Personalized Federated Learning via Knowledge Sharing-based Model Structure Adaption

Xiaochan Wang
Shenzhen International Graduate School, Tsinghua University
Shenzhen, China
wxc20@mails.tsinghua.edu.cn

Zhi Wang
Tsinghua-Berkeley Shenzhen Institute, Tsinghua University
Shenzhen, China
wangzhi@sz.tsinghua.edu.cn

***Abstract*—Federated Learning (FL) enables multiple clients to jointly train a global model without exchanging their individual data. Due to the diversity of data and systems in FL systems, conventional global training methods are insufficient. Personalized techniques have thus been explored. However, existing approaches concentrate mainly on modifying model parameters rather than model structures. This paper presents FedKMA, a personalized FL framework that identifies the optimal architecture for each client to fully reflect its characteristics. FedKMA employs the neural architecture search (NAS) approach in a federated setting to personalize model structures. To reconcile collaboration and personalization, FedKMA features a knowledge sharing algorithm that uses the intermediate results of local inference. The algorithm generates a knowledge sharing matrix to determine the weighted aggregation weights by capturing the similarity of learning results among clients. Thus, FedKMA achieves a balance between collaboration and personalization. Experiments on several benchmark datasets reveal that FedKMA significantly outperforms the existing state-of-the-art in terms of accuracy.**

I. Introduction

Federated learning (FL) [18], [30] allows for collaboration among clients without sharing private data. The goal of FL is to generate a global model aggregated from local models trained by clients [40]. However, due to the heterogeneity of clients, including differences in data and systems, the benefit of collaboration is limited [23]. This has led to an increased focus on personalized FL, which allows clients to train on different models instead of one identical shared model [36]. Personalized FL has enabled clients to achieve better local performance, but most efforts have only focused on varying model parameters, ignoring the benefits of personalized model structures [8], [9], [15]. The mismatch between model structures and client characteristics can cause various problems. Allocating complex models to clients with limited computational resources can lead to lower convergence and higher resource consumption, and mismatching between model complexity and data features can result in overfitting or underfitting. Therefore, we propose allowing clients to train on heterogeneous model structures for improved local performance, including higher accuracy and faster convergence rate.

Corresponding author: Zhi Wang, wangzhi@sz.tsinghua.edu.cn. This work was supported by Shenzhen Science and Technology Program (Grant No. RCYX20200714114523079 and JCYJ20220818101014030).

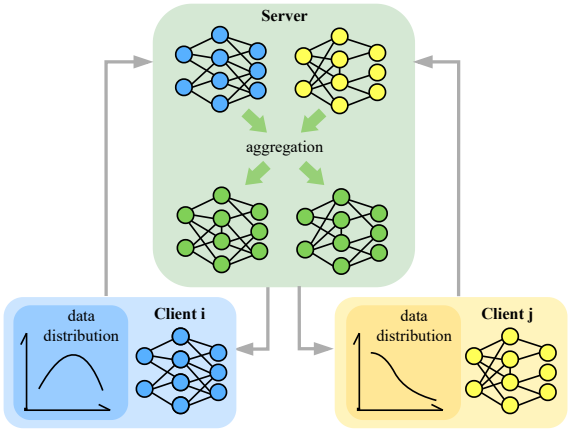


Fig. 1: An illustration of personalized FL with model heterogeneity. For the purpose of obtaining model heterogeneity while saving computational resources, the clients train on different models and the server obtains different personalized models after aggregating.

Despite the benefits of applying model structure personalization, great challenges still exist. First, it is difficult to allocate suitable models to clients without knowing their characteristics including local data distributions and other system capacities [41]. Models should be assigned based on client characteristics, as mentioned previously. However, the allocating strategies are no longer effective while the client features are unknown to the server. Second, aggregating models with different structures remains a problem. Since the aggregating in FL is actually a weighted average among vectors with the same size, the model parameters are no longer additive when they are in different shapes [34]. Moreover, most of the personalized FL methods require clients to perform local fine-tuning. However, the local fine-tuning on model structures brings extra computational overhead [39], [42], since structure fine-tuning requires supernets with much larger parameter space, thereby resulting in extra computational resource consumption. Thus, the structure fine-tuning slows down the entire training process while the federated clients are often resource-constraint [26].

To overcome these challenges, we propose FedKMA—a

personalized Federated learning framework via Knowledge-sharing based Model structure Adaption. FedKMA focuses on maintaining model heterogeneity in FL system while pursuing higher local performance. To achieve that, FedKMA tackles the first two technical challenges above by leveraging the neural architecture search [28] techniques to allow clients to search optimal models on the same super-network. On the basis of neural architecture search, we further propose a knowledge sharing algorithm that denotes the proportion in which the local models are shared with others, in order to lighten the computational consumption caused by local fine-tuning in the model heterogeneous FL system. Different from existing FL algorithms which evaluate client relevancy by computing the distances among updated gradients/parameters [10], [33], our knowledge sharing algorithm computes clients relevancy by computing the KL-divergence between the averaged inference logits from each client.

Our key contributions can be summarized as follows.

1) We propose FedKMA, a personalized FL framework with local model structure adaption under heterogeneous settings. We leverage neural architecture search in FedKMA to allow clients to search for their optimal local model structures, which also addresses the challenge of coordinate-wise aggregation in personalized FL with model structure adaption.
2) Within FedKMA, we propose a knowledge sharing algorithm based on inference logits and local model structures to determine the aggregating proportion. The knowledge sharing algorithm evaluates the learning status of clients through logits to determine the proportion of personalized aggregation and achieves personalized allocation.
3) Our evaluation results demonstrate that FedKMA brings enhancement to data heterogeneous FL by involving personalized models and improving the local inference accuracy by 0.4%-32% for each client. FedKMA also shows the ability to capture client relevancy.

II. Related Work

In this section, we first introduce efforts in traditional personalized FL that do not support model heterogeneity. Then, we introduce model heterogeneous personalized FL which allows clients to train on different models.

A. Traditional Personalized FL

The personalized FL methods emerged as a new solution for addressing heterogeneity in FL. Instead of training one global model collaboratively, personalized FL tries to find optimal models for each client. Existing efforts achieved personalization mostly by parameter adjustment. Some efforts are based on multi-task learning while [6], [35] enabled clients to learn similar models, [9], [11], [17] regularized local models from a global model, [15] proposed FedAMP that enforces pair-wise collaboration among FL clients with similar model parameters. Some methods focused on the post-processing (e.g., fine-tuning on the global model) [1], [5], [25] which also showed efficiency.

There also exists a series of approaches that focused on partitioning clients into different homogeneous groups and performing classical aggregation [3], [12], [33]. However, these efforts failed on allowing clients to train on different model structures.

B. FL with Model Heterogeneity

There are two main types of approaches to support model heterogeneity, which are knowledge distillation-based and pruning-based approaches. Instead of requiring clients to share model parameters of the same size or dimension, knowledge distillation-based approaches require clients to share intermediate learning results (e.g., logits) to accomplish knowledge migration [4], [14], [21], [32], thus enabling different clients to train models with different structures. The pruning-based approach, on the other hand, assigns the same dense network to all clients in the initial stage and requires clients to complete pruning individually in the training process, thus eventually achieving model heterogeneity among clients.

FD [16] and FedMD [21] allowed the server to collect the class scores (e.g., logits) of the public data set on each client model, and obtain the average logits as the updated consensus. FedDF [27] leveraged ensemble distillation for model aggregation. KT-pFL [41] updates the personalized soft prediction of each client by a linear combination of all local soft predictions, thus allowing model aggregation among various backbones. Compared with other personalized approaches, KT-pFL experiences roughly performance degradation. Moreover, almost all the methods based on knowledge distillation required a carefully-designed public dataset based on prior knowledge of local data distributions, which is conflicted with the FL settings.

Other approaches achieved model heterogeneity by pruning, e.g., generating sub-networks through a super-network or by pruning global models into personalized ones [20], [29]. HeteroFL [8] applied personalization by assigning models with different sizes to clients with heterogeneous computational capacities. FedNAS [13] and SPIDER [31] utilized neural architecture search yet paid less attention to balance model structure and parameters sharing and personalization.

In summary, existing methods usually focused on heterogeneous model fusion or model structure adjustment while there is rarely research focused on mitigating both the model allocation and model fusion under model heterogeneous settings. In this paper, we will propose a personalized FL framework that obtains local optimal models and performs personalized aggregation simultaneously. Our method involves both the insight of knowledge distillation-based and pruning-based methods while avoiding the disadvantages of previous methods.

III. Background and Preliminaries

In this section, we first introduce the problem setting of personalized FL. Then, we introduce the background knowledge

of differential NAS and why we choose differential NAS for model structure adaption.

A. Formulation of Personalized FL

In this section, we give a formulation of personalized FL. Different from existing personalized FL approaches that request clients to adjust their local optimization objective functions, we only revise the model aggregation settings for both simplicity and efficiency.

We first introduce a FL settings with N clients holding with isolated data sources $\mathcal{D} = \{\mathcal{D}_1, \mathcal{D}_2, \cdots, \mathcal{D}_N\}$. For data source $\mathcal{D}_i$ of client i, $\mathcal{D}_i = \{(\boldsymbol{x}_u, y_u)\}_{u=1}^{|\mathcal{D}_i|}$. We notate the inference loss value of client i as $\mathcal{L}(\theta_i)$. We also notated θ_i as the model parameters from i^{th} client and θ_g as the global model parameters generated by the server. The goal for most of the FL frameworks is to generate a global model which can be formulated as

$$\theta_g = \sum_{i}^{N} p_i \theta_k, \text{ where } \theta_i = \arg\min_{\theta_i} \mathcal{L}_i(\theta_i) \tag{1}$$

where p_i represents the aggregating proportion of client i in global model. For vanilla FL approaches like FedAvg, the aggregating proportion is determined by the volume of local samples and $p_i = |D_i|/|D|$.

The aggregation algorithm that generates one global model no longer works for personalized FL since each client requires a personalized model. To arrive at a trade-off between global parameter sharing and personalization, we first give a problem definition of personalized FL as

$$\boldsymbol{\theta} = \boldsymbol{M} \cdot \hat{\boldsymbol{\theta}}, \text{ where } \hat{\theta}_i = \arg\min_{\theta_i} \mathcal{L}_i(\theta_i) \tag{2}$$

where $\boldsymbol{\theta} = [\theta_1, \theta_2, \cdots, \theta_N]^{\mathrm{T}}$ and $\hat{\boldsymbol{\theta}} = [\hat{\theta}_1, \hat{\theta}_2, \cdots, \hat{\theta}_N]^{\mathrm{T}}$. θ_i is the aggregated personalized models that will be allocated to client i; $\hat{\theta}_i$ is the local model which is trained separately on data from client i; $\boldsymbol{M}$ is a parameter allocation matrix that denotes how clients obtain their personalized models according to training results from the others.

As shown in Equation 2, the goal of personalized FL is to minimize the loss for all clients. Different from the classical FL setting that allocates parameters through sample numbers, personalized FL specifies each client a weight allocation matrix $\boldsymbol{M} = [\boldsymbol{M}_1, \boldsymbol{M}_2, \cdots, \boldsymbol{M}_N]^{\mathrm{T}}$.

B. Differential NAS

In this paper, we utilize the representative differential NAS method, DARTS [28], as the searching backbone. We take a brief glance of details in how differential NAS can solve with the challenge of model heterogeneity.

Differential NAS provides a super-network, which is composed of all candidate sub-networks. The target of differential NAS is to search the optimal sub-network out of the super-network. After representing architecture parameters as α and network weights parameters as ω, differential NAS performs bi-level optimization method to train train network parameters and search network architectures by gradient descent alternately. The bi-level optimization problem is presented as

$$\min_{\alpha} \mathcal{L}_{valid}(\omega^*(\alpha), \alpha) \tag{3}$$

$$s.t.\ \omega^*(\alpha) = \arg\min_{\omega} \mathcal{L}_{train}(\omega, \alpha) \tag{4}$$

By gradient descent optimizing on α, clients are capable to search sub-network architectures which are most suitable with their local data distributions. Since differential NAS performs searching and training on super-network, we can perform coordinate-concerned aggregation with differential NAS even if the clients hold different model architectures since we can aggregate the weight of super-networks directly.

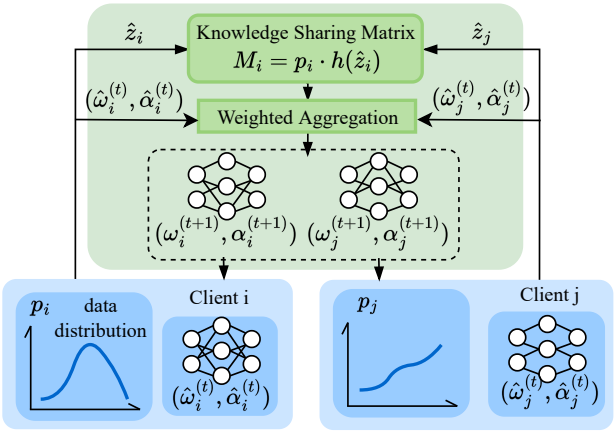


Fig. 2: A workflow overview of FedKMA under the non-IID scenario. In the beginning, FedKMA allocates the same (both parameters and structures) initialized models to clients. In one communication round, the clients update models and inference logits $\hat{\boldsymbol{z}}_i$ to server. Then, the server calculates the knowledge sharing matrix M according to logits, and performs weighted average aggregation to obtain personalized aggregation results for each client.

IV. Personalized FL on Model Structures

In this section, we introduce FedKMA—a FL framework that tackle with the challenges while training on personalized models. Before we go into further details, we will first give an overview of FedKMA. Then, we introduce how FedKMA tackles the challenges while performing personalization on model structures.

A. Overview of FedKMA

An overview of our proposed framework is shown in Figure 2. In FedKMA, clients are allowed to search the optimal model structures on their local data by applying neural architecture search (NAS) techniques; the server will obtain a *knowledge sharing matrix* $\boldsymbol{M}$ (as mentioned in Section III) by using logits

(the inputs to the final softmax) from clients, and aggregate personalized models for clients based on the knowledge sharing matrix.

The server receives models $(\hat{\omega}^{(t)}, \hat{\alpha}^{(t)})$ which are in different structures, and then calculates an irrelevancy matrix H which indicates the difference among clients. Then, the clients receive the irrelevancy matrix and calculate their own knowledge sharing matrix M_i for the convenience of server to generate personalized models $(\hat{\omega}^{(t+1)}, \hat{\alpha}^{(t+1)})$ for them.

Using neural architecture search and knowledge sharing algorithm we proposed, FedKMA are able to achieve continuous model personalization when a new communication round starts. Thus, FedKMA can address the heterogeneous challenges by personalized model structures.

B. Problem Formulation

Now, considering the personalization, we can rewrite the personalized FL problem with model structure adaption as

$$\begin{aligned} \{(\omega_i^*, \alpha_i^*)\}_{i=1}^N &= \arg\min_{\omega_i, \alpha_i} \mathcal{L}_i(\omega_i, \alpha_i) \\ \omega_i &= \boldsymbol{M}_i \hat{\boldsymbol{\omega}} \\ \alpha_i &= \boldsymbol{M}_i \hat{\boldsymbol{\alpha}} \end{aligned} \tag{5}$$

where $\boldsymbol{M}_i$ denotes the knowledge sharing matrix for model parameters and model structure of client i respectively. We use $\hat{\boldsymbol{\omega}}$ and $\hat{\boldsymbol{\alpha}}$ to notate the updated models (parameters and structures, respectively) from clients.

$$\begin{aligned} \hat{\boldsymbol{\omega}} &= [\hat{\omega_1}, \hat{\omega_2}, \cdots, \hat{\omega_N}]^{\mathrm{T}}, \text{ where } \hat{\omega_i} = \arg\min_{\omega_i} \mathcal{L}_i(\omega_i, \alpha_i) \\ \hat{\boldsymbol{\alpha}} &= [\hat{\alpha_1}, \hat{\alpha_2}, \cdots, \hat{\alpha_N}]^{\mathrm{T}}, \text{ where } \hat{\alpha_i} = \arg\min_{\alpha_i} \mathcal{L}_i(\omega_i, \alpha_i) \end{aligned} \tag{6}$$

Thus, FedKMA solves Equation 5 by server and clients collaboratively.

C. Local Model Structure Adaption

We give a solution to the first two challenges of personalized FL with heterogeneous model structures as we mentioned before: (1) how to adjust model structures independently and (2) how to achieve coordinate-wise aggregation given personalized model structures. We propose to use differential neural architecture search (NAS) to address the challenges. NAS is able to obtain an optimal network structure that suits the data distribution most by iterative search and evaluation under certain searching space and data domain. Based on the concept of NAS, differential NAS continuousizes the original discrete network structure searching space and solves it by gradient descent. Thus, differential NAS techniques can get the network parameters (e.g., convolution kernel, etc.) while searching the structures in an efficient way.

Differential NAS encodes the discrete network structure into a continuous parameter which represents the confidence level of the corresponding operator in the network. After finishing the optimization of continuous network encoding, the network structure can be re-discretized by selecting the operator with the maximum confidence level. We can formulate differential NAS by

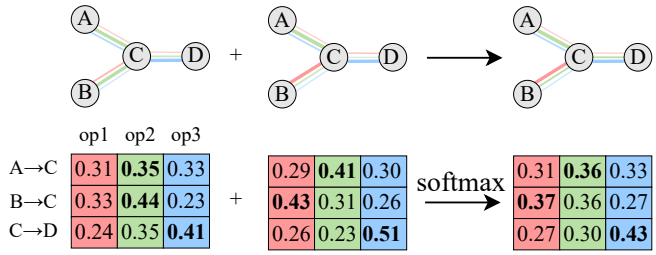


Fig. 3: An illustration of personalized aggregation utilizing neural architecture search. The structure parameter is denoted as a 3×3 matrix where the rows represent edges in networks while the columns represent different operations of the same edge. For each edge, we select the operation with the largest structure weight. We highlight the operations which are eventually chosen in the three model structures.

$$\min_{\omega, \alpha} \mathcal{L}(\omega, \alpha) = \frac{1}{|\mathcal{D}|} \sum_{u=1}^{|\mathcal{D}|} l(x_u, y_u; \omega, \alpha) \tag{7}$$

where ω denotes the network parameter and α denotes the network structure encoding. While applying differential NAS into FL, the problem can be formulated naturally as

$$\min_{\omega_g, \alpha_g} \mathcal{L}(\omega_g, \alpha_g) = \sum_{i=1}^{N} \frac{|\mathcal{D}_i|}{|\mathcal{D}|} \mathcal{L}_i(\omega_g, \alpha_g) \tag{8}$$

where ω_g and α_g denote the global optimal network parameters and network structure encoding.

Thus, the model heterogeneous FL challenge of adjusting local model structures independently is addressed by differential NAS. Meanwhile, differential NAS is also competent with the challenge of coordinate-concerned aggregation. Since the model structure encoding αs among clients have identical shape, the new model structure can be obtained by simply aggregating local α together proportionally according to Equation 1.

D. Knowledge Sharing Algorithm

We propose a knowledge sharing algorithm to help the central server allocate personalized models to clients. The knowledge sharing algorithm enables efficient training results exchange without sharing original local data samples or distributions. As discussed before, we suggest that the server can customize the weight aggregation proportion p for each client.

We denote the knowledge sharing matrix at t^{th} communication round as $\boldsymbol{M}^{(t)} = [\boldsymbol{M}_1^{(t)}, \boldsymbol{M}_2^{(t)}, \cdots, \boldsymbol{M}_N^{(t)}]^{\mathrm{T}}$ where $\boldsymbol{M}_i^{(t)} = [m_{i1}^{(t)}, m_{i2}^{(t)}, \cdots, m_{iN}^{(t)}]$ is the knowledge sharing weight for client i. At the t^{th} communication round, server can aggregate the i^{th} personalized model θ_i with all updated local models $\hat{\boldsymbol{\theta}} = [\hat{\theta}_1, \hat{\theta}_2, \cdots, \hat{\theta}_N]^{\mathrm{T}}$ by

$$\theta_i^{(t)} = \boldsymbol{M}_i^{(t)} \hat{\boldsymbol{\theta}}^{(t)} = m_{i1} \hat{\theta}_1^{(t)} + \cdots + m_{iN} \hat{\theta}_N^{(t)} \tag{9}$$

Thus, the challenge comes with how to obtain $\boldsymbol{M}^{(t)}$ without knowing clients' information like local data distributions. In most of the FL settings, clients update local trained model parameters to the server. However, it is also feasible for clients to submit more results without violating the data privacy constraint so that the server is able to aggregate models with additional hints. To obtain the learning status of each client's local model for different categories, we assume that FL is dealing with a single classification problem with K categories. We use $\hat{\boldsymbol{z}}_{ik}$ to denote the averaged inference logits of the i^{th} client for the k^{th} category of samples. $\hat{\boldsymbol{z}}_{ik}$ can be represented as

$$\hat{\boldsymbol{z}}_{ik} = \mathrm{E}_{x \sim P_i(X, Y=k)}[f(x;\theta)] \tag{10}$$

$P_i(X, Y=k)$ indicates the local data distribution of client i.

Thus, we obtain $\hat{\boldsymbol{z}}_i = [\hat{\boldsymbol{z}}_{i1}, \hat{\boldsymbol{z}}_{i2}, \cdots, \hat{\boldsymbol{z}}_{iK}]$. In neural networks, the averaged inference logits $\hat{\boldsymbol{z}}$ contain the information of how well the model learns samples of a certain category. Instead of sharing sample granularity logits, sharing averaged logits requires less computation and communication costs, and does not need additional public datasets. With $\hat{\boldsymbol{z}}_i$, we are able to explore the relationships between client pairs.

When logits are the aggregated score for each category derived from various information within the model, one can infer whether this model learned how to classify a certain category. Thus, we propose to adopt logits from local models to measure the divergence of the clients' current learning status on different categories of samples. After obtaining the divergence among clients, the server is able to obtain a knowledge sharing matrix $\boldsymbol{M}^{(t)}$. We propose to calculate a knowledge sharing matrix M_i by evaluating the learning status of each client. M_i can be calculated as

$$\boldsymbol{M}_i = g(p_i \cdot h(\hat{\boldsymbol{z}}_i)) \tag{11}$$

$h(\hat{\boldsymbol{z}}_i)$ is generated by $\boldsymbol{H}_i$ while $\boldsymbol{H}_i$ denotes the distances between client i and other clients in the system and we define $g(\cdot)$ as $g(x) = 1/e^{-x}$. To measure the distances, we calculate the irrelevancy matrix $\boldsymbol{H}_i = [\boldsymbol{H}_{i1}, \cdots, \boldsymbol{H}_{iN}], \boldsymbol{H}_i \in \mathbb{R}^{N \times N}$ as

$$\boldsymbol{H}_{ijk} = \lambda D_{KL}(\hat{\boldsymbol{z}}_{ik} || \hat{\boldsymbol{z}}_{jk}) + (1-\lambda) D_{KL}(\hat{\boldsymbol{z}}_{ik} || \boldsymbol{e}_k), \tag{12}$$

where $\boldsymbol{e}_k$ denotes the one-hot label for the k^{th} category and λ denotes a hyper-parameter that balances the clients' similarities and inferring accuracy. We will discuss the value of λ in the evaluation section. Unlike previous efforts that used geometric distance like cosine distance to denote how similar clients are, we give the distance measurement H_{ijk} that computes the relationship of client i and client j on the k^{th} category samples. Equation 12 is represented as a combination of two items, as we measure the client relationships by evaluating (1) how similar clients learn samples from different categories, and (2) how well a client learns the samples from a specific category.

Algorithm 1 Training process of FedKMA

Input: Number of communication rounds T, number of clients N, number of sample categories K. Initialized model $(\omega^{(0)}, \alpha^{(0)})$.

Output: Personalized models $\{\omega_i^{(T)}, \alpha_i^{(T)}\}_{i=1}^{N}$.

Allocate initialized model $(\omega^{(0)}, \alpha^{(0)})$ to clients
for communication round $t = 1, 2, \cdots, T$ **do**
  Server executes:
  for client $i = 1, 2, \cdots, N$ **do** ▷ in parallel
    Receive $(\hat{\omega}_i^{(t)}, \hat{\alpha}_i^{(t)}), \hat{\boldsymbol{z}}_i^{(t)}$
  end for
  for client $i = 1, 2, \cdots, N$ **do**
    for client $j = 1, 2, \cdots, N$ **do**
      $h_{ij} \leftarrow [\lambda D_{KL}(\hat{\boldsymbol{z}}_{ik} || \hat{\boldsymbol{z}}_{jk}) + (1-\lambda) D_{KL}(\hat{\boldsymbol{z}}_{ik} || \boldsymbol{e}_k)]_{k=1}^{K}$
    end for
    Send $\boldsymbol{H}_i = [h_{i1}, \cdots, h_{iN}]$ to client i
  end for
  for client $i = 1, 2, \cdots, N$ **do** ▷ in parallel
    Receive knowledge sharing matrix M_i from client i
    $\omega_i^{(t)}, \alpha_i^{(t)} \leftarrow \boldsymbol{M}_i \hat{\boldsymbol{\omega}}, \boldsymbol{M}_i \hat{\boldsymbol{\alpha}}$
    Send $\omega_i^{(t)}, \alpha_i^{(t)}$ to client i
  end for

  Clients execute: ▷ in parallel
  $\hat{\omega}_i^{(t)}, \hat{\alpha}_i^{(t)} \leftarrow \arg\min_{\omega, \alpha} \mathcal{L}(\omega_i^{(t-1)}, \alpha_i^{(t-1)})$
  Send $(\hat{\omega}_i^{(t)}, \hat{\alpha}_i^{(t)}), \hat{\boldsymbol{z}}_i$ to server
  Receive irrelevancy matrix H_i from server
  $\boldsymbol{M}_i \leftarrow 1/\exp(-p_i \cdot \boldsymbol{H}_i)$
  Send knowledge sharing matrix M_i to server
  Receive personalized aggregated model $(\omega_i^{(t)}, \alpha_i^{(t)})$ from server
end for

However, $\boldsymbol{H}_i$ only conveys whether client i shows a well and similar learning status when focusing on different categories, it does not consider the local data distribution of client i. That is to say, for a client which shows high performance in classifying samples from a specific category, its parameters will still be redundant to other clients who do not own samples of this specific category. Thus, we propose that instead of computing the knowledge sharing weight directly by $\boldsymbol{H}_i$, the clients should combine $\boldsymbol{H}_i$ with its local data distribution to obtain the knowledge sharing matrix. Since we denote the knowledge sharing matrix of client i as $\boldsymbol{M}_i$ and the local data distribution as $p_i = P_i(Y|X=k)$, the knowledge sharing matrix is computed as Equation 11.

V. Evaluation Results

In this section, we first demonstrate the efficiency of the proposed FedKMA with experiments under several benchmark image classification datasets. Then, we specifically present

why knowledge sharing matrix helps improve the personalized aggregation in FedKMA.

A. Experimental Setup

a) Datasets: We consider three image classification datasets including MNIST [7], CIFAR-10, and CIFAR-100 [19]. To generate non-IID in FL settings, we partition three datasets under the guidance of NIID-Bench [22]. We partition datasets by both quantity-skewed method and distribution-skewed method. For quantity-skewed partition method, each client is allocated with data samples of a fixed number of labels. We use $C = k$ to denote the case that each client only has data samples of k different labels. Notice that the smaller the k is, the more skewed the data distributions are. In this paper, we choose $k = 3, 5$ in evaluations on MNIST/CIFAR-10, and choose $k = 30, 50$ in evaluations on CIFAR-100.

For distribution-skewed partition method, each client is allocated with a proportion of the samples of each label according to Dirichlet distribution. We use $\beta = b$ to denote the case that each client is allocated with data samples according to Dirichlet distribution with a concentration parameter β. We perform both quantity-skewed partition and distribution-skewed partition on the three image classification datasets mentioned above. In this paper, we choose $\beta = 0.5, 1$ in evaluations. Notice that the smaller the β is, the more skewed the data distributions are.

b) Models and Implementation Details: We utilize Differential NAS (DARTS) [28] in model architecture adjustment. For fair comparison, we apply networks searched by DARTS in other FL methods without architecture adjustment. We implement all the methods by PyTorch on 4 NVIDIA Tesla V100 GPUs. We set the number of clients at 10. By default, the clients and servers communicate for 50 rounds. On the client side, each client runs 5 epochs per communication round with a batch size of 16. We use momentum SGD for model training and searching with initial learning rate at 0.01, momentum at 0.9, and weight decay at 3×10^{-4}. For the hyper-parameter λ in Equation 12, we set $\lambda = 0.5$ in the experiments.

c) Evaluation Metrics: To evaluate the enhancement of FedKMA in local inference performance, we quantify the performance of methods by computing the inference accuracies on local datasets with respect to each client. We first represent the *average local inference accuracy* (short for ALIA), which indicates the performance of the global system. We also adopt the metric of *best local inference accuracy* for each client (short for BLIA).

B. Comparison with Other Baselines

We first compare FedKMA with local only training manner (no collaboration procedure is performed) and several global FL methods including FedAvg, FedProx [24]. We further compare FedKMA with personalized FL methods including traditional personalized FL methods (FedPer [2], FeSEM [38]) and model heterogeneity personalized FL methods which utilize knowledge distillation-based or pruning-based methods respectively (FedMD [21] and SPIDER [31]).

TABLE I: ALIA (%) achieved by compared FL methods and FedKMA on different datasets under partition-skewed non-IID settings.

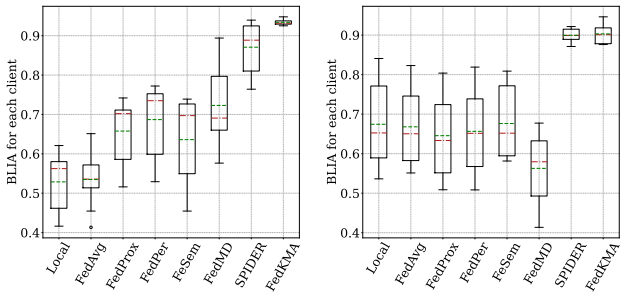
Dataset	MNIST		CIFAR-10		CIFAR-100	
Partition	C=3	C=5	C=3	C=5	C=30	C=50
Local Only	97.11	98.18	52.87	47.16	31.49	48.67
FedAvg	98.02	98.37	53.48	47.74	32.01	51.34
FedProx	97.29	98.79	65.78	44.9	38.72	54.03
FeSEM	97.31	98.18	74.02	82.56	30.69	53.22
FedPer	98.08	98.05	68.69	54.96	35.88	56.06
FedMD	97.95	98.48	70.31	59.33	38.01	57.20
SPIDER	**98.76**	98.92	87.08	89.51	48.10	58.25
FedKMA	98.46	**99.09**	**93.40**	**91.22**	**49.52**	**58.87**

TABLE II: ALIA (%) achieved by compared FL methods and FedKMA on different datasets under distribution-skewed non-IID settings.

Dataset	MNIST		CIFAR-10		CIFAR-100	
Partition	β=0.5	β=1	β=0.5	β=1	β=0.5	β=1
Local Only	95.83	96.11	57.43	58.07	31.58	34.15
FedAvg	95.48	96.22	53.79	47.5	21.01	22.64
FedProx	95.29	95.15	59.55	61.73	19.83	24.07
FeSEM	94.12	93.59	62.63	77.21	25.29	23.71
FedPer	95.87	95.23	60.65	60.49	24.18	27.33
FedMD	96.16	96.87	56.25	59.44	28.01	32.75
SPIDER	98.35	98.26	88.96	89.47	**54.14**	58.97
FedKMA	**98.91**	**99.19**	**89.31**	**91.23**	51.45	**59.38**

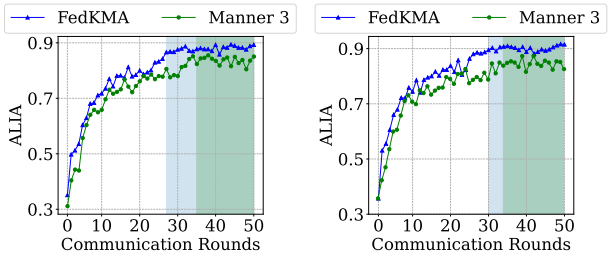
The ALIA of all methods being compared under different non-IID settings are summarized in Table I and Table II. The local only method is a standalone training manner that none of the collaboration procedure is performed. Thus, local only is a baseline to all of the FL methods. However, the global FL methods like FedAvg are defeated by local only under most of the non-IID scenarios. This is because that improper collaboration among clients hinders the convergence on local training, thus leading to poor performance on global FL method. Conversely, the training convergence does not receive negative affects in local only training. This result again demonstrates the inefficiency of aggregating model parameters only based on local sample numbers so that a proper aggregation method is required. Compared with the global FL methods, FedKMA shows outstanding performance while facing different non-IID scenarios. The results indicate the efficiency of personalization, where most of the personalized FL methods achieve improvement in accuracy by up to 20%.

When it comes to personalized FL methods, FedKMA also show outstanding performance. Since FedPer only performs hard parameter sharing among clients, only high-level knowledge is transferred among FL clients. Thus, FedPer shows shortcomings in non-IID settings. Notice that FeSEM shows better performance in quantity-skewed non-IID scenarios than distribution-skewed scenarios. This is because that FeSEM and other personalized FL methods that group clients into several clusters based on a strong assumption that the local data distributions among clients can be partitioned and local



(a) $C = 3$ scenario. (b) $\beta = 0.5$ scenario.

Fig. 4: The visualization of BLIA of each client for FedKMA and other baseline methods on CIFAR-10 with two non-IID settings.



(a) $C = 3$ scenario. (b) $\beta = 0.5$ scenario.

Fig. 5: The convergence curve of FedKMA and Manner 3 under different non-IID settings. The blue and green shades indicate the convergence communication rounds of FedKMA and Manner 3 respectively.

TABLE III: Results on ablation study of FedKMA. The baseline infers the vanilla FL training manner, while Manner 2 and 3 are FL training manners without model structure adaption/knowledge sharing algorithm.

	C=3		$\beta = 0.5$	
Baseline	53.48	47.74	53.79	47.5
Manner 2	58.06	50.93	55.12	49.61
Manner 3	87.08	89.51	88.96	89.47
FedKMA	93.4 (**+39.92**)	91.22 (**+43.48**)	89.31 (**+35.52**)	50.93 (**+42.65**)

distributions remain IID inside the subsets. Also, clustering clients according to gradients/parameters similarity not always works [34], [37]. Thus, FeSEM fails when it comes to distribution-skewed scenarios that no significant clustered data distributions exist. When it comes to model heterogeneous personalized FL methods including FedMD and SPIDER, FedKMA outperforms up to 32.8% on ALIA.

Specifically, we take a discussion on the communication overhead between FedKMA and FedMD. As these two methods both require clients to send local inference logits to the server, FedKMA saves the communication costs since FedKMA only requires the averaged inference logits on each category, while FedMD needs the inference logits of each data sample in public datasets. The communication cost of FedKMA is $K \times K$ while FedMD is $K \times |D|$ (K and $|D|$ indicates the number of data categories and samples of public datasets respectively, notice that $|D| \gg K$).

We also examine the BLIA for each client in all methods. To test the best inference performance of each method in details, we visualize the BLIA performance for clients in Figure 4. Compared with other global and personalized FL methods, FedKMA owns clients with higher averaged BLIA and lower variance on CIFAR-10 under different non-IID scenarios.

C. Ablation Study

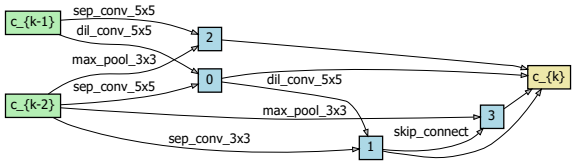
We perform an ablation study to verify the efficiency of the two main components of FedKMA—model architecture adaption and knowledge sharing algorithm. We obtain the following FL manners: (1) without neither model structure adaption nor the knowledge sharing algorithm (Baseline), (2) without model structure adaption (Manner 2), (3) without the knowledge sharing algorithm (Manner 3), and (4) the full design of FedKMA. Table III shows the results of applying the above four training manners under different non-IID partitions. Compared with FL manners that without model heterogeneous settings, the design of letting clients adapt local model structures highly improves the performance. Moreover, the addition of the knowledge sharing algorithm is also effective in improving performance. Manner 2 improves ALIA up to 8% over the baseline method and FedKMA improves the ALIA by 5% over Manner 3.

Furthermore, we also compare the difference in convergence time to measure whether the knowledge sharing algorithm is able to reduce the local fine-tuning time, thus achieving a higher convergence rate. In Figure 5 we visualize the ALIA curve under different non-IID settings. The blue shades in Figure 5 denote the round of FedKMA convergence while the green shades denote that Manner 3 converges. Since the blue shades always covered the green shades, we can observe that compared with Manner 3, our proposed FedKMA converges earlier.

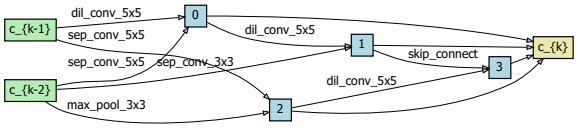
D. How FedKMA Achieves Personalization?

We give further details about how the two solutions above help FedKMA achieve personalization and provide performance improvement contribution.

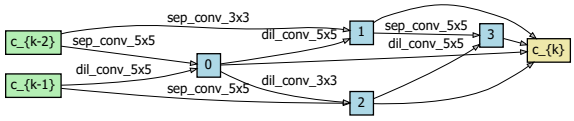
a) Efforts of Generating Different Structures: We observe the local model structures with local data distribution respectively when training on CIFAR-10. For different data distribution, FedKMA generates heterogeneous model structures. We show that FedKMA performs the capability of local model adaption with local data distributions. We observe a few of the local model structures with local data distribution respectively when training on CIFAR-10. As we shown in Figure 6 For different data distributions, FedKMA generates heterogeneous model structures: for more complex distributions (i.e., data with more categories), the personalized model structure tends to be deeper with more complex operations



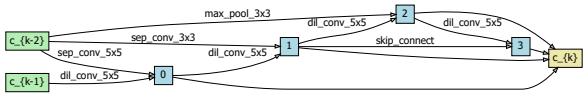
(a) Client 2



(b) Client 3

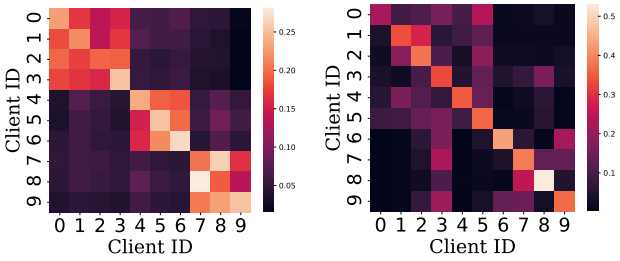


(c) Client 4



(d) Client 5

Fig. 6: The personalized model structures generated by proposed FedKMA.



(a) $C = 3$ scenario. (b) $\beta = 0.5$ scenario.

Fig. 7: The visualization of the knowledge sharing matrix M under two non-IID settings on CIFAR-10. The x-axis and y-axis show the client IDs respectively.

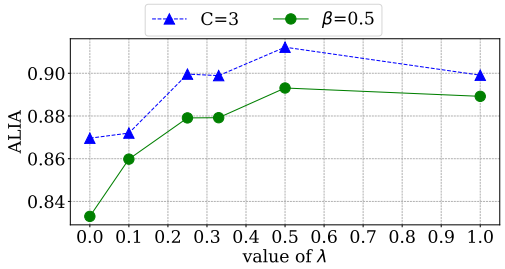


Fig. 8: The ALIA obtained from different λ values under $C = 3$ and $\beta = 0.5$ distribution scenarios.

like convolution; for simpler distributions (i.e., data with less categories), the model structure tends to be more shallow with simper operations like pooling.

b) The Efficiency of Knowledge Sharing Matrix: Although FedKMA utilizes neural architecture search in local training, the model heterogeneity is also based on the personalized aggregation. We evaluate the efficiency of knowledge sharing matrix when FedKMA meets different non-IID scenarios. We visualize the knowledge sharing matrix under $C = 3$ and $\beta = 0.5$ non-IID scenario with 10 clients.

From Figure 7a we can discover that in $C = 3$ scenario, the knowledge sharing matrix captures the relationships among client pairs accurately. Since we partition clients into three groups with similar data distribution (within the same group, clients share samples from the same categories), we could see three blocks in the matrix just as how we divide the clients. Additionally, for $\beta = 0.5$ scenario in Figure 7b, unlike other personalized FL approaches which allocated the largest weight for clients themselves, FedKMA also captures inherent similarity among distributions.

c) Selection of Hyper-parameter λ: In Equation 12, we set a scaling hyper-parameter λ that balances the clients similarities and inferring accuracy. We evaluate the influence of λ by tuning it in the set $\{0, 0.01, 0.25, 0.33, 0.5, 1\}$ on CIFAR-10 with $C = 3$ and $\beta = 0.5$ scenarios since these two settings exhibit more significant heterogeneity. We illustrate the ALIA in Figure 8. It can be observed that a larger λ ($\lambda = 0.5, 1$) gives better results in both of the heterogeneous scenarios. From these results, we can infer that the similarity of learning status among clients is important information for knowledge sharing algorithm.

VI. Conclusion

In this paper, we proposed FedKMA, a novel personalized FL framework with knowledge sharing-based model structure adaption. FedKMA allows clients to train and infer on personalized models with heterogeneous structures. We showed that FedKMA achieves enhancement on local inference performance for every client in the system since the knowledge sharing matrix captures the relationships between client pairs so that an efficient aggregation is well performed. Compared with other personalized FL approaches, experiment results on several benchmark datasets demonstrate that FedKMA not only outperforms the existing state-of-the-art in accuracy significantly but also achieves a higher convergence rate.

REFERENCES

[1] Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. Federated learning with personalization layers. *CoRR*, abs/1912.00818, 2019.

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

[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 2020, Glasgow, United Kingdom, July 19-24, 2020*, pages 1–9. IEEE, 2020.

[4] Hongyan Chang, Virat Shejwalkar, Reza Shokri, and Amir Houmansadr. Cronus: Robust and heterogeneous collaborative learning with black-box knowledge transfer. *CoRR*, abs/1912.11279, 2019.

[5] Gary Cheng, Karan Chadha, and John Duchi. Fine-tuning is fine in federated learning. *arXiv preprint arXiv:2108.07313*, 2021.

[6] Luca Corinzia, Ami Beuret, and Joachim M Buhmann. Variational federated multi-task learning. *arXiv preprint arXiv:1906.06268*, 2019.

[7] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[8] Enmao Diao, Jie Ding, and Vahid Tarokh. Heterofl: Computation and communication efficient federated learning for heterogeneous clients. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[9] Canh T. Dinh, Nguyen Hoang Tran, and Tuan Dung Nguyen. Personalized federated learning with moreau envelopes. 2020.

[10] Moming Duan, Duo Liu, Xinyuan Ji, Yu Wu, Liang Liang, Xianzhang Chen, Yujuan Tan, and Ao Ren. Flexible clustered federated learning for client-level data distribution shift. *IEEE Transactions on Parallel and Distributed Systems*, 2021.

[11] Alireza Fallah, Aryan Mokhtari, and Asuman E. Ozdaglar. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. 2020.

[12] Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An efficient framework for clustered federated learning. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[13] Chaoyang He, Murali Annavaram, and Salman Avestimehr. Fednas: Federated deep learning via neural architecture search. *CoRR*, abs/2004.08546, 2020.

[14] Wenke Huang, Mang Ye, and Bo Du. Learn from others and be yourself in heterogeneous federated learning. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10133–10143, 2022.

[15] 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 *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Virtual Event, February 2-9, 2021*, pages 7865–7873. AAAI Press, 2021.

[16] 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.

[17] Yihan Jiang, Jakub Konečný, Keith Rush, and Sreeram Kannan. Improving federated learning personalization via model agnostic meta learning. *CoRR*, abs/1909.12488, 2019.

[18] Jakub Konečný, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *CoRR*, abs/1610.05492, 2016.

[19] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical Report 0, University of Toronto, Toronto, Ontario, 2009.

[20] Ang Li, Jingwei Sun, Pengcheng Li, Yu Pu, Hai Li, and Yiran Chen. Hermes: an efficient federated learning framework for heterogeneous mobile clients. In *ACM MobiCom '21: The 27th Annual International Conference on Mobile Computing and Networking, New Orleans, Louisiana, USA, October 25-29, 2021*, pages 420–437. ACM, 2021.

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

[22] Qinbin Li, Yiqun Diao, Quan Chen, and Bingsheng He. Federated learning on non-iid data silos: An experimental study. In *38th IEEE International Conference on Data Engineering, ICDE 2022, Kuala Lumpur, Malaysia, May 9-12, 2022*, pages 965–978. IEEE, 2022.

[23] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

[24] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. 2020.

[25] 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. *CoRR*, abs/2001.01523, 2020.

[26] Wei Yang Bryan Lim, Nguyen Cong Luong, Dinh Thai Hoang, Yutao Jiao, Ying-Chang Liang, Qiang Yang, Dusit Niyato, and Chunyan Miao. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 22(3):2031–2063, 2020.

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

[28] Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: differentiable architecture search. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[29] Yang Liu, Yi Zhao, Guangmeng Zhou, and Ke Xu. Fedprune: Personalized and communication-efficient federated learning on non-iid data. In Teddy Mantoro, Minho Lee, Media Anugerah Ayu, Kok Wai Wong, and Achmad Nizar Hidayanto, editors, *Neural Information Processing - 28th International Conference, ICONIP 2021, Sanur, Bali, Indonesia, December 8-12, 2021, Proceedings, Part V*, volume 1516 of *Communications in Computer and Information Science*, pages 430–437. Springer, 2021.

[30] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In Aarti Singh and Xiaojin (Jerry) Zhu, editors, *Artificial Intelligence and Statistics Conference (AISTATS)*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR, 2017.

[31] Erum Mushtaq, Chaoyang He, Jie Ding, and Salman Avestimehr. SPIDER: searching personalized neural architecture for federated learning. *CoRR*, abs/2112.13939, 2021.

[32] Felix Sattler, Arturo Marbán, Roman Rischke, and Wojciech Samek. CFD: communication-efficient federated distillation via soft-label quantization and delta coding. *IEEE Transactions on Network Science and Engineering*, 9(4):2025–2038, 2022.

[33] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE Transactions on Neural Networks Learning System (TNNLS)*, 32(8):3710–3722, 2021.

[34] Sidak Pal Singh and Martin Jaggi. Model fusion via optimal transport. 2020.

[35] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet Talwalkar. Federated multi-task learning. pages 4424–4434, 2017.

[36] Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. Towards personalized federated learning. *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)*, 2022.

[37] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[38] Ming Xie, Guodong Long, Tao Shen, Tianyi Zhou, Xianzhi Wang, Jing Jiang, and Chengqi Zhang. Multi-center federated learning. *CoRR*, abs/2108.08647, 2021.

[39] Yuhui Xu, Lingxi Xie, Xiaopeng Zhang, Xin Chen, Guo-Jun Qi, Qi Tian, and Hongkai Xiong. PC-DARTS: partial channel connections for memory-efficient architecture search. 2020.

[40] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Trans. Intell. Syst. Technol.*, 10(2):12:1–12:19, 2019.

[41] Jie Zhang, Song Guo, Xiaosong Ma, Haozhao Wang, Wenchao Xu, and Feijie Wu. Parameterized knowledge transfer for personalized federated learning. pages 10092–10104, 2021.

[42] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. Learning transferable architectures for scalable image recognition. pages 8697–8710, 2018.