerged as an effective solution for handling Non-IID data. Mainstream pFL methods can be broadly divided into two groups, one focusing on training a global model that generalizes well across all clients, while the other on training personalized models for each client to better address data heterogeneity. For instance, the algorithms reported in [2, 10, 30] construct client-specific models by dividing the model for each client into a feature extractor (the backbone) and a classifier. The backbone serves as the shared component to capture generalization information across clients, while the classifier focuses on learning personalized information. Motivated by the observation that the decoupling approach, which extracts information from model parameters, may not fully exploit all the potential of the data, Zhang *et al*. [51] propose generating a conditional policy for each sample to separate global and personalized information in its characteristics. Furthermore, it has been found that retaining all layers sensitive to Non-IID data for extracting personalization information can degrade the collaborative effect. To address this, Wu *et al*. [47] propose a sensitivity-

[*] Equal contribution.
† Corresponding author

based quantitative metric to assess each parameter and identify those most sensitive for personalization. Recently, Yang *et al*. [50] tackle the inter-client and intra-client inconsistency between personalized and shared components by introducing a federated parameter-alignment and client-synchronization method, which shows promising results.

The aforementioned parameter decoupling method all adopt a serial structure, where information of the backbone and classifier is processed sequentially. However, the two modules are often dependent on each other's performance. If the backbone is suboptimal, it can significantly affect the classifier's ability to function well. Additionally, a serial structure often requires a unified model to handle data heterogeneity, which, however, may lead to suboptimal results when the data distributions are highly different. More importantly, information is processed step-by-step in serial structures. Thus, there is a potential for information loss or degradation, especially when one component (like the backbone) overshadows or fails to convey crucial details that would benefit the classifier.

Therefore, we propose setting up a parallel structure of a supervisor and an inter-learning model for personalization and generalization, respectively. Our proposed algorithm is termed **FedSimSup** (Federated learning by considering Similarity across clients under a local Supervisor). The contributions of our work are summarized as follows.

- We propose a novel supervisor-assisted pFL framework. Each client is assigned a local unique supervisor to monitor the information contained in the aggregated inter-learning model received from the server. By utilizing a parallel structure, each part of the model is better able to preserve the original input data. The two branches ensure that distinct features or characteristics of the data are captured independently, thereby minimizing the risk of inaccurate final predictions that can arise from inconsistencies between different components, as seen in a serial structure.
- We propose leveraging the local data distribution of each client to enhance model training. By evaluating the similarity based on the differences in local data distribution between clients, each client can selectively learn from others. Specifically, we maintain an inter-learning model on the server for each client. If a client does not participate in a particular round of communication, we aggregate the inter-learning models of that client and others, using similarity values to update the client's inter-learning model.
- We conduct experiments by adopting different supervisor architectures across various clients. Compared to the serial structure, the parallel architecture of our FedSimSup enables clients to build supervisor architectures tailored to their specific needs and computational capabilities, offering greater flexibility when computational resources vary across clients.

2. Related Work

Personalized Federated Learning is an effective way to address data heterogeneous settings. Existing methods can generally be categorized into several types. First, *data augmentation* [13, 20, 40] aims to reduce data heterogeneity, enabling the use of standard FL to address the problem. Following this, *regularization* [1, 16, 26, 42] prevents client overfitting and accelerates global convergence, enhancing the overall robustness of the model. Additionally, *meta learning* [14, 21, 38] enables the global model to achieve personalization more quickly on the client side. Furthermore, *multi-task learning* [19, 41] treats each client as a different task and leverages relationships between them to handle heterogeneous settings. Moreover, *clustering* [3, 15, 37] divides clients into different homogeneous groups, within which FL is performed more effectively. *Knowledge distillation* [22, 25, 45, 48] transfers knowledge from the server or other clients to a specific client, ensuring that each client benefits from shared insights. *Model interpolation* learns personalized models by combining local models with the global model, thus balancing the model's generalization and personalization capabilities. Along this line, Hanzely *et al*. [16] design a new objective function that incorporates a penalty term with a coefficient. Deng *et al*. [12] propose a method to find an optimal combination of local and global models, aiming to enhance model performance under diverse client data distributions. Chen *et al*. [6] propose elastic aggregation, which performs adaptive interpolation based on the sensitivity of the model parameters, allowing for dynamic adjustments according to the specific needs of each client.

Parameter decoupling refers to separating the model's parameters and implementing stepwise training, with one set of parameters being globally shared and another set trained locally, thereby enhancing the personalization capability. There are several main decoupling methods. The first method divides the network into base layers and personalized layers [2, 31, 49], with the base layers being globally shared to obtain the generalized feature information, while the personalized layers are trained only locally to allow different clients to process the features in their own ways. The second method uses embeddings from each client as personalization layers [4, 30], aiming to extract unique features to be processed by the global model. Other methods, like [28], propose FedRAP, which learns a global view and a personalized view locally on each client to achieve personalization. Parameter decoupling reduces the amount of transmitted parameters, thereby decreasing communication overhead to some extent. Although parameter decoupling has demonstrated its effectiveness in multiple aspects, it still faces challenges in handling scenarios with extreme

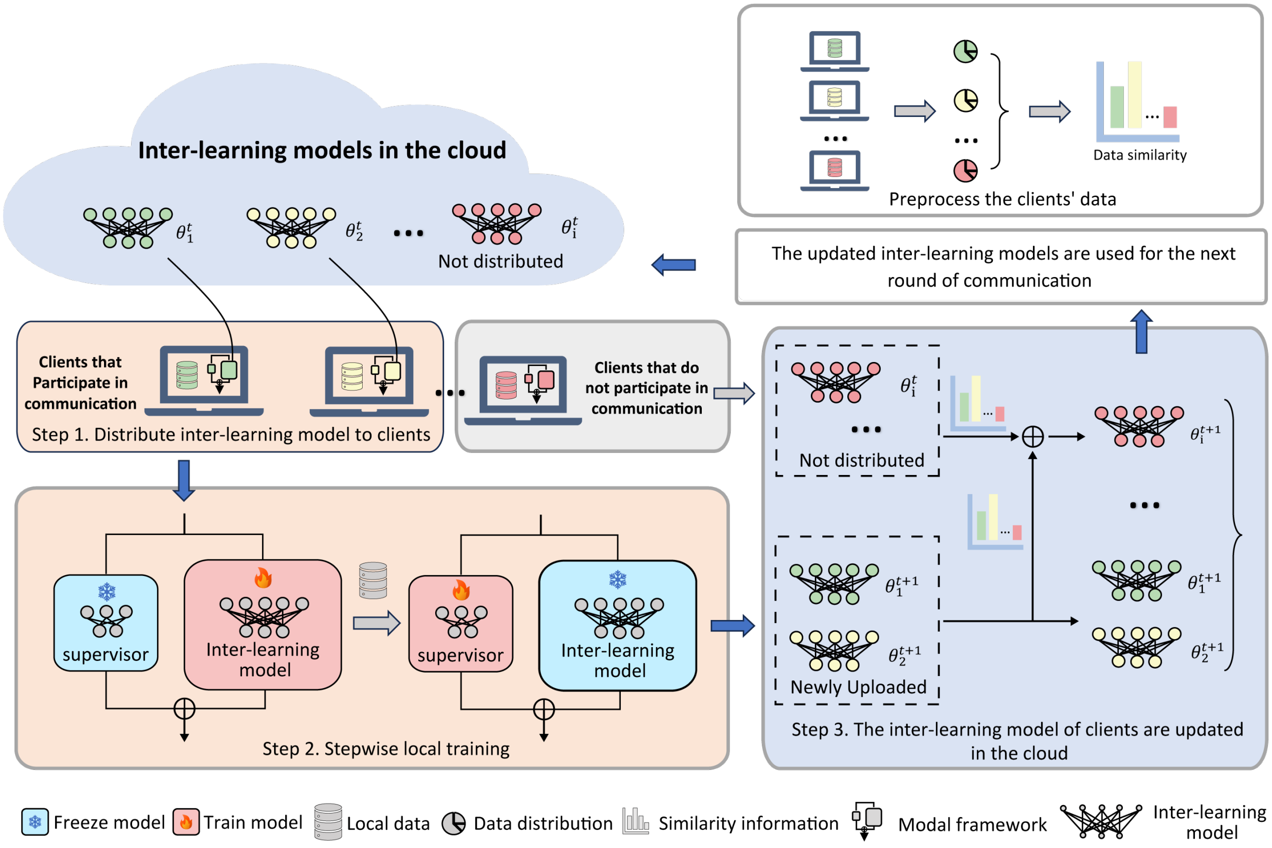

Inter-learning models in the cloud
θ_1^t
θ_2^t
θ_i^t
Not distributed
Preprocess the clients' data
Data similarity
The updated inter-learning models are used for the next round of communication
Clients that Participate in communication
Step 1. Distribute inter-learning model to clients
Clients that do not participate in communication
Not distributed
θ_i^{t+1}
θ_1^{t+1}
θ_2^{t+1}
Newly Uploaded
Step 3. The inter-learning model of clients are updated in the cloud
supervisor
Inter-learning model
supervisor
Inter-learning model
Step 2. Stepwise local training
Freeze model
Train model
Local data
Data distribution
Similarity information
Modal framework
Inter-learning model

Figure 1. The main framework of FedSimSup. During each communication round, the server distributes the corresponding inter-learning models to the participating clients. These clients train their inter-learning models under the supervision of the local supervisor. Once the communication concludes, the inter-learning models are uploaded to the server, while the non-participating clients aggregate their inter-learning model with the trained inter-learning model based on similarity information.

data heterogeneity. Future research could explore more efficient decoupling strategies to optimize the performance of federated learning.

As mentioned above, due to data heterogeneity, each client's model is complemented by the data of other clients with different weights. As a subfield of pFL, there is also a line of research that manages to address potential conflicts of interest between competitive clients when FL is adopted to provide personalized models to clients [7, 8, 29, 44, 46, 52].

3. Method

3.1. Problem Formulation

In this work, we assume supervised federated learning with a total of n clients, each having its own Non-IID distributed dataset $D_i = \left\{\left(x_1^i, y_1^i\right), \left(x_2^i, y_2^i\right), \ldots, \left(x_{m_i}^i, y_{m_i}^i\right)\right\} \subset \mathcal{X} \times \mathcal{Y}$, for $i \in \{1, 2, \ldots, n\}$, where m_i is the amount of data for client i. We specifically focus on statistical heterogeneity, i.e., the differences in data distributions across clients in this work. To model this, we use Dirichlet [17] and Pathological distributions [33], which are commonly employed to simulate Non-IID data scenarios (detailed partitioning methods are provided in Sec. 7.2 in the Appendix).

The model of each client $q_{\theta_i} : \mathcal{X} \to \mathcal{Y}$ maps the input $x_j^i \in \mathcal{X}$ to predict the label $q_{\theta_i}(x_j^i) \in \mathcal{Y}$, which is compared to the true label corresponding to $y_j^i \in \mathcal{Y}$. The local models q_{θ_i} are the same in standard FL, but differ in pFL. $\theta_i \in \Theta$ represents the model parameters. The parameters of each client's model θ_i are trained based on its local dataset by minimizing the following objective function:

$$\min_{\theta_i \in \Theta} L(D_i, \theta_i) = \frac{1}{m_i} \sum_{j=1}^{m_i} \ell\left(q_{\theta_i}(x_j^i), y_j^i\right), \tag{1}$$

where $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ is the loss function that measures the degree of inconsistency between the predicted labels $q_{\theta_i}(x_j^i)$ and the true labels y_j^i.

The server distributes the model to clients, who then train the model locally using the local objective function Eq. (1). After training, the clients upload their models to the server for aggregation [33]:

$$\theta^{t+1} = \frac{\sum_{i \in \mathcal{N}(t)} m_i \theta_i^t}{\sum_{i \in \mathcal{N}(t)} m_i}, \tag{2}$$

where θ_i^t is the model of client i after completing local training in the t-th communication round, and $\mathcal{N}(t)$ denotes the set of clients participating in the t-th communication round.

Standard FL struggles with slow convergence and suboptimal model performance when dealing with Non-IID data across clients. Given that pFL has proven effective in addressing this challenge by learning personalized models for each client, we build upon the pFL framework and present a new, highly effective solution for handling statistically heterogeneous data, called FedSimSup. The details of our approach are outlined in the following sections.

3.2. Learning under Supervisor

In this work, instead of using a serial model as in existing pFL approaches, we divide the model into two parallel parts. The first part is the supervisor, which is trained locally but not uploaded. The second part is the inter-learning model, which is uploaded and aggregated. The role of the supervisor is to guide the local model during training by providing oversight based on the previously learned local data. It helps prevent the model from deviating too much from its fit to the local data while still incorporating beneficial global updates. The supervisor ensures that the inter-learning model maintains a balance between leveraging global information and staying aligned with the local distribution.

The local objective function also changes from Eq. (1) to

$$\begin{aligned}&\min_{s_i\in\mathcal{S},\ \theta_i\in\Theta} L\left(D_i, s_i, \theta_i\right)\\&=\frac{1}{m_i}\sum\nolimits_{j=1}^{m_i}\ell\left(q_{\theta_i}\left(x_j^i\right)+q_{s_i}\left(x_j^i\right), y_j^i\right),\end{aligned} \tag{3}$$

where $s_i \in \mathcal{S}$ is the parameters of supervisor and $\theta_i \in \Theta$ is the parameters of inter-learning model. Here, we simply sum the results of the two models. The training process of our model is divided into two parts:

$$\min_{s_i\in\mathcal{S}} L\left(D_i, s_i, \theta_i\right), \tag{4}$$

$$\min_{\theta_i\in\Theta} L\left(D_i, s_i, \theta_i\right). \tag{5}$$

The purpose of Eq. (4) is that if the global model contains more beneficial information, the supervisor will undergo a significant update to better assist the training process. However, if the global model is not beneficial to the local data, the supervisor has already been fitted to the local data, we hypothesize that it will undergo only minor updates or remain unchanged. The purpose of Eq. (5) is to train the model under the supervision of the supervisor, ensuring that after acquiring global information, it becomes more fitted to the local data.

For demonstration purposes, we directly scale down the inter-learning model proportionally to create the supervisor, which then assists the inter-learning model in its usage. In practice, the architecture of the supervisor does not need to be the same for every client. Each client can independently design their own supervisor architecture according to their specific needs and capabilities. The server only needs to manage the inter-learning model but not the whole local model. This approach significantly enhances the personalization capability of the model while simplifying management. We demonstrate the performance results when clients adopt different supervisor structures in Sec. 5.2. The effect of the supervisor is verified in Sec. 7.5.

3.3. Utilization of Similarity Information

We establish an inter-learning model for each client based on their local data distribution. At the end of a local training round, we perform the following operations on all clients' inter-learning models (the following operations are performed on the inter-learning model, unrelated to the supervisor).

If a client i participates in this round of communication, then the inter-learning model θ_i^{t+1} of the i-th client at $t+1$ round is set to the updated θ_i^t after training without aggregating information from other clients

$$\theta_i^{t+1}=\theta_i^t, \qquad \text{if } i\in\mathcal{N}\left(t\right). \tag{6}$$

If the client i does not participate in this round of communication, then θ_i^{t+1} of the i-th client is updated as follows.

$$\theta_i^{t+1}=\left(1-\alpha_i^t\right)\theta_i^t+\alpha_i^t\sum_{j\in\mathcal{N}(t)}\frac{s_{ij}}{\sum_{j\in\mathcal{N}(t)}s_{ij}}\theta_j^t, \text{if } i\notin\mathcal{N}(t), \tag{7}$$

$$\alpha_i^t=\lambda_i^t\beta^t, \tag{8}$$

$$\lambda_i^t=\frac{\sum_{j\in\mathcal{N}(t)}m_j}{\sum_{j\in\mathcal{N}(t)}m_j+K\cdot m_i}, \tag{9}$$

$$\beta^t=\begin{cases}1, & t<CT^\gamma\\ \left(\frac{CT^\gamma}{t}\right)^2, & t\geq CT^\gamma,\end{cases} \tag{10}$$

where α_i^t is a parameter that represents how much the current client is to learn from the clients participating in the t-th communication. It is related to the volume of data and the number of communication rounds. λ_i^t measures the amount of data, calculated based on the ratio of the amount of local data to the total amount of data of clients participating in the t-th communication, which aligns with the original standard FL concept. K is the number of clients participating in communication in each round. β^t is influenced by the current number of communications rounds. In the initial stages, it is set to 1, in which case α_i^t will only be affected by the amount of data. In later stages, to ensure convergence, β^t gradually decreases under the control of parameters C

Algorithm 1 FedSimSup

```
Input: Dataset distributed across $n$ clients $D = \{D_1, D_2 \cdots D_n\}$, client participating rate $r$, the global communication rounds $T$, inter-learning model epochs $\tau_\theta$, supervisor epochs $\tau_s$
Initialize $\theta_1^0, \theta_2^0 \cdots \theta_n^0, s_1^0, s_2^0 \cdots s_n^0$
for $t = 1, 2 \cdots T$ do
    $\mathcal{N}(t) \leftarrow$ Server randomly samples $max(1, nr)$ clients
    for Each client $i \in \mathcal{N}(t)$ do
        Client $i$ initializes $s_i^{t,0} \leftarrow s_i^{t-1,\tau_s}$
        Server sends $\theta_i^{t-1,\tau_\theta}$ to client $i$ as $\theta_i^{t,0}$
        $s_i^{t,\tau_s}, \theta_i^{t,\tau_\theta} \leftarrow$ LocalUpdate($s_i^{t,0}, \theta_i^{t,0}, f_i, D_i$)
        Client $i$ sends updated inter-learning model $\theta_i^{t,end}$ to server
    end for
    for Each client $i \notin \mathcal{N}(t)$ do
        Set $s_i^{t,\tau_s} \leftarrow s_i^{t-1,\tau_s}$
        Aggregate $\theta_i^{t,\tau_\theta}$ by Eq. (7)
    end for
end for

LocalUpdate($s^0, \theta^0, f, D$):
for $j = 1, 2 \cdots \tau_s$ do
    $s^j \leftarrow SGD\left(f(s^{j-1}, \theta^0), s^{j-1}\right)$
end for
for $j = 1, 2 \cdots \tau_\theta$ do
    $\theta^j \leftarrow SGD\left(f(s^{\tau_s}, \theta^{j-1}), \theta^{j-1}\right)$
end for
return $s^{\tau_s}, \theta^{\tau_\theta}$
```

and γ, where C and γ are constant values and T denotes global communication rounds. $s_{ij} \in [0, 1]$ is the value that measures the similarity between the client i and the client j. A higher value of s_{ij} indicates a greater similarity between client i and j. In Eq. (7), we aggregate the inter-learning models of clients who do not participate in communication, based on the volume of data, the similarity to the clients participating in communication, and the current number of rounds of communication. By doing this, we can ensure that clients who do not participate in the training in each round can still benefit from the clients who participate in the training, thereby promoting effective global information aggregation. In this work, the similarity information is represented by the cosine similarity between the proportions of each client's data-label distribution, which we believe better reflects the intrinsic similarity between clients. This requires us to collect the proportions of the label of the clients at the beginning of the entire task and to compute the similarity between each client on the server, as shown in Fig. 1.

3.4. FedSimSup Algorithm

We provide the pseudocode for FedSimSup in Algorithm 1, and below we will explain it in detail.

Local Update. In each communication round, clients are randomly selected to participate based on a fixed participation rate r and receive the inter-learning model θ sent by the server. Client i participates in the t-th round, receives the inter-learning model θ_i^t, and has a supervisor s_i^t stored locally. The local supervisor is updated for τ_s epochs.

$$s_i^{t,j} \leftarrow SGD\left(f(s_i^{t,j-1}, \theta_i^{t,0}), s_i^{t,j-1}\right), \tag{11}$$

where $j \in (1, 2, \cdots \tau_s)$, and $\theta_i^{t,0}$ denotes the inter-learning model of client i that has not been updated. we use Stochastic Gradient Descent (SGD) [36] to update $s_i^{t,j}$ based on the gradient of $s_i^{t,j}$. Then, the inter-learning model is updated within round τ_θ:

$$\theta_i^{t,j} \leftarrow SGD\left(f(s_i^{t,\tau_s}, \theta_i^{t,j-1}), \theta_i^{t,j-1}\right), \tag{12}$$

where $j \in (1, 2, \cdots \tau_\theta)$. After completing these two processes locally, the client i saves the supervisor s_i^{t,τ_s} and uploads the inter-learning model θ_i^{t,τ_θ} for aggregation of other clients.

Server Update The server receives the inter-learning models uploaded from client set $\mathcal{N}(t)$, without modifying them. For clients who did not participate in the communication, it aggregates their models based on Eq. (2), leveraging similarity information to learn from the clients that have participated in this round of training.

4. Convergence Analysis of FedSimSup

4.1. Assumptions

We make the following standard assumptions that are widely used in convergence analysis in federated learning [14, 16, 26, 27, 35].

Assumption 1 (Bounded Loss)**.** *There exists constant $F^* \in \mathbb{R}$ such that for any client $i \in \{1, \ldots, n\}$, f_i is bounded from below by F^*, $f_i(s, \theta) > F^*, \forall s, \theta$.*

Assumption 2 (Smoothness)**.** *There exists $L > 0$ such that for any client $i \in \{1, \ldots, n\}$, $\nabla_s f_i(\cdot, \theta)$, $\nabla_s f_i(s, \cdot)$, $\nabla_\theta f_i(\cdot, \theta)$ and $\nabla_\theta f_i(s, \cdot)$ are L-Lipschitz.*

Assumption 3 (Bounded Gradient)**.** *For all $i \in \{1, \ldots, n\}$, the gradient of loss function f_i is bounded. There exists $G > 0$ such that*

$$\|\nabla_s f_i(s, \theta)\| \leq G, \quad \|\nabla_\theta f_i(s, \theta)\| \leq G, \quad \forall s, \theta. \tag{13}$$

Assumption 4 (Unbiasedness)**.** *SGD estimator is unbiased. There exists $\sigma > 0$ such that for any client $i \in \{1, \ldots, n\}$,*

$$\begin{aligned} \mathbb{E}[SGD(f_i(s, \theta), s)] &= \nabla_s f_i(s, \theta), \quad \forall s, \theta, \\ \mathbb{E}[SGD(f_i(s, \theta), \theta)] &= \nabla_\theta f_i(s, \theta), \quad \forall s, \theta. \end{aligned} \tag{14}$$

Assumption 5 (Bounded Variance)**.** *The variance of SGD estimator is bounded. That is, for any client $i \in \{1, \dots, n\}$,*

$$\mathbb{E}\Big[\|SGD(f_i(s,\theta), s) - \nabla_s f_i(s,\theta)\|^2\Big] \leq \sigma^2, \forall s, \theta. \quad (15)$$

$$\mathbb{E}\Big[\|SGD(f_i(s,\theta), \theta) - \nabla_\theta f_i(s,\theta)\|^2\Big] \leq \sigma^2, \forall s, \theta. \quad (16)$$

4.2. Main Theorem

With the above assumptions in Sec. 4.1, we present the convergence of the proposed FedSimSup. The proof of the theorem is given in Sec. 8.

Theorem 1 (Convergence of FedSimSup)**.** *Suppose Assumptions 1 to 5 hold, and the learning rates in FedSimSup are chosen as*

$$\eta_s^t = \eta/\sqrt{T}L\tau_s, \quad \eta_\theta^t = \eta/\sqrt{T}L\tau_\theta \quad (17)$$

with $\eta < 1$. The hyperparameters of FedSimSup are chosen as $\gamma \in (0, 1/2)$ and $C > 0$. Then we have the following bound for all client i.

$$\begin{aligned} &\frac{1}{T}\sum_{i=0}^{T-1}\left[\left\|\nabla_s f_i(s_i^{t,0}, \theta_i^{t,0})\right\|^2 + 2\left\|\nabla_\theta f_i(s_i^{t,0}, \theta_i^{t,0})\right\|^2\right] \\ &\leq \frac{L}{2T^{1/2}r\eta}(f_i(s_i^{0,0}, \theta_i^{0,0}) - F^*) + \frac{3\eta^2 G^2}{2T} + \frac{\eta\sigma^2}{T^{1/2}\bar{\tau}} \\ &\quad + \frac{(1-r)\lambda_i G^2}{2r}\Big(-C^2(1-2\gamma)T^{2\gamma-1}\ln T \\ &\quad + C^2(1+2\ln C)T^{2\gamma-1} + 3CT^{\gamma-1} + 1/T - C^2T^{2\gamma-2}\Big) \\ &\quad + \frac{(1-r)\lambda_i^2\eta^2 G^2}{2rL}\Big(3CT^{3\gamma-3/2} + 9C^2T^{2\gamma-3/2} \\ &\quad + 3CT^{\gamma-3/2} - 3C^4T^{4\gamma-9/2}\Big). \end{aligned} \quad (18)$$

Here, $\bar{\tau} = 2/(1/\tau_s + 1/\tau_\theta)$ and $\lambda_i = \max_{|\mathcal{N}| \subset \{1,\dots,n\}}(\sum_{j\in\mathcal{N}} m_j)/(\sum_{j\in\mathcal{N}} m_j + Km_i)$.

The left-hand side of Eq. (18) is the average over time of a weighted sum of the gradient norm. Convergence is measured in the rate at which this quantity decays to zero. We note that Theorem 1 and Fig. 2 together demonstrate a trade-off between generalization and convergence speed. As shown in Fig. 9 in the Appendix, larger values of C and γ encourage inter-learning models to incorporate more global information, enhancing their generalization performance. In contrast, smaller values of C and γ direct the local models to prioritize private data, leading to faster convergence. This highlights the need for careful hyperparameter selection. Additionally, clients with more private data (smaller λ_i) require less global information and therefore achieve faster convergence.

5. Experiments

5.1. Experimental Settings

Datasets. We evaluate FedSimSup on classification tasks using CIFAR10, CIFAR100 [23], FEMNIST [5], and IMAGENET [11]. For CIFAR10 and CIFAR100, we simulate a Non-IID setting by partitioning data based on a Dirichlet distribution with $\alpha = 0.1$ and $\alpha = 0.5$, where a lower α indicates a greater heterogeneity. FEMNIST and IMAGENET are partitioned using a Dirichlet distribution with $\alpha = 0.1$. To our knowledge, this is the first evaluation of pFLs on IMAGENET. We compare the performance of FedSimSup and competing algorithms on CIFAR10, CIFAR100, and IMAGENET under the Dirichlet distribution, as well as under the Pathological distribution for CIFAR10 and CIFAR100. Further details on data partitioning and the datasets are provided in Sec. 7.1 and Sec. 7.2 in the Appendix.

Baselines. We compare FedSimSup with eight state-of-the-art methods, including FedAvg [33], FedProx [26], Per-FedAvg [14], FedRep [10], FedProto [45], FedPac [48], pFedFda [32] and FedAs [50]. Additionally, we also compare our FedSimSup with the performance of conducting local training separately on each client. Introduction of the baselines, experimental settings of baseline, and training details are provided in Sec. 7.3.

Model. Like most pFL approaches, FedSimSup uses LeNet-5 [24] as the local model for each client, considering the communication cost. LeNet-5 consists of two convolutional layers and two linear layers. In fairness, we use LeNet-5 as the model for all the algorithms in this work. Since our FedSimSup includes both a supervisor and an inter-learning model in each client. Thus, to ensure that the number of parameters of FedSimSup is almost the same as that of competing algorithms, we proportionally reduce the size of LeNet-5 to approximately one-sixth of that of the inter-learning model.

5.2. Experimental Results

Performance comparison. On the FEMNIST and IMAGENET datasets under a Dirichlet distribution, as shown in Tab. 1, FedSimSup consistently outperforms all other methods on both datasets. We observe that FedAvg performs surprisingly well on the FEMNIST dataset, likely because, despite the non-IID nature of the data, clients share common visual patterns such as digits and letters. Tab. 2 and Tab. 3 show the performance comparison in the CIFAR10 and CIFAR100 datasets under the Dirichlet distribution and the Pathological distribution, respectively. In the Dirichlet distribution, our FedSimSup method consistently outperforms all other methods across both the CIFAR10 and CIFAR100 datasets, achieving the highest accuracy in all cases. In the Pathological distribution, although the accuracy of all meth-

	Local	FedAvg	FedProx	Per-FedAvg	FedRep	FedProto	FedPac	pFedFda	FedAs	FedSimSup
FEMNIST	.660(.003)	.797(.023)	.742(.017)	.025(.04)	.812(.027)	.099(.06)	.782(.014)	.773(.004)	.764(.006)	**.843(.006)**
IMAGENET	.133(.002)	.094(.033)	.089(.024)	.013(.005)	.209(.007)	.127(.013)	.132(.004)	.212(.005)	.211(.003)	**.224(.008)**

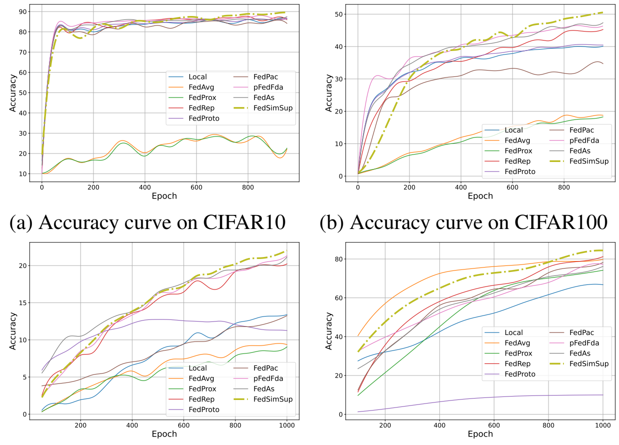
Table 1. Accuracy comparison on FEMNIST and IMAGENET under Dirichlet distribution with $\alpha = 0.1$ (best valued per setup in bold).

	CIFAR10				CIFAR100			
No. of Clients (Dir)	100 (0.1)	50 (0.1)	100 (0.5)	50 (0.5)	100 (0.1)	50 (0.1)	100 (0.5)	50 (0.5)
Local	.863(.001)	.853(.030)	.588(.004)	.611(.003)	.399(.002)	.415(.006)	.175(.003)	.209(.002)
FedAvg (2017)	.276(.026)	.341(.058)	.484(.015)	.506(.028)	.191(.006)	.189(.009)	.224(.005)	.252(.010)
FedProx (2020)	.275(.023)	.317(.039)	.488(.011)	.509(.028)	.184(.006)	.184(.008)	.210(.007)	.244(.008)
Per-FedAvg (2020)	.774(.005)	.750(.024)	.368(.006)	.464(.015)	.038(.001)	.103(.002)	.016(.001)	.003(.001)
FedRep (2021)	.875(.009)	.840(.053)	.706(.011)	.718(.015)	.454(.008)	.497(.020)	.245(.009)	.312(.011)
FedProto (2022)	.865(.001)	.828(.030)	.592(.004)	.612(.002)	.407(.002)	.424(.003)	.173(.002)	.213(.002)
FedPac (2023)	.839(.014)	.787(.047)	.599(.017)	.603(.033)	.348(.006)	.398(.023)	.178(.008)	.230(.006)
pFedFda (2025)	.878(.006)	.867(.005)	.703(.009)	.714(.005)	.463(.008)	.509(.005)	.307(.006)	.332(.009)
FedAs (2024)	.874(.003)	.865(.006)	.712(.004)	.720(.004)	.467(.005)	.512(.004)	.316(.005)	.342(.008)
FedSimSup (Ours)	**.892(.002)**	**.882(.004)**	**.725(.005)**	**.736(.006)**	**.503(.004)**	**.546(.003)**	**.331(.006)**	**.385(.005)**

Table 2. Accuracy comparison on CIFAR10 and CIFAR 100 under Dirichlet distribution (best valued per setup in bold).

ods is generally lower compared to the Dirichlet case, our FedSimSup still leads the performance. Methods like FedRep and FedProto show moderate performance, but are still outperformed by newer methods like FedAs (2024) and pFedFda (2025) in some cases. These results underscore the superior effectiveness of FedSimSup in federated learning tasks, particularly in the Dirichlet distribution setup.

We also compare the convergence speeds of different methods. In Fig. 2, we present the accuracy changes of various methods under Non-IID distributions for CIFAR10 and CIFAR100, using the Dir (0.1) setting over 1000 communication rounds. On the CIFAR10 dataset, all methods exhibit faster convergence, except for FedAvg and FedProx. Notably, our proposed FedSimSup achieves the best performance. In the more challenging CIFAR100 task, which involves a larger number of categories, FedSimSup shows a slower initial improvement compared to other methods. However, it eventually catches up and surpasses the others, highlighting its robust learning capabilities, particularly in settings with a higher number of categories. We provide the accuracy curves on CIFAR10 and CIFAR 100 under all Non-IID settings in Fig. 7 and Fig. 8 in the Appendix.



(a) Accuracy curve on CIFAR10

(b) Accuracy curve on CIFAR100

(c) Accuracy curve on IMAGENET

(d) Accuracy curve on FEMNIST

Figure 2. Accuracy curve along global training rounds.

Different Supervisor Architectures. To demonstrate the performance of FedSimSup with different supervisor architectures across various clients, we allow each client to randomly select one of three supervisor architectures: a Transformer, a CNN, or LeNet-5. This variant of FedSimSup, where clients use different supervisor architectures, is termed FedSimSup-TCL. We refer to the average accuracy achieved by the subset of clients using the Transformer, CNN, and LeNet-5 as FedSimSup-T, FedSimSup-C, and FedSimSup-L, respectively.

Tab. 4 presents a performance comparison between FedSimSup and FedSimSup-TCL. We use a pre-trained Transformer, which explains the competitive performance of FedSimSup-T. We also note that the overall performance of FedSimSup-TCL is slightly lower than that of FedSimSup, which is an inevitable consequence of model heterogeneity. However, the performance gap is not substantial, and some clients achieved better results by selecting models better suited to their individual needs. Therefore, we conclude that our proposed FedSimSup is flexible enough to accommodate different model architectures for different clients based on their computational resources and requirements, enabling them to achieve improved performance and faster inference.

Ablation Studies. To demonstrate that similarity-based model aggregation can accelerate the convergence speed, we replace Eq. (7) with a direct model average as in Fe-

	CIFAR10				CIFAR100			
No. of Clients (Shard)	100 (2)	50 (2)	100 (5)	50 (5)	100 (5)	50 (5)	100 (20)	50 (20)
Local	.859(.001)	**.881(.001)**	.646(.004)	.681(.002)	**.664(.002)**	.647(.003)	.275(.002)	.340(.006)
FedAvg (2017)	.348(.024)	.312(.038)	.484(.021)	.478(.033)	.119(.007)	.124(.012)	.192(.006)	.294(.011)
FedProx (2020)	.340(.021)	.295(.032)	.483(.019)	.481(.031)	.111(.008)	.117(.011)	.184(.007)	.184(.012)
Per-FedAvg (2020)	.515(.001)	.675(.012)	.290(.001)	.476(.050)	.107(.001)	.242(.023)	.064(.001)	.069(.002)
FedRep (2021)	.858(.004)	.871(.009)	.735(.008)	.759(.010)	.614(.010)	.657(.016)	.380(.007)	.450(.010)
FedProto (2022)	.858(.001)	.876(.001)	.638(.002)	.668(.001)	.652(.003)	.642(.003)	.276(.003)	.328(.002)
FedPac (2023)	.839(.016)	.857(.015)	.648(.014)	.658(.002)	.513(.014)	.572(.018)	.201(.004)	.330(.009)
pFedFda (2025)	.862(.007)	.871(.009)	.746(.003)	**.763(.005)**	.642(.007)	**.683(.002)**	.443(.003)	.440(.005)
FedAs (2024)	.853(.004)	.881(.007)	.742(.002)	.760(.006)	.630(.002)	.643(.005)	.427(.005)	.442(.004)
FedSimSup (Ours)	**.866(.006)**	.877(.008)	**.757(.005)**	.757(.002)	.637(.005)	.652(.004)	**.464(.006)**	**.480(.007)**

Table 3. Accuracy on CIFAR10 and CIFAR 100 under Pathological distribution (best valued per setup in bold).

	CIFAR10				CIFAR100			
clients num (Dir)	100 (0.1)	50 (0.1)	100 (0.5)	50 (0.5)	100 (0.1)	50 (0.1)	100 (0.5)	50 (0.5)
FedSimSup-T	**.898(.006)**	**.908(.003)**	.693(.008)	.721(.007)	**.504(.003)**	**.548(.004)**	.293(.007)	.354(.005)
FedSimSup-C	.835(.004)	.864(.005)	.680(.008)	.707(.006)	.432(.005)	.437(.004)	.244(.008)	.296(.003)
FedSimSup-L	.851(.007)	.843(.004)	.664(.012)	.693(.004)	.445(.003)	.465(.007)	.227(.003)	.320(.001)
FedSimSup-TCL	.861(.005)	.871(.005)	.679(.009)	.707(.005)	.460(.003)	.482(.006)	.254(.007)	.323(.004)
FedSimSup	.892(.002)	.882(.004)	**.725(.005)**	**.736(.006)**	.503(.004)	.546(.003)	**.331(.006)**	**.385(.005)**

Table 4. Experiments of using different supervisor architectures.

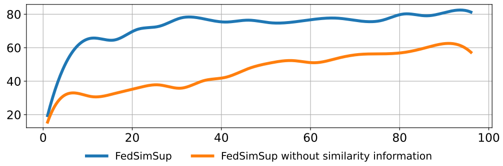


Figure 3. The accuracy with and without similarity information on CIFAR10

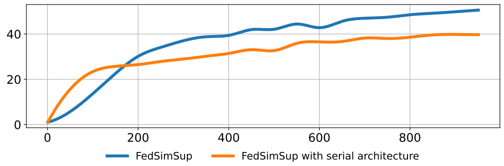


Figure 4. The accuracy with and without parallel architecture on CIFAR100

dAvg. In Fig. 3, we compare the impact of using similarity information versus not using it on the convergence speed. The experiment is carried out on CIFAR10 in Dirichlet distribution with $\alpha = 0.1$, and we display the results for the first 100 epochs. The results show that the use of similarity information accelerates the convergence speed, demonstrating the effectiveness of our proposed similarity measurement and aggregation strategy. Furthermore, to demonstrate the effectiveness of the parallel structure in FedSimSup, we replace the parallel structure with a serial model, as in FedRep [10]. Specifically, we adopt the inter-learning model on each client and increase the number of parameters to ensure fairness. Only the backbone of the model is transmitted between the server and the clients. We conduct experiments on CIFAR100 under Dirichlet distribution with $\alpha = 0.1$. As shown in Fig. 4, the performance was subpar, demonstrating the effectiveness of our parallel architecture. More detailed ablation experiments are provided in Sec. 7.4 in the Appendix.

Parameter Sensitivity Analyses. In the model aggregation process of FedSimSup (see Eq. (7)), two critical parameters, C and γ, are involved. These parameters work together to control α_i^t in Eq. (7), which determines how much the current client i learns from other clients. To analyze their impact on performance, we evaluate four pairs of C and γ. Specifically, we set C and γ to (10, 1/3), (20, 2/5), (40, 2/5), and (40, 3/7). The results on CIFAR10 and CIFAR100, shown in Fig. 9 in the Appendix, demonstrate that setting C and γ to 40 and 3/7 yields the best performance.

6. Conclusion

Our proposed FedSimSup allows each client to employ their own supervisor with flexible architectures to assist local training, preventing the model from deviating too much from the local data. Additionally, we utilize the similarity information to standardize the way of clients learning from other clients' information. Overall, FedSimSup provides better performance in handling Non-IID scenarios, while allowing clients to customize their model architectures.

Acknowledgments

This work is funded in part by an international Collaboration Fund for Creative Research of National Science Foundation of China (NSFC ICFCRT) under the Grant no. W2441019. This work is also supported in part by the National Natural Science Foundation of China under Grant No. 62302147 and in part by Selected Funding Program for Zhejiang Province Postdoctoral Research Projects.

References

[1] Durmus Alp Emre Acar, Yue Zhao, Ramon Matas Navarro, Matthew Mattina, Paul N Whatmough, and Venkatesh Saligrama. Federated learning based on dynamic regularization. *arXiv preprint arXiv:2111.04263*, 2021. 2

[2] Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. Federated learning with personalization layers. *arXiv preprint arXiv:1912.00818*, 2019. 1, 2

[3] Christopher Briggs, Zhong Fan, and Peter Andras. Federated learning with hierarchical clustering of local updates to improve training on non-IID data. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–9. IEEE, 2020. 2

[4] Duc Bui, Kshitiz Malik, Jack Goetz, Honglei Liu, Seungwhan Moon, Anuj Kumar, and Kang G Shin. Federated user representation learning. *arXiv preprint arXiv:1909.12535*, 2019. 2

[5] Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečnỳ, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018. 6

[6] Dengsheng Chen, Jie Hu, Vince Junkai Tan, Xiaoming Wei, and Enhua Wu. Elastic aggregation for federated optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12187–12197, 2023. 2

[7] Mengmeng Chen, Xiaohu Wu, Qiqi Liu, Tiantian He, Yew-Soon Ong, Yaochu Jin, Qicheng Lao, and Han Yu. Voronoi-grid-based pareto front learning and its application to collaborative federated learning. In *The Forty-second International Conference on Machine Learning*, 2025. 3

[8] Mengmeng Chen, Xiaohu Wu, Xiaoli Tang, Tiantian He, Yew-Soon Ong, Qiqi Liu, Qicheng Lao, and Han Yu. Free-rider and conflict aware collaboration formation for cross-silo federated learning. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*. Curran Associates Inc., 2025. 3

[9] Yong Cheng, Yang Liu, Tianjian Chen, and Qiang Yang. Federated learning for privacy-preserving ai. *Communications of the ACM*, 63(12):33–36, 2020. 1

[10] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. In *International Conference on Machine Learning*, pages 2089–2099. PMLR, 2021. 1, 6, 8, 2

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 6

[12] Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Adaptive personalized federated learning. *arXiv preprint arXiv:2003.13461*, 2020. 2

[13] Moming Duan, Duo Liu, Xianzhang Chen, Yujuan Tan, Jinting Ren, Lei Qiao, and Liang Liang. Astraea: Self-balancing federated learning for improving classification accuracy of mobile deep learning applications. In *2019 IEEE 37th International Conference on Computer Design (ICCD)*, pages 246–254. IEEE, 2019. 2

[14] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. *Advances In Neural Information Processing Systems*, 33:3557–3568, 2020. 2, 5, 6, 1, 3

[15] Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An efficient framework for clustered federated learning. *Advances in Neural Information Processing Systems*, 33:19586–19597, 2020. 2

[16] Filip Hanzely and Peter Richtárik. Federated learning of a mixture of global and local models. *arXiv preprint arXiv:2002.05516*, 2020. 2, 5, 3

[17] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019. 3

[18] Wenke Huang, Mang Ye, Zekun Shi, and Bo Du. Generalizable heterogeneous federated cross-correlation and instance similarity learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(2):712–728, 2023. 1

[19] Yutao Huang, Lingyang Chu, Zirui Zhou, Lanjun Wang, Jiangchuan Liu, Jian Pei, and Yong Zhang. Personalized cross-silo federated learning on non-iid data. In *Proceedings of the AAAI conference on Artificial Intelligence*, pages 7865–7873, 2021. 2

[20] Eunjeong Jeong, Seungeun Oh, Hyesung Kim, Jihong Park, Mehdi Bennis, and Seong-Lyun Kim. Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data. *arXiv preprint arXiv:1811.11479*, 2018. 2

[21] Yihan Jiang, Jakub Konečnỳ, Keith Rush, and Sreeram Kannan. Improving federated learning personalization via model agnostic meta learning. *arXiv preprint arXiv:1909.12488*, 2019. 2

[22] Michael Kamp, Jonas Fischer, and Jilles Vreeken. Federated learning from small datasets. In *Eleventh International Conference on Learning Representations*. OpenReview. net, 2023. 2

[23] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 6

[24] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 6

[25] Daliang Li and Junpu Wang. Fedmd: Heterogenous federated learning via model distillation. *arXiv preprint arXiv:1910.03581*, 2019. 2

[26] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2:429–450, 2020. 2, 5, 6, 1, 3

[27] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. In *International Conference on Learning Representations*. 5, 3

[28] Zhiwei Li, Guodong Long, and Tianyi Zhou. Federated recommendation with additive personalization, 2024. 2

[29] Zhilong Li, Xiaohu Wu, Xiaoli Tang, Tiantian He, Yew-Soon Ong, Mengmeng Chen, Qiqi Liu, Qicheng Lao, and Han Yu. Benchmarking data heterogeneity evaluation approaches for personalized federated learning. In *Federated Learning in the Age of Foundation Models - FL 2024 International Workshops*, pages 77–92, Cham, 2025. Springer Nature Switzerland. 3

[30] Paul Pu Liang, Terrance Liu, Liu Ziyin, Nicholas B Allen, Randy P Auerbach, David Brent, Ruslan Salakhutdinov, and Louis-Philippe Morency. Think locally, act globally: Federated learning with local and global representations. *arXiv preprint arXiv:2001.01523*, 2020. 1, 2

[31] Renpu Liu, Cong Shen, and Jing Yang. Federated representation learning in the under-parameterized regime. *arXiv preprint arXiv:2406.04596*, 2024. 2

[32] Connor Mclaughlin and Lili Su. Personalized federated learning via feature distribution adaptation. *Advances in Neural Information Processing Systems*, 37:77038–77059, 2025. 6, 2

[33] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, 2017. 1, 3, 6

[34] Matias Mendieta, Taojiannan Yang, Pu Wang, Minwoo Lee, Zhengming Ding, and Chen Chen. Local learning matters: Rethinking data heterogeneity in federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8397–8406, 2022. 1

[35] Krishna Pillutla, Kshitiz Malik, Abdel-Rahman Mohamed, Mike Rabbat, Maziar Sanjabi, and Lin Xiao. Federated learning with partial model personalization. In *International Conference on Machine Learning*, pages 17716–17758. PMLR, 2022. 5, 3

[36] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, pages 400–407, 1951. 5

[37] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE Transactions on Neural Networks and Learning Systems*, 32 (8):3710–3722, 2020. 2

[38] Jonathan A Scott, Hossein Zakerinia, and Christoph Lampert. Pefll: Personalized federated learning by learning to learn. In *12th International Conference on Learning Representations*, 2024. 2

[39] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 618–626, 2017. 2

[40] MyungJae Shin, Chihoon Hwang, Joongheon Kim, Jihong Park, Mehdi Bennis, and Seong-Lyun Kim. Xor mixup: Privacy-preserving data augmentation for one-shot federated learning. *arXiv preprint arXiv:2006.05148*, 2020. 2

[41] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. *Advances in Neural Information Processing Systems*, 30, 2017. 2

[42] Canh T Dinh, Nguyen Tran, and Josh Nguyen. Personalized federated learning with moreau envelopes. *Advances in Neural Information Processing Systems*, 33:21394–21405, 2020. 2

[43] Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. Towards personalized federated learning. *IEEE Transactions on Neural Networks and Learning Systems*, 34(12):9587–9603, 2022. 1

[44] Shanli Tan, Hao Cheng, Xiaohu Wu, Han Yu, Tiantian He, Yew Soon Ong, Chongjun Wang, and Xiaofeng Tao. Fedcompetitors: Harmonious collaboration in federated learning with competing participants. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(14):15231–15239, 2024. 3

[45] Yue Tan, Guodong Long, Lu Liu, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. Fedproto: Federated prototype learning across heterogeneous clients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8432–8440, 2022. 2, 6

[46] Xiaohu Wu and Han Yu. Mars-fl: Enabling competitors to collaborate in federated learning. *IEEE Transactions on Big Data*, 10(6):801–811, 2024. 3

[47] Xinghao Wu, Xuefeng Liu, Jianwei Niu, Guogang Zhu, and Shaojie Tang. Bold but cautious: Unlocking the potential of personalized federated learning through cautiously aggressive collaboration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19375–19384, 2023. 1

[48] Jian Xu, Xinyi Tong, and Shao-Lun Huang. Personalized federated learning with feature alignment and classifier collaboration. *arXiv preprint arXiv:2306.11867*, 2023. 2, 6

[49] Jian Xu, Xinyi Tong, and Shao-Lun Huang. Personalized federated learning with feature alignment and classifier collaboration, 2023. 2

[50] Xiyuan Yang, Wenke Huang, and Mang Ye. Fedas: Bridging inconsistency in personalized federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11986–11995, 2024. 2, 6

[51] Jianqing Zhang, Yang Hua, Hao Wang, Tao Song, Zhengui Xue, Ruhui Ma, and Haibing Guan. Fedcp: Separating feature information for personalized federated learning via conditional policy. In *Proceedings of the 29th ACM SIGKDD*

conference on knowledge discovery and data mining, pages 3249–3261, 2023. 1

[52] Ziran Zhou, Guanyu Gao, Xiaohu Wu, and Yan Lyu. Personalized federated learning via learning dynamic graphs, 2025. 3