FedAS: Bridging Inconsistency in Personalized Federated Learning

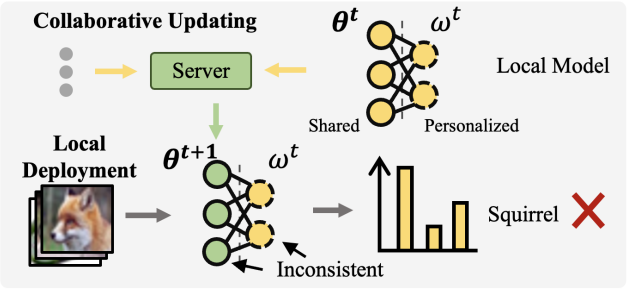
Xiyuan Yang[1] Wenke Huang[1] Mang Ye[1,2*]
[1]National Engineering Research Center for Multimedia Software,
School of Computer Science, Wuhan University, Wuhan, China
[2]Taikang Center for Life and Medical Sciences, Wuhan University, Wuhan, China
{yangxiyuan,wenkehuang,yemang}@whu.edu.cn
https://github.com/xiyuanyang45/FedAS

Abstract

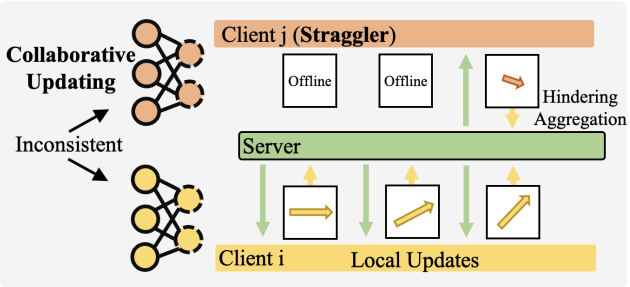
*Personalized Federated Learning (PFL) is primarily designed to provide customized models for each client to better fit the non-iid distributed client data, which is a inherent challenge in Federated Learning. However, current PFL methods suffer from inconsistencies in both intra-client and inter-client levels: 1) The intra-client inconsistency stems from the asynchronous update strategy for personalized and shared parameters. In PFL, clients update their shared parameters to communicate and learn from others, while keeping personalized parts unchanged, leading to poor coordination between these two components. 2) The Inter-client inconsistency arises from "stragglers" - inactive clients that communicate and train with the server less frequently. This results in their under-trained personalized models and impedes the collaborative training stage for other clients. In this paper, we present a novel PFL framework named **FedAS**, which uses **Fed**erated Parameter-**A**lignment and Client-**S**ynchronization to overcome above challenges. Initially, we enhance the localization of global parameters by infusing them with local insights. We make the shared parts learn from previous model, thereby increasing their local relevance and reducing the impact of parameter inconsistency. Furthermore, we design a robust aggregation method to mitigate the impact of stragglers by preventing the incorporation of their under-trained knowledge into aggregated model. Experimental results on Cifar10 and Cifar100 validate the effectiveness of our FedAS in achieving better performance and robustness against data heterogeneity.*

1. Introduction

Federated Learning (FL) is a distributed manner that facilitates collaborative model training across decentralized clients, all while preserving privacy. As a cornerstone solution, the FedAvg [36] algorithm maintains a global model in the central server, during the training process, this global model is distributed to the participating clients for further local training. After client-wise updating, the server collects and aggregates the optimized client network parameter to update the global model. Notably, the model optimized via the FedAvg shows a performance degradation due to data heterogeneity [15–17, 20, 27, 55, 65]. Specifically, as data is collected from different parties, *e.g.*, users, devices, organizations, and other diverse sources, it in-



(a) **Intra-client** Inconsistency



(b) **Inter-client** Inconsistency

Figure 1. **Problem Illustration** of Intra-client and Inter-client Inconsistency. The asynchronous parameter update in the personalized federated learning leads to the Intra-client Inconsistency between personalized and shared parts. The stragglers results in the Inter-client Inconsistency, which distorts aggregated update in the collaborative learning stage.

*Corresponding Author

evitably results in data heterogeneity of distribution shifts [19, 26, 45, 52, 56, 57]. Consequently, clients data is seldom independently and identically distributed (non-iid), leading to convergence difficulty in FL. Facing data heterogeneity, the primary task is to make model better attuned to individual data distributions. To address this problem, Personalized Federated Learning (PFL) [3, 31, 42, 54, 66] allows each client to create client-specific models to further adapt to local data.

Numerous studies have delved into various personalization techniques in PFL. A prevailing stream is parameter decoupling, which divides model parameters into personalized and shared parts. Only shared parts have been aggregated in server and transferred for inter-clients communication. In terms of decoupling methods, some research in representation learning [13, 21, 30, 40, 63] has proved the benefits of decoupling the training process into representation and prediction phases. Furthermore, Mendieta et al. [37], Yu et al. [58] points out that the prediction phase is highly related to the type of task. Based on these insights, it's a dominant trend in PFL to divide model parameters into a feature extractor (the backbone) and a classifier (the head) [9, 40, 53, 60–62]. This decoupling strategy set the model backbone as the shared part, synchronizing the feature extraction capabilities across clients. Concurrently, the classifier is set as the personalized part, focusing on a client-specific, task-related classification.

Despite the initial progress of PFL, current personalized federated learning methods generally exhibit inconsistencies on both intra-client and inter-client levels, as Figure 1 illustrates. ❶ **Firstly**, intra-client inconsistencies manifest between personalized and shared parameters due to their asynchronous update method. Specifically, at the start of client training, shared parameters are replaced by global parameters from server for collaborative updates, while personalized parameters remain unchanged. This parameter substitution discards the local information of the old shared parts, replacing it entirely with global knowledge, while the personalized part retains only local information. The difference in knowledge, data processing approaches, and optimization objectives between local and global side leads to discrepancies and conflicts within the shared and personalized parameters. These conflicts disrupt model prediction and training, which can ultimately lead to a reduction in the generalization capability and overall effectiveness of the personalized model. ❷ **Secondly**, we observe inter-client inconsistencies arising from the presence of offline clients, referred to as "stragglers", in the PFL training process [7, 33, 41, 43, 46]. Due to communication barriers, computational capacity, and other limitations, these stragglers seldom participate in local training and collaborative learning. Consequently, the personalized models of stragglers remain under-trained and under-optimized, possessing a limited scope of learned features. When these stragglers are considered in server aggregation, the less optimized parameters inside straggler model will be incorporated into global model. This, in turn, hinders the training of other online clients and slow down the overall model convergence.

In light of these challenges, we adopt a dual-step approach to address both issues. To address the ❶ intra-client inconsistency issue, our proposed strategy aim to localizes the global shared parameters, thereby incorporate it with local knowledge. This is done by aligning the output vectors of the new shared parts with previous shared parts, which is locally trained in last round. In doing so, our method enables the shared parameters to learn and adapt to the local data characteristics, improving consistency between both parameters. This Parameter-Alignment method bridge the local-global knowledge gap caused by asynchronous update and enhance the model's overall performance on client-specific data. In response to the ❷ inter-client inconsistency issue, a critical measure is to mitigate the adverse impact of stragglers. The key point of this challenge lies in the characteristics of stragglers, whose parameters are under-optimized, under-trained and exhibit a narrow range of learned knowledge. Thus, we propose leveraging the under-optimization characteristic in straggler models to address the issue. Our method utilizes the Fisher Information Matrix (FIM), a statistical tool that quantifies the information content of a model [23, 47, 54] by assessing the sharpness of log-likelihood function. Since an under-trained model's parameters are less informed and more uncertain, the corresponding FIM values—and by extension, the trace of the FIM (t-FIM)—tend to be lower for stragglers. Thus, by calculating the t-FIM value, we can effectively gauge the level of optimization and information content within each client's model. We incorporate this insight into server aggregation process by assigning weights based on the t-FIM value. This differential weighting safeguards the training process of online clients from being skewed by lesser-informed updates, thereby enhancing the convergence and robustness of the PFL system.

In this paper, we introduce the **Fed**erated Parameter-**A**lignment and Client-**S**ynchronization (FedAS) method. We leverage the Parameter Alignment method to localize shared parameters, making it consistent with the personalized parameters to handle intra-client inconsistency issue. We also adopt a Client Synchronization method, leveraging optimized-level based weighting method to prevent stragglers hindering active clients, alleviating the inter-client inconsistency issue. We believe that these two components together make FedAS a competitive method for PFL. Our main contributions are summarized as follows:

- We focus on personalized federated learning, highlighting two factors of inconsistency evident both at the intra-client and inter-client levels, and reveal that these incon-

sistencies degrades personalized models in performance, and hindering the overall-training process.
- We propose a novel personalized federated learning framework, obtaining more effective and accurate personalized models by leveraging client-end Parameter Alignment and server-end Client Synchronization.
- We conduct extensive evaluations on real datasets with various level of data heterogeneity and client online ratio. Accompanied with a set of ablative studies, promising results validate the efficacy of FedAS and the indispensability of each module.

2. Related Works

2.1 Data Heterogeneous Federated Learning

Federated learning was introduced to address privacy concerns in model training scenarios where data is not centralized. A fundamental method FedAvg [36] updates the global model by aggregating parameters trained by all clients on their private datasets. However, this method heavily relies on the assumption that all client data is independently and identically distributed (iid). In non-iid real-world scenarios, the performance of the FedAvg method declines significantly. In response to this challenge, several methods have been proposed, mainly including adding different penalty terms to clients and altering aggregation weights at the server-side. Specifically, methods such as FedProx [28], FedCurv [47], pFedME [48], and FedDyn [1] introduce magnitude regularization-based penalty terms to client parameters, drawing the client models closer to the global model. This helps in avoiding client model with data heterogeneous sources from deviating too far from the global model. Moreover, approaches like MOON [25], FCCL [14], FedUFO [64], FedProto [49], FPL [18], and FedProc [38] incorporate alignment-based penalty terms to their parameters, aiming to align feature representations among clients and thereby address data heterogeneity challenges. In addition, SCAFFOLD [22] and FedDC [12] employ gradient correction penalties that align with global gradient directions, ensuring consistent global update directions. Concurrently, techniques like Elastic Aggregation [5], RHFL [10], AugHFL [11] and FedHEAL [8] adjust the global aggregation weights to prioritize beneficial client updates to address data heterogeneity. While these methods improve upon FedAvg and can learn features from heterogeneous client data, they lack generalization in face of severe data heterogeneity paradigms. In this paper, we adopt Personalized Federated Learning (PFL) to tackle these issues and integrate local parameter alignment and global client synchronization to further enhance the learning of personalized models.

2.2 Personalized Federated Learning

Personalized Federated Learning (PFL) has attracted much attention due to its ability to address the issues of non-iid data distribution across clients. Current PFL methodologies primarily employ the following strategies to obtain client-specific models. Mainstream personalized federated learning techniques employ parameter decoupling to split models into two components: the feature extractor (the backbone) and the classifier (the head). These components undergo distinct training processes to achieve personalized federation. For instance, the LG-FedAvg [31] method treats the backbone as the personalized component and the head as the shared component. This design aims to provide clients with consistent classification strategies, mitigating the impact of non-iid data. Conversely, methods such as FedPer [3], FedRoD [6], FedGC [39], and FedBABU [40] designate the model's head as the personalized component. This approach allows various clients to share the same feature extraction method for heterogeneous data while employing client-specific classification strategies. Moreover, techniques like FedRep [9], FedPAC [53], FedProto [49], and FedPCL [50] utilize features outputted by the backbone, often referred to as the "prototype", to align outputs from different classifiers across clients. Additionally, methods like pFedME [48] and Ditto [29] maintain both personalized and global models by further training the global model locally to make it better adapt to the client data. Despite these advancements, these personalized techniques primarily focus on client-specific models to cater to heterogeneous data. However, they often overlook the inconsistencies that can arise when combining personalized parameters with shared parameters, and also neglect the synchronization issues caused by inactive clients during the training process. In our research, we build upon the foundation of PFL and introduce alignments both at the client and server levels, effectively addressing these overlooked challenges.

2.3 Fisher Information Matrix

The Fisher Information Matrix (FIM) [2, 34], a pivotal concept in statistical estimation theory, encapsulates the information an unknown parameter possesses about a random distribution. With respect to deep learning, FIM has been employed to study the loss functional curvature, guide optimization, and parameter information evaluation [4, 23, 35]. For instance, the Kronecker-Factored Approximate Curvature (K-FAC) method [35] utilizes a Kronecker product approximation to the FIM for more efficient natural gradient computations. The layer-wise relevance propagation method [4] uses the diagonal of the FIM to quantify the importance of features, thereby enhancing model interpretability. The Elastic Weight Consolidation algorithm [23] employs FIM to guard important parameters during the learn-

Notation	Description	Notation	Description
ω_i	Local Head	θ	Global Backbone
i	Client Index	θ_i	Local Backbone
D_i	Dataset	M	Number of clients
N_i	Size of D_i	α_i'	Aggregation weight
E_L	Local Epochs	W_i	Model of client i
α_i	Trace of FIM	H_i	Feature Embedding

Table 1. **Notations** occurred in the paper.

ing of new tasks, mitigating catastrophic forgetting. All these methods utilize the diagonal of the FIM for approximation, reducing computational complexity and facilitating more efficient learning processes. Although these methodologies have made significant strides, their application in personalized federated learning is restricted. Specifically, these methods are not directly applicable to personalized federated learning, as they don't account for the unique challenges posed by non-iid data distributions. In this work, we take a pioneering step by integrating the Fisher Information Matrix into the personalized federated learning framework. We utilize the sum of the trace of the FIM to gauge the knowledge acquired by the model and to ascertain the model's training progression. Our method further employs this value for weighted aggregation of client parameters. This innovative strategy minimizes the influence of stragglers, ensuring consistent training progression across clients and thus enhancing personalization outcomes.

3. Methodology

3.1 Preliminaries

Personalized Federated Learning. Personalized Federated Learning (PFL) extends Federated Learning (FL) by addressing the non-IID distribution of client data. In PFL, parameter of client model is decoupled into a local part, w to adapt to client data distribution, and a global part, θ to share knowledge among all clients. Assume we have M clients, each client i possesses a unique private dataset, denoted by $D_i = \{(x_j, y_j)\}_{j=1}^{N_i}$, where N_i signifies the size of the D_i. Each client has their own model with W as its parameter, we define the d_W as the dimension of model parameter vector W, the $W \in \mathbb{R}^{d_W}$ for each client i is separated into local and global parts, forming $W_i = (\theta, w_i)$, by personalization techniques. For simplicity and to include mainstream personalized federated learning techniques, we adopt the basic sharing backbone and personalize head method in the following statements. Furthermore, we discuss the compatibility of FedAS with other personalized parameter partitioning methods in Section 4.3.

The optimization objective in PFL is defined as follows:

$$\min_{\theta,\omega_{1:m}} \left\{ F(\theta, \omega_{1:m}) := \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}_i(\theta, \omega_i) \right\}, \tag{1}$$

where $w_{1:m}$ represents $(\omega_1, \ldots, \omega_m) \in \mathbb{R}^{d_\omega \times m}$ and the $\mathcal{L}_i(\theta, \omega_i) := \mathbb{E}_{D_i \sim P_i}[f_i(\theta, \omega_i, D_i)]$ is the emprical risk of client i, which can be also recognized as the loss function. PFL use a iteration of two steps to solve this problem:

- **Client Training:** During the local update phase, each client i initialize model with shared θ from server and personalized w_i : $W_i^t = (\theta^t, \omega_i^t)$, and then performs E_{local} iterations of updates on the model to obtain new parameters $W_i^{t+1} = (\theta_i^t, \omega_i^{t+1})$. Subsequently, they compute the update on shared parameters $\Delta\theta_i^t$ by calculating the difference between trained θ_i^t and shared θ^t.
- **Collaborative Update:** In the collaborative update phase, all clients send their local updates Δw to the server. The server then aggregates these updates to obtain new shared parameter $\theta^{t+1} = \theta^t + \frac{1}{m}\sum_{i=1}^{m} \Delta\theta_i^t$, which are then distributed back to all clients.

3.2 Aligning Intra-client Inconsistency

Motivation. Current personalizaiton methods decouple parameters in personalized head and shared backbone, directly combining them together to initialize client model leads to collaboration failure in the following training. Specifically, the backbone possess global knowledge and would generate a feature representation more similar to other clients, while the head is tuned locally and can only classify local feature representations. This discrepancy leads to different manners in processing local data, and makes it challenging for local head to correctly classify representations from other clients. In response to this issue, we aim to incorporate the local knowledge into shared parameters, enable it to generate local representation thus contribute to a better consistency with the local classifier. Therefore, we make the global backbone θ^t, to learn local representation knowledge from previous local-tuned θ_i^{t-1}, by aligning their outputs, thereby ensuring consistency with the personalized head, as shown in Figure 2.

Federated Parameter Alignment. To begin with, we leverage the previous backbone θ_i^{t-1} to generate feature embedding on client i's local dataset $D_i = \{(x, y)\}^{N_i}$ as :

$$H_i = \{f(\theta_i^{t-1}, D_i)\}^{N_i} \in \mathbb{R}^{d_{f(\theta)} \times N_i}, \tag{2}$$

where j is the index of local data, and N_i represents the number of images in the i-th client. The function $f(\theta_i^{t-1})$ represents the output embedding of parameter θ_i^{t-1}. In the next step, based on H_i obtained from θ_i^{t-1}, we also obtain the feature embedding from θ^t in the same way, and align it with the previous H_i according to the following formula:

$$\theta^t \leftarrow \theta^t - \nabla\mathcal{L}(f(\theta^t, D_i), H_i) \tag{3}$$

$$\mathcal{L}(f(\theta^t, D_i), H_i) = \left\| f(\theta^t, D_i) - H_i \right\|_2^2. \tag{4}$$

Here, the objective $\mathcal{L}$, known as the MSE-Loss, is to minimize the squared Euclidean distance between the feature

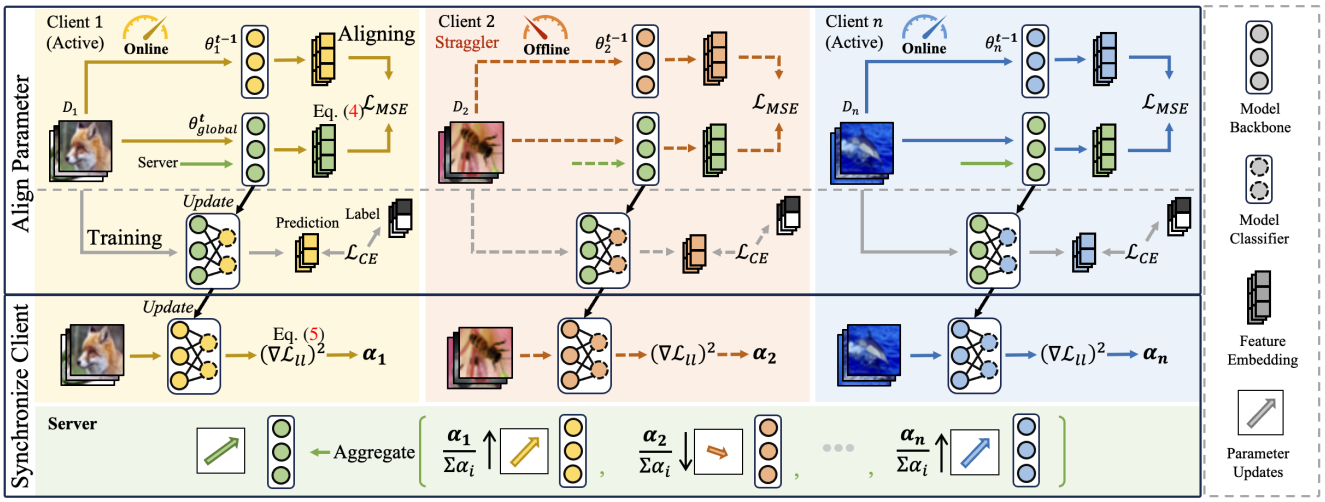
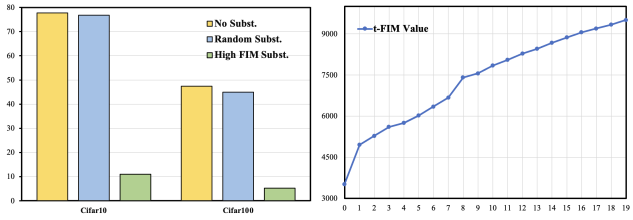


Figure 2. **Architecture Illustration** of the FedAS. Firstly, each client use the Parameter Alignment (Sec. 3.2) to localize the received backbone, followed by the local training. Then, we perform the Client Synchronization (Sec. 3.3), where each client gets the α value by calculating square gradient of log-likelihood function ($\nabla\mathcal{L}_{ll}$). On the server side, the server conduct a weighted model aggregation by α_i.



(a) Parameter Substitution Accuracy. (b) t-FIM Value Rising.

Figure 3. **Motivation Illustration** for Sec 3.3. In Fig.3a, substituting top 5% parameters with highest FIM value leads to huge accuracy drop. Fig.3b shows a continuous increase in t-FIM along with training, representing that model acquired more knowledge.

embedding using H_i. By updating θ^t, we effectively align the shared parameter's outputs with those of the previous round. This alignment process enable global backbone to learn the local feature embedding knowledge, thereby works more effectively with the local head, resulting in a better model personalization performance.

For the completeness of our method, we further compare our methods with other parameter decoupling methods such as Shen et al. [44], in which feature embeddings cannot be completed solely through shared parameters $f(\theta_i^{t-1}, D_i)$. In this situation, we adopt $f(W_i, D_i)$ to generate those embeddings as an alternative, as discussed in Section 4.3.

3.3 Synchronizing Inter-client Inconsistency

Motivation. The need to synchronize different clients arises from the stragglers in PFL. The stragglers model are under-trained, and their updates can influence the direction and magnitude of aggregated model, hindering training process. As discussed in Section 2.3, the FIM value can measure the information content of parameters. To make this view solid, we conducted experiments as Figure 3a shows. It demonstrates that substituting parameter with the top-5% highest FIM value leads to catastrophic accuracy degradation, while randomly substituting 5% parameters results minor effects, showing that FIM value effectively identify the information content of parameter. When it comes to summing FIM value (denote as t-FIM) across all parameters, it would represent the knowledge a client model learned. As we observed a constantly rising of t-FIM value for online clients, as Figure 3b shows, we found that stragglers have a relatively low t-FIM value comparing others. Leveraging this property, we perform weighted aggregation strategy based on t-FIM, as shown in Figure 2, to prevent the global aggregation influenced by stragglers.

Federated Client Synchronization. To implement the weightings based on the t-FIM value (noted as α), each client first calculates the t-FIM value at the end of the client training:

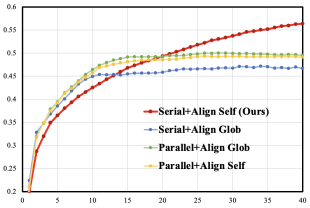
$$\text{FIM} = \nabla_{W_i} \log p(D_i|W_i) \cdot \nabla_{W_i} \log p(D_i|W_i)^T, \quad (5)$$

$$\alpha_i = \sum_{j=1}^{d_W} \text{diag}(\text{FIM})_j. \quad (6)$$

Here, $\nabla_{W_i} \log p(D_i|W_i)$ is the gradient of the model's log-likelihood function ($\mathcal{L}_{ll}$) with respect to its parameters W_i, α_i is the trace of FIM matrix(t-FIM), and D_i represents the private data for the i^{th} client. Following this, each client sends the calculated value α_i to the server. The server then uses these values to normalize and weight each client's parameters for aggregation:

$$\alpha_i' = \frac{\alpha_i}{\sum_{i=1}^{m} \alpha_i}, \quad (7)$$

(a) **Aggregation Weight** for Different Clients.



(b) Comparing with other Align Manner.

Figure 4. **Discussions** on the FedAS. In Fig.4a, stragglers have lower weight in aggregation. Fig.4b shows the superiority of our alignment method compared to other alignment implementations.

$$\theta^{t+1} = \theta^t + \frac{1}{m}\sum_{i=1}^{m} \alpha_i' * \Delta\theta_i^t, \tag{8}$$

Where θ^{t+1} signifies the new shared parameters, and θ_i^t is the model parameters from the i^{th}.

The proposed weighting strategy ensures that the aggregated model is influenced more by active clients and less by the under-trained stragglers. This approach not only speeds up the federated learning process but also improves the overall generalization capability of the aggregated model.

Algorithm 1: FedAS

Input: Global epochs E_g, local epochs E_l, participants number M_t, i^{th} client's private dataset D_i, initial parameters ω_i^* and θ^*, learning rate η

Client Training:
for $i = 1, 2, ..., M_t$ **do**
  Calculate output of shared parameters H_i
  Paramaters Alignment by (4)
  for $e = 1, 2, ..., E_l$ **do**
    $(\theta_i^t, \omega_i^{t+1}) \leftarrow (\theta^t, \omega_i^t) - \eta \cdot \nabla \text{CrossEntropy}(W_t^i)$
  end
  Caltulate α_i by (5) and (6)
  Calculate parameter updates $\Delta\theta_i^t = \theta_i^t - \theta^t$
  Return α_i and $\Delta\theta_i^t$ to Server
end
Collaborative update:
for $t = 1, 2, ..., E_g$ **do**
  $\Delta\theta_i^t, \alpha_i \leftarrow$ **Client Training( )**
  Calculate α_i' as aggregation weights using (7)
  Aggregate parameters using (8)
  for $i = 1, 2, ..., m$ **do**
    Send θ^{t+1} to i^{th} participant
  end
end

3.4 Discussion and Limitation.

Comparison with Analogous Methods. Some other methods in [18, 49, 51, 53, 59] also utilize the output of the backbone, often referred to as 'prototype'. However, their prototype alignment only focus on learning from other clients, neglecting learning from local representation for localization. In our approach, the constructed H_i in Eq.(2) can also be perceived as a kind of prototype. However, a key insight of our proposed method is that we are aligning self-prototype, instead of global prototype in aforementioned methods. Our method adapt global model into local representation by H_i, achieving a better personalization model.

Effectiveness of the t-FIM Metrics. As discussed in Section 3.3, we can observe relatively low t-FIM value on stragglers, and resulting in lower aggregation weight. To prove the effectiveness of this metric, we conducted experiments, simulating a fixed straggler portion, say 0.5, and we show the aggregation weight α_i in Figure 4a. It's clear that online clients have higher α than stragglers, demonstrating the effectiveness of our proposed method.

Discussion on Parameter Alignment. We proposed a serial training method on client side: Client conducted the Parameter Alignment before local training. Our insight is that: 1) The PA method is to adapt global backbone to local knowledge, promoting a better collaboration between backbone and local head. If treating this alignment in a parallel manner ,overemphasize local knowledge would hinder model training. 2) Comparing aligning with global feature, the alignment is more beneficial towards local feature, contributing to better personalization. We conduct experiments on different align manner and align object in Figure 4b, and we observed that our FedAS achieved the best performance.

Limitation. Our proposed method FedAS assumes all clients share the same model architecture, which may not always be the case in real-world scenarios. Heterogeneity in model structures across clients can potentially limit the utility of our alignment method, which is an area where further research is needed. It should be noted that this limitation is not unique to our approach but is also shared by most of the above state-of-the-art methods [28, 29, 31, 53, 61].

4. Experiments

4.1 Experimental Setup

Datasets. We evaluate our methods on two image classification tasks:

- Cifar10 [24]: Cifar10 consists of 60,000 32x32 color images, evenly distributed across 10 different classes, making up 6,000 images per class.
- Cifar100 [24]: Cifar100 is a sister dataset to Cifar10, which consists of 60,000 32x32 color images but is distributed across 100 classes, with 600 images per class.

For both datasets, we partitioned the data based on a Dirichlet distribution [32], and distributed the partitioned dataset across all clients. We adjust the β parameter in the Dirchlet distribution to simulate degrees of non-IID distribution and

Methods	$\beta = 0.1$				$\beta = 0.3$				$\beta = 0.5$			
	$P = 0.2$	$P = 0.4$	$P = 0.6$	$P = 1$	$P = 0.2$	$P = 0.4$	$P = 0.6$	$P = 1$	$P = 0.2$	$P = 0.4$	$P = 0.6$	$P = 1$
FedAvg-P[36]	27.61	36.42	41.61	47.57	27.87	32.67	36.35	38.67	27.44	30.66	32.07	33.82
FedProx-P[28]	27.42	35.40	40.41	45.96	27.27	32.25	35.29	37.85	26.99	30.22	31.36	33.46
FedPer[3]	44.96	45.12	44.32	44.40	34.78	34.23	33.66	33.08	29.77	28.71	27.79	27.17
Ditto[29]	32.27	38.19	41.90	44.48	24.90	26.89	29.40	30.91	25.71	27.39	27.58	27.92
LG-FedAvg[31]	42.68	44.04	46.51	46.37	29.95	31.67	33.26	33.31	24.97	26.32	26.96	27.20
FedBABU[40]	49.85	49.79	49.88	49.78	38.18	38.89	38.81	38.69	33.60	33.88	33.18	33.38
FedProto[49]	26.66	44.32	49.85	51.88	20.66	33.70	36.99	38.23	17.62	28.46	31.58	32.19
FedALA[61]	45.52	48.40	50.67	51.06	36.54	40.41	41.20	42.08	31.19	35.07	36.85	37.15
FedPAC[53]	45.08	48.63	50.87	52.03	34.15	37.33	38.60	39.80	29.99	33.56	34.97	35.94
FedAS(Ours)	**52.41**	**55.20**	**55.76**	**56.37**	**40.68**	**43.51**	**44.47**	**44.91**	**35.48**	**38.21**	**38.70**	**39.57**
Δ	↑**6.89**	↑**5.41**	↑**4.89**	↑**4.34**	↑**4.14**	↑**3.10**	↑**3.27**	↑**2.83**	↑**4.29**	↑**3.14**	↑**1.85**	↑**2.42**

Table 2. Comparison with State-Of-The-Art Mehtods on **Cifar100** Classification Task. See Details in Sec.4.2.

Methods	$\beta = 0.1$				$\beta = 0.3$				$\beta = 0.5$			
	$P = 0.2$	$P = 0.4$	$P = 0.6$	$P = 1$	$P = 0.2$	$P = 0.4$	$P = 0.6$	$P = 1$	$P = 0.2$	$P = 0.4$	$P = 0.6$	$P = 1$
FedAvg-P[31]	56.95	67.04	78.71	85.26	57.71	68.26	73.64	76.60	63.47	68.21	72.15	73.59
FedProx-P[28]	56.84	66.03	78.04	84.67	57.39	67.50	73.09	76.18	62.14	67.45	71.54	72.86
FedPer[3]	88.51	89.17	89.65	89.79	77.83	77.39	77.24	76.99	73.71	72.70	72.51	72.17
Ditto[29]	83.59	86.00	87.26	87.86	66.58	71.26	73.58	74.97	60.41	62.60	65.23	66.80
LG-FedAvg[31]	87.86	88.52	88.66	88.95	75.40	75.94	76.87	77.13	67.29	67.77	68.48	68.47
FedBABU[40]	88.45	88.66	88.48	88.57	78.35	78.34	78.16	78.26	73.23	72.75	72.83	72.71
FedProto[49]	61.17	84.18	87.47	88.30	64.12	74.13	76.49	77.88	58.58	66.47	69.22	70.55
FedALA[61]	89.61	89.89	90.00	90.06	79.03	80.92	81.02	81.40	75.05	76.24	76.52	76.75
FedPAC[53]	87.48	88.36	88.60	87.08	78.36	80.52	81.55	82.04	74.98	76.47	77.23	77.89
FedAS(Ours)	**89.92**	**90.60**	**90.81**	**91.03**	**80.23**	**81.58**	**82.01**	**82.12**	**75.93**	**77.29**	**77.35**	**77.97**
Δ	↑**0.31**	↑**0.71**	↑**0.81**	↑**0.97**	↑**1.20**	↑**0.66**	↑**0.46**	↑**0.08**	↑**0.95**	↑**0.82**	↑**0.12**	↑**0.08**

Table 3. Comparison with State-Of-The-Art Mehtods on **Cifar10** Classification Task. See Details in Sec.4.2.

assess accuracy accordingly. This wide-ranging experimental setup reflects practical scenarios and ensures a thorough assessment of our method's robustness and compatibility.

Model. For the classification tasks on both datasets, we employ a 6-layer CNN network. For the Cifar10 dataset, the output of the final layer is set to 10. Correspondingly, for the Cifar100 task, the output of the final layer is 100.

Counterparts. We compare our FedAS against many strong state-of-the-art PFL methods that focus on learning client-specific models: LG-FedAvg [31], FedPer [3], Ditto [29], FedBABU [40], FedALA [61], FedProto [49], and FedPAC [53]. It's worth noting that, as foundational algorithms in FL, FedAvg [36] and FedProx [28] aim to achieve a generalized global model, which is a different objective from that of PFL. To include these two cornerstone into our comparison, we further finetune global models of FedAvg and FedProx to get personalized models, and the accuracy of these adapted models are referred to as FedAvg-P (Peronsliazed) and FedProx-P in the table respectively.

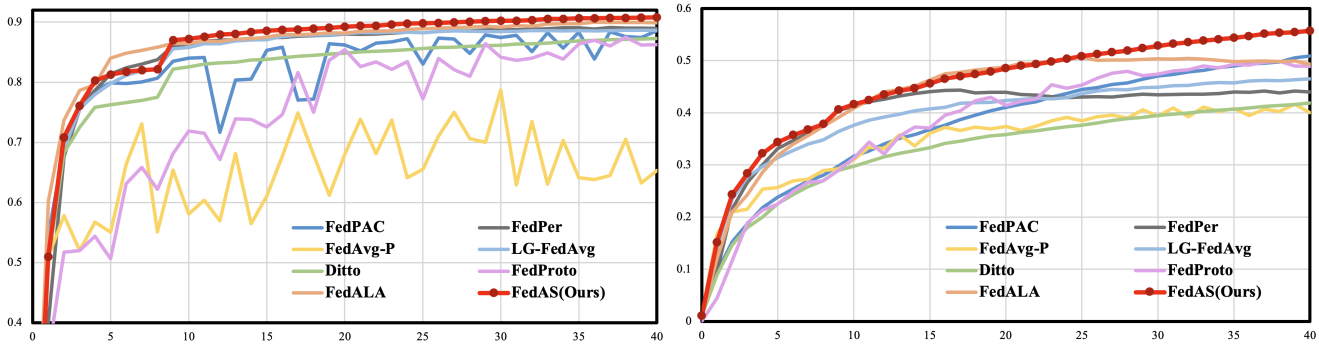
Implementation Details. To ensure a fair comparison, we use the same hyper-parameter settings across all experiments. In the experiments, we set the number of clients to 20, set the global round $E_g = 40$, and set the local updating epoch $E_l = 5$. We employed the SGD optimizer, with a learning rate of $5e-3$, and the learning rate decay value was set to $1e-3$. The training batch size is 16. For all the accuracy value in tables, we set the random seed of pytorch and numpy as 0, and the highest top-1 accuracy is adopted.

4.2 Comparison to State-of-the-Arts

We present experiment comparing FedAS with state-of-the-art (SOTA) methods on Cifar10 and Cifar100 datasets [24]. Throughout all experiments, we set the parameter β in $\{0.1, 0.3, 0.5\}$ to simulate different level of data heterogeneity, a larger β value represents a more non-iid distribution. We also vary client online ratios, noted as P in $\{0.2, 0.4, 0.6, 1\}$, a smaller P means more stragglers in all clients. As shown in Table 2 and 3, our method consistently achieves highest accuracy among all settings, demonstrating its effectiveness in personalization and robustness against stragglers. We also present the accuracy curve in Figure 4, showing its high accuracy and fast convergence.

4.3 Diagnostic Analysis

Ablation Study. For thoroughly analyzing the efficacy of each module, we perform an ablation study to investigate the effectiveness of Parameter-Alignment (PA) and Client-Synchronization(CS). We present the overall accuracy in Table 4. The results illustrate that both PA and CS contribute significantly to the performance of the model. The combination of both modules provides best results, underscoring the effectiveness of our proposed method.

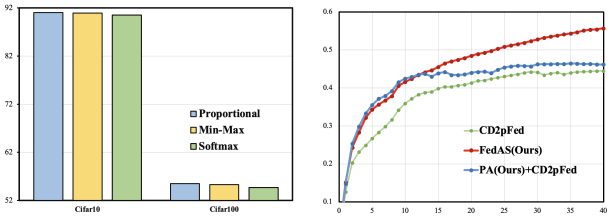


(a) Accuracy Curve on **Cifar10**. (b) Accuracy Curve on **Cifar100**.

Figure 5. **Accuracy Curve** along with global training rounds. The hyperparameter is set to $P = 0.6$ and $\beta = 0.1$. See Details in Sec.4.2.

Module PA	Module CS	Test Accuracy $P = 0.2$	$P = 0.4$	$P = 0.6$	$P = 1$
		88.51	89.17	89.65	89.79
✓		89.34	89.50	89.72	89.91
	✓	89.05	89.39	89.90	89.94
✓	✓	**89.92** ↑0.58	**90.60** ↑1.10	**90.81** ↑0.91	**91.03** ↑1.09

Table 4. Ablation Study. See details in 4.3

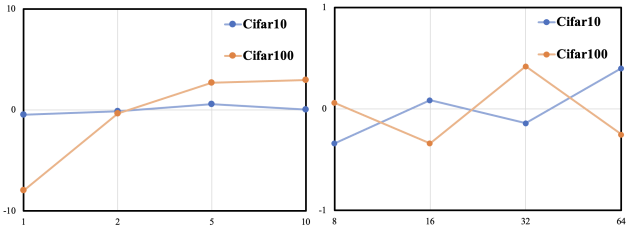


(a) Compatibility with Normalization Methods. (b) Compatibility with Decoupling Methods.

Figure 6. **Discussions** on Compatibility. We adopted other normalization methods when calculating aggregation weight (in Fig.6a), and changed the parameter decoupling methods (in Fig.6b) to show the compatibility of FedAS.

Compatibility with other Normalization Methods. In the Server Aggregation Phase, we calculate the aggregation weight α_i by Eq.(7), which is a proportional based normalization method. We also conducted experiments with other weight normalization methods as Figure 6a shows. The $softmax$ curve uses a $\frac{e^{w_i}}{\Sigma_{i=1}^{n} e^{w_i}}$ in normalization, and the $min\text{-}max$ method normalize weights in a $\frac{w_i - \text{MIN}(w)}{\text{MAX}(w) - \text{MIN}(w)}$ manner. It's clear that all methods achieve fairly high accuracy, showing robustness against normalization methods.

Compatibility with other Decoupling Methods. We incorporate our method with a channel-wise parameter decoupling method, named CD2pFed [44], which isn't similar with common head-backbone split methods. The accuracy curve is as shown in Figure 6b. The addition of our parameter-Alignment(PA) method increased this personalization method in performance significantly, further validating the compatibility of our approach.



(a) Accuracy Changes with E_l. (b) Accuracy Changes with BS.

Figure 7. **Discussions** on Hyper-Parameters. We present the change of accuracy (percentage) comparing to mean-value in above picture. The E_l is local epoch and BS is the batch size.

Discussion on other Hyper-Parameters. We also explored the influence of batch-size BS and local epochs E_l on model accuracy. For fairness in comparison, we ensure the same update steps for each client model by changing global round E_g accordingly. We change the value of BS in $\{8, 16, 32, 64\}$, E_l in $\{1, 2, 5, 10\}$, and present the value changing comparing with average accuracy (the central zero-line) and corresponding results in Figure 7. The curve shows that our proposed method shows robustness against batch size and training epochs, validating the effectiveness and robustness of out method.

5. Conclusion

In this paper, we explore the inconsistency problem in personalized federated learning. Our work introduces a simple yet effective algorithm FedAS. We leverage the Parameter-Alignment and Client-Synchronization to tackle these two problems. The effectiveness of FedAS has been thoroughly validated with many popular counterparts over various classification tasks. We wish this work to pave the way for future research on personalized federated learning.

Acknowledgement. This work is supported by National Natural Science Foundation of China under Grant (62361166629,62176188,62272354).

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. 3

[2] Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 1998. 3

[3] Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. Federated learning with personalization layers. *arXiv preprint arXiv:1912.00818*, 2019. 2, 3, 7

[4] Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, Klaus-Robert Müller, and Wojciech Samek. Layer-wise relevance propagation for neural networks with local renormalization layers. In *ICANN*, 2016. 3

[5] Dengsheng Chen, Jie Hu, Vince Junkai Tan, Xiaoming Wei, and Enhua Wu. Elastic aggregation for federated optimization. In *CVPR*, 2023. 3

[6] Hong-You Chen and Wei-Lun Chao. On bridging generic and personalized federated learning for image classification. In *ICLR*, 2022. 3

[7] Rui Chen, Qiyu Wan, Pavana Prakash, Lan Zhang, Xu Yuan, Yanmin Gong, Xin Fu, and Miao Pan. Workie-talkie: Accelerating federated learning by overlapping computing and communications via contrastive regularization. In *ICCV*, 2023. 2

[8] Yuhang Chen, Wenke Huang, and Mang Ye. Fair federated learning under domain skew with local consistency and domain diversity. In *CVPR*, 2024. 3

[9] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. In *ICML*, 2021. 2, 3

[10] Xiuwen Fang and Mang Ye. Robust federated learning with noisy and heterogeneous clients. In *CVPR*, 2022. 3

[11] Xiuwen Fang, Mang Ye, and Xiyuan Yang. Robust heterogeneous federated learning under data corruption. In *ICCV*, 2023. 3

[12] Liang Gao, Huazhu Fu, Li Li, Yingwen Chen, Ming Xu, and Cheng-Zhong Xu. Feddc: Federated learning with non-iid data via local drift decoupling and correction. In *CVPR*, 2022. 3

[13] Erdong Hu, Yuxin Tang, Anastasios Kyrillidis, and Chris Jermaine. Federated learning over images: Vertical decompositions and pre-trained backbones are difficult to beat. In *ICCV*, 2023. 2

[14] Wenke Huang, Mang Ye, and Bo Du. Learn from others and be yourself in heterogeneous federated learning. In *CVPR*, pages 10133–10143, 2022. 3

[15] Wenke Huang, Mang Ye, Xiang Gao, and Bo Du. Few-shot model agnostic federated learning. In *ACM MM*, 2022. 1

[16] Wenke Huang, Guancheng Wan, Mang Ye, and Bo Du. Federated graph semantic and structural learning. In *IJCAI*, 2023.

[17] Wenke Huang, Mang Ye, Zekun Shi, and Bo Du. Generalizable heterogeneous federated cross-correlation and instance similarity learning. *IEEE TPAMI*, 2023. 1

[18] Wenke Huang, Mang Ye, Zekun Shi, He Li, and Bo Du. Rethinking federated learning with domain shift: A prototype view. In *CVPR*, 2023. 3, 6

[19] Xuefeng Jiang, Sheng Sun, Yuwei Wang, and Min Liu. Towards federated learning against noisy labels via local self-regularization. In *CIKM*, 2022. 2

[20] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Found. Trends Mach. Learn.*, 2021. 1

[21] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. In *ICLR*, 2020. 2

[22] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *ICML*, 2020. 3

[23] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *PNAS*, 2017. 2, 3

[24] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 6, 7

[25] Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. In *CVPR*, 2021. 3

[26] Qinbin Li, Yiqun Diao, Quan Chen, and Bingsheng He. Federated learning on non-iid data silos: An experimental study. In *ICDE*, 2022. 2

[27] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE SPM*, 2020. 1

[28] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *MLSys*, 2020. 3, 6, 7

[29] Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through personalization. In *ICML*, 2021. 3, 6, 7

[30] Zexi Li, Xinyi Shang, Rui He, Tao Lin, and Chao Wu. No fear of classifier biases: Neural collapse inspired federated learning with synthetic and fixed classifier. In *ICCV*, 2023. 2

[31] Paul Pu Liang, Terrance Liu, Ziyin Liu, Ruslan Salakhutdinov, and Louis-Philippe Morency. Think locally, act globally: Federated learning with local and global representations. In *NeurIPS*, 2019. 2, 3, 6, 7

[32] Tao Lin, Lingjing Kong, Sebastian U Stich, and Martin Jaggi. Ensemble distillation for robust model fusion in federated learning. 2020. 6

[33] Kangyang Luo, Xiang Li, Yunshi Lan, and Ming Gao. Gradma: A gradient-memory-based accelerated federated learning with alleviated catastrophic forgetting. In *CVPR*, 2023. 2

[34] Alexander Ly, Maarten Marsman, Josine Verhagen, Raoul PPP Grasman, and Eric-Jan Wagenmakers. A tutorial

on fisher information. *Journal of Mathematical Psychology*, 2017. 3

[35] James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *ICML*, 2015. 3

[36] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017. 1, 3, 7

[37] Matias Mendieta, Taojiannan Yang, Pu Wang, Minwoo Lee, Zhengming Ding, and Chen Chen. Local learning matters: Rethinking data heterogeneity in federated learning. In *CVPR*, 2022. 2

[38] Xutong Mu, Yulong Shen, Ke Cheng, Xueli Geng, Jiaxuan Fu, Tao Zhang, and Zhiwei Zhang. Fedproc: Prototypical contrastive federated learning on non-iid data. *Future Generation Computer Systems*, 2023. 3

[39] Yifan Niu and Weihong Deng. Federated learning for face recognition with gradient correction. In *AAAI*, 2022. 3

[40] JAE HOON OH, Sangmook Kim, and Seyoung Yun. Fedbabu: Toward enhanced representation for federated image classification. In *ICLR*, 2022. 2, 3, 7

[41] Zhe Qu, Rui Duan, Lixing Chen, Jie Xu, Zhuo Lu, and Yao Liu. Context-aware online client selection for hierarchical federated learning. *IEEE TPDS*, 2022. 2

[42] Zhe Qu, Xingyu Li, Xiao Han, Rui Duan, Chengchao Shen, and Lixing Chen. How to prevent the poor performance clients for personalized federated learning? In *CVPR*, 2023. 2

[43] Amirhossein Reisizadeh, Aryan Mokhtari, Hamed Hassani, Ali Jadbabaie, and Ramtin Pedarsani. Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. In *AISTATS*, 2020. 2

[44] Yiqing Shen, Yuyin Zhou, and Lequan Yu. Cd2-pfed: Cyclic distillation-guided channel decoupling for model personalization in federated learning. In *CVPR*, 2022. 5, 8

[45] Jiangming Shi, Shanshan Zheng, Xiangbo Yin, Yang Lu, Yuan Xie, and Yanyun Qu. Clip-guided federated learning on heterogeneous and long-tailed data. *arXiv preprint arXiv:2312.08648*, 2023. 2

[46] Nir Shlezinger, Mingzhe Chen, Yonina C Eldar, H Vincent Poor, and Shuguang Cui. Uveqfed: Universal vector quantization for federated learning. *IEEE TSP*, 2020. 2

[47] Neta Shoham, Tomer Avidor, Aviv Keren, Nadav Israel, Daniel Benditkis, Liron Mor-Yosef, and Itai Zeitak. Overcoming forgetting in federated learning on non-iid data. *arXiv preprint arXiv:1910.07796*, 2019. 2, 3

[48] Canh T Dinh, Nguyen Tran, and Josh Nguyen. Personalized federated learning with moreau envelopes. In *NeurIPS*, 2020. 3

[49] Yue Tan, Guodong Long, Lu Liu, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. Fedproto: Federated prototype learning across heterogeneous clients. In *AAAI*, 2022. 3, 6, 7

[50] Yue Tan, Guodong Long, Jie Ma, Lu Liu, Tianyi Zhou, and Jing Jiang. Federated learning from pre-trained models: A contrastive learning approach. 2022. 3

[51] Guancheng Wan, Wenke Huang, and Mang Ye. Federated graph learning under domain shift with generalizable prototypes. In *AAAI*, 2024. 6

[52] Nannan Wu, Li Yu, Xin Yang, Kwang-Ting Cheng, and Zengqiang Yan. Fediic: Towards robust federated learning for class-imbalanced medical image classification. In *MICCAI*. Springer, 2023. 2

[53] Jian Xu, Xinyi Tong, and Shao-Lun Huang. Personalized federated learning with feature alignment and classifier collaboration. In *ICLR*, 2023. 2, 3, 6, 7

[54] Xiyuan Yang, Wenke Huang, and Mang Ye. Dynamic personalized federated learning with adaptive differential privacy. In *NeurIPS*, 2023. 2

[55] Mang Ye, Xiuwen Fang, Bo Du, Pong C Yuen, and Dacheng Tao. Heterogeneous federated learning: State-of-the-art and research challenges. *ACM CSUR*, 2023. 1

[56] Mang Ye, Wenke Huang, Zekun Shi, He Li, and Du Bo. Revisiting federated learning with label skew: An overconfidence perspective. *SCIS*, 2024. 2

[57] Mang Ye, Wei Shen, Junwu Zhang, Yao Yang, and Bo Du. Securereid: Privacy-preserving anonymization for person re-identification. *IEEE TIFS*, 2024. 2

[58] Haiyang Yu, Ningyu Zhang, Shumin Deng, Zonggang Yuan, Yantao Jia, and Huajun Chen. The devil is the classifier: Investigating long tail relation classification with decoupling analysis. *arXiv preprint arXiv:2009.07022*, 2020. 2

[59] Jianqing Zhang, Yang Hua, Jian Cao, Hao Wang, Tao Song, Zhengui XUE, Ruhui Ma, and Haibing Guan. Eliminating domain bias for federated learning in representation space. In *NeurIPS*, 2023. 6

[60] Jianqing Zhang, Yang Hua, Hao Wang, Tao Song, Zhengui Xue, Ruhui Ma, Jian Cao, and Haibing Guan. Gpfl: Simultaneously learning global and personalized feature information for personalized federated learning. In *ICCV*, 2023. 2

[61] Jianqing Zhang, Yang Hua, Hao Wang, Tao Song, Zhengui Xue, Ruhui Ma, and Haibing Guan. Fedala: Adaptive local aggregation for personalized federated learning. In *AAAI*, 2023. 6, 7

[62] 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 *KDD*, 2023. 2

[63] Jianqing Zhang, Yang Liu, Yang Hua, and Jian Cao. Fedtgp: Trainable global prototypes with adaptive-margin-enhanced contrastive learning for data and model heterogeneity in federated learning. In *AAAI*, 2024. 2

[64] Lin Zhang, Yong Luo, Yan Bai, Bo Du, and Ling-Yu Duan. Federated learning for non-iid data via unified feature learning and optimization objective alignment. In *CVPR*, pages 4420–4428, 2021. 3

[65] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018. 1

[66] Junyi Zhu, Xingchen Ma, and Matthew B. Blaschko. Confidence-aware personalized federated learning via variational expectation maximization. In *CVPR*, 2023. 2