AGNOSTIC PERSONALIZED FEDERATED LEARNING WITH KERNEL FACTORIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Considering the futuristic scenarios of federated learning at a worldwide scale, it is highly probable that local participants can have their own personalized labels, which might not be compatible with each other even for the same class, and can be also possibly from a variety of multiple domains. Nevertheless, they should be benefited from others while selectively taking helpful knowledge. Toward such extreme scenarios of federated learning, however, most existing approaches are limited in that they often assume: (1) labeling schemes are all synchronized amongst clients; (2) the local data is from the same single dataset (domain). In this sense, we introduce an intensively realistic problem of federated learning, namely *Agnostic Personalized Federated Learning (APFL)*, where any clients, regardless of what they have learned with their personalized labels, can collaboratively learn while benefiting each other. We then study two essential challenges of the agnostic personalized federated learning, which are (1) *Label Heterogeneity* where local clients learn from the same single domain but labeling schemes are not synchronized with each other and (2) *Domain Heterogeneity* where the clients learn from the different datasets which can be semantically similar or dissimilar for each other. To tackle these problems, we propose our novel method, namely *Similarity Matching and Kernel Factorization (SimFed)*. Our method measures semantic similarity/dissimilarity between locally learned knowledge and matches/aggregates the relevant ones that are beneficial to each other. Furthermore, we factorize our model parameters into two basis vectors and the sparse masks to effectively capture permutation-robust representations and reduce information loss when aggregating the heterogeneous knowledge. We exhaustively validate our method on both single- and multi-domain datasets, showing that our method outperforms the current state-of-the-art federated learning methods.

1 INTRODUCTION

Federated Learning (FL) aims to learn the universal global model while merging locally-learned knowledge from multiple clients rather than collecting the local data directly to the central server (McMahan et al., 2017; Li et al., 2020b; Wang et al., 2020; Karimireddy et al., 2021; Li et al., 2021). Although such decentralized features of federated learning have various advantages, such as preservation of the data privacy, reduction of computing and data maintenance costs, etc., it at the same time shows significant vulnerability in terms of the local heterogeneity. When the local data is not independent and identically distributed (Non-IID), the locally-learned knowledge might not be mutually helpful depending on the level of distributional shifts, yielding performance degradation of the universal model, which is considered as the critical challenge of federated learning.

One of the successful solutions to such problem is *Personalized Federated Learning (PFL)*, whose main goal is to adapt individual clients to their own local distribution better, rather than learning the single universal model to generalize over all local heterogeneous distributions, which might be sub-optimal (Arivazhagan et al., 2019; Liang et al., 2020; Fallah et al., 2020; Zhang et al., 2021). For example, FedFOMO (Zhang et al., 2021), which is the current state-of-the-art approach, downloads a few set of random other clients' knowledge and then selectively takes only beneficial knowledge that are performing well on the local obejective, which is different form the conventional federated learning methods that often aggregate all local knowledge. Since such personalized federated learning

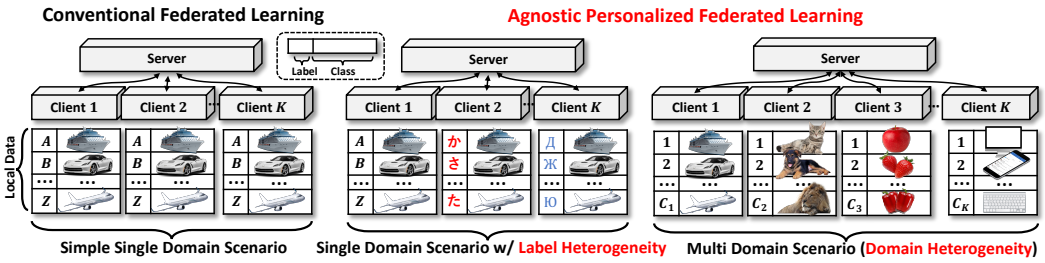


Figure 1: **Overview of the Conventional and Agnostic Personalized Federated Learning:** ***Left*** illustrates the conventional federated learning with K clients. ***Middle*** represents scenario of personalized targets, where labels are not synchronized across all clients. ***Right*** shows the multi-domain scenario, where each local client learns K different datasets.

approaches are client-oriented, they have shown stronger robustness compared to the the traditional federated learning approaches in handling the local heterogeneity at each local device.

Toward further realistic scenarios of federated learning at world-wide scale, however, most existing methods are limited in that they often suppose that (1) target labels are the exactly same across all clients and (2) local data is from a certain single dataset (domain). For the former supposition, all clients share the synchronized labeling schemes in the single domain scenarios (see Figure 1 Left), while it is more natural that labels are slightly different depending on the user environment (see Figure 1 Middle). For example, diverse labels, such as "Cat", "Kitty", or even "Neko" in Japanese, can be used for the single "Cats" class. Furthermore, both index and number of classes can vary per client, such as {1:Cats, 2:Dogs, . . . , 10:Horse} for some clients, {1:Horses, 2: Cats, . . . , 7: Dogs} for the other clients. We find out that such *Label Heterogeneity* is highly non-trivial problem in that the most current approaches are significantly degenerated when the target labels are not synchronized each other (see Table 1).

For the latter assumption, majority of existing methods mainly tackle distributional shifts, e.g. Non-IID, sampled by a certain distribution, e.g. Dirichlet dist., from the same single dataset. However, considering multiple domains is highly essential (see Figure 1 Right), since sharing knowledge even between the heterogeneous domains can be beneficial to each other depending on the semantic similarity or dissimilarity amongst the local domains. For instance, when we have to quickly learn COVID-19 with scarce data, utilizing the mature knowledge of Pneumonia can be more powerful than only learning from COVID-19, while some irrelevant knowledge may harm. We also find that discriminating which domain knowledge is beneficial to which domain is also very difficult, as the existing methods fail to handle such *Domain Heterogeneity* (see Table 2).

To this end, we introduce an extremely realistic problem of federated learning, namely ***Agnostic Personalized Federated Learning (APFL)***, where any clients, regardless of what they have learned with their personalized labels, can collaboratively learn while benefiting each other. We then study two critical challenges of it: (1) ***Label Heterogeneity*** which labeling schemes are not synchronized for each client; (2) ***Domain Heterogeneity*** which each client learns non-identical datasets. To tackle the problems, we propose a novel method, namely ***Similarity Matching and Kernel Factorization (SimFed)***. Our method measures semantic similarity/dissimilarity between locally-learned knowledge and aggregates the relevant knowledge that are beneficial to each other. Furthermore, we factorize our model parameters into two basis vectors and the sparse masks to effectively capture permutation-robust representations and reduce information loss when aggregating the heterogeneous knowledge. We exhaustively validate our method on both singl domain datastes (Permuted IID/Non-IID) and multi domain datasets (CIFAR-100 20 superclasses) and show our method outperforms the current state-of-the-art approaches on all experiments.

In sum, the main contributions of this work are as follow:

- We introduce a realistic problem of ***Agnostic Personalized Federated Learning (APFL)***, and discuss two possible critical challenges of it, namely ***Label-*** and ***Domain-Heterogeneity***.
- We propose a novel method to tackle the problems, namely ***Similarity Matching and Kernel Factorization (SimFed)***, which utilizes the semantic similarity and factorizes network parameters for permutation robustness and handling the heterogeneous knowledge.
- We extensively validate our method in both label- and domain-heterogeneous scenarios and show that our method outperforms the current state-of-the-art baselines.

2 Related Work

Federated Learning A variety of algorithms have been proposed for federated learning after firstly introduced by McMahan et al. (2017), but we specifically focus on studies tackling the heterogeneity problems, e.g. Non-IID. FedProx (Li et al., 2020b) intorduces proximal regularization to enforce locally-learned knowledge to be close to the global knowledge. FedMA (Wang et al., 2020) learns the layer-wise permutation when averaging weights, while SCAFFOLD (Karimireddy et al., 2021) utilizes control variates to adjust for the client-drift in the local updates. MOON (Li et al., 2021) performs model-level contrastive learning between global and local models. In this paper, as we are pursuing the task-level heterogeneity, we do not cover the architecture-level heterogeneity (Seo et al., 2020; Zhu et al., 2021; Diao et al., 2021; Shamsian et al., 2021).

Personalized Federated Learning aims to improve the individual local clients instead of learning the universal global model via clustering-based approaches (Sattler et al., 2019; Ghosh et al., 2021), the mixture methods (Mansour et al., 2020; Deng et al., 2020; Hanzely & Richtárik, 2021), or meta-learning approach (Fallah et al., 2020). FedPer (Arivazhagan et al., 2019) divides local models into base and personalized layers and shares only base layers, while LG-FedAvg (Liang et al., 2020) instead shares the prediction layers with the others and keeps the base layers locally. FedFOMO (Zhang et al., 2021) is the current state of the art method, which downloads a few random other clients' knowledge and accepts only helpful knowledge on their own local objective.

Toward Multiple Domains Beyond learning from the single domain, several recent studies have been proposed to achieve the universal global model well-generalized over multiple domains via the mixture of the client target distributions (Mohri et al., 2019), a distributed multi-task-learning framework (Smith et al., 2018; Li et al., 2020a), domain adaptation for unlabeled data (Peng et al., 2020; Yao et al., 2021). All these prior works, which aim to learn the global model from the multiple domains, however, are orthogonal to our direction, which is to adapt the individual client to their own local domain well that is mainly based on personalized federated learning approaches.

3 Problem Definition

We begin with the formal definition of the conventional federated learning and then describe *Agnostic Personalized Federated Learning (APFL)* and define two essential scenarios for *Label Heterogeneity* and *Domain Heterogeneity*, respectively.

3.1 Preliminaries

Let M be the global model, $\mathcal{M} = \{m_k\}_{k=1}^{K}$ be a set of K local models, and $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^{N}$ be a given dataset, where K is the number of local models, N is the number of instances, $\mathbf{x}_i \in \mathbb{R}^{W \times H \times D}$ is ith examples in a size of width W, height H, and depth D, with a corresponding target class $y_i \in \{1, \dots, C\}$ for the C-way multi-class classification problem. The given dataset $\mathcal{D}$ is then disjointly split into K sub-partitions $\mathcal{P}_k = \{\mathbf{x}_{k,i}, y_{k,i}\}_{i=1}^{N_k}$ s.t. $\mathcal{D} = \bigcup_{k=1}^{K} \mathcal{P}_k$, which are distributed to the corresponding local model m_k. Let R be the total number of the communication rounds and r be the current round index. At the first round r=1, the global model M initialize the global weights $\theta_M^{(1)}$ and broadcasts $\theta_M^{(1)}$ to an arbitrary subset of local models that are available for training at round r, such that $\mathcal{M}^{(r)} \subset \mathcal{M}$, $|\mathcal{M}^{(r)}| = K^{(r)}$, and $K^{(r)} \leq K$, where $K^{(r)}$ is the number of available local models at round r. Then the active local models $m_k \in \mathcal{M}^{(r)}$ perform local training to minimize loss $\mathcal{L}(\theta_k^{(r)})$ on the corresponding sub-partition $\mathcal{P}_k$ and update their local weight $\theta_k^{(r+1)} \leftarrow \theta_k^{(r)} - \eta \nabla \mathcal{L}(\theta_k^{(r)})$, where $\theta_k^{(r)}$ is local weights for local model m_k at round r and $\mathcal{L}(\cdot)$ is the loss function to minimize locally. When the local training is done, the global model M collects and aggregates the learned weights $\theta_M^{(r+1)} \leftarrow \frac{N_k}{N} \sum_{i=1}^{K^{(r)}} \theta_k^{(r)}$ and then broadcasts newly updated weights to the local models available at the next round $r+1$. These learning procedures are repeated until the final round R, and this entire framework refers to as *Federated Learning*. On the other hand, *Personalized Federated Learning* aims to adapt the individual local models $m_{1:K}$ to their local data distribution $\mathcal{P}_{1:K}$ better rather than achieving the globally universal model M over all local distribution, and thus merging local knowledge is not necessarily in a form of $\theta_M^{(r+1)} \leftarrow \frac{N_k}{N} \sum_{i=1}^{K^{(r)}} \theta_k^{(r)}$ depending on the algorithms.

3.2 AGNOSTIC PERSONALIZED FEDERATED LEARNING

Agnostic Personalized Federated Learning is personalized federated learning where any local participants from diverse domains with their own personalized labeling schemes can collaboratively learn while benefiting each other. To achieve such objective, following two critical challenges should be tackled: (1) *Label Heterogeneity* and (2) *Domain Heterogeneity*.

Scenario for Label Heterogeneity This scenario assumes that the labeling schemes are not perfectly synchronized across all clients, as described in Section 1 and Figure 1 Middle. Most underlying setting for this scenario is the same as the conventional single-domained setting described in Section 3.1, except that labels are arbitrarily permuted amongst clients. The local data $\mathcal{P}_k$ for the local model m_k is now defined as $\mathcal{P}_k = \{\mathbf{x}_{k,i}, \varphi_k(y_{k,i})\}_{i=1}^{N_k}$, where $\varphi_k(\cdot)$ is a mapping function for local model m_k that maps a given class $y_{k,i}$ with a randomly permuted label $p_{k,i}$=$\varphi_k(y_{k,i})$. Let the jth layer out of L layers in the neural networks of local model m_k be ℓ_k^j and the last layer ℓ_k^L be a classifier layer. Now that each client has differently permuted labels, the personalized classifiers $\ell_{1:K}^L$ are no longer compatible each other, and thus they are enforced to be remained locally and not undesirably merged for the fairness of this label-heterogeneous scenario.

Scenario for Domain Heterogeneity This scenario presumes that local clients learn on their own entire datasets $\mathcal{D}$, rather than partitions $\mathcal{P}_k$ from the single dataset, as described in Section 1 and Figure 1 Right. In this setting, K disjoint datasets $\mathcal{D}_{1:K}$ are assigned to the K local clients $m_{1:K}$, where $\mathcal{D}_k = \{\mathbf{x}_{k,i}, y_{k,i}\}_{i=1}^{N_k}$ is the entire dataset assigned to the local model m_k, where the number of target classes are possibly different, such that $y_{k,i} \in \{1, \dots, C_k\}$. Between the datasets, there is *no instance-wise and class-wise overlapping*, such that $\varnothing = \bigcap_{k=1}^{K} \mathcal{D}_k$. Similarly to the label-heterogeneous scenario described above, each client has disjoint target classes, and thus the personalized classifiers $\ell_{1:K}^L$ are no longer compatible each other. Hence, they are enforced to be remained locally and not shared each other for the fairness of this domain-heterogeneous scenario.

4 SIMFED: SIMILARITY MATCHING AND KERNEL FACTORIZATION

We now elaborate on in-depth description of our novel algorithm for agnostic personalized federated learning, namely *Similarity Matching and Kernel Factorization (SimFed)* in the following subsections.

4.1 SIMILARITY MATCHING

Our underlying assumption is that we can discriminate which local models are helpful or harmful each other if we can differentiate which local data distributions are semantically relevant or irrelevant. That is, if we can successfully encode all the heterogeneous local data distributions into a certain latent space, then we can measure similarity between the local domains in the embedding space, which eventually enables us to determine which models are beneficial each other accordingly. In order to do this, the most important part is how to encode the local data distributions to achieve the latent space.

Instead of adding any secondary models for that, our simple-yet-effective idea is to view the individual local models as respective set encoders that can effectively transform their local data distribution to their own personalized embedding vectors. If we equally feed the same inputs to all encoders, then they will interpret the criteria inputs depending on what they have learned in their own local environments and produce their own personalized transformations, which we can efficiently utilize them as representations of local distributions to measure the relevance.

Formally describing, the neural networks f_k of local model m_k can be represented as:

$$f_k(\hat{\mathbf{y}}|\mathbf{x}) = P_k(\hat{\mathbf{y}}|E_k(\mathbf{z}|\mathbf{x})) \tag{1}$$

where f_k is a combination of an encoder $E_k(\mathbf{z}|\mathbf{x})$ and a predictor $P_k(\hat{\mathbf{y}}|\mathbf{z})$, where $\mathbf{z} \in \mathbb{R}^d$ is d-length transformed latent vector of input data $\mathbf{x}$ and $\hat{\mathbf{y}} \in \mathbb{R}^C$ is a logit for the final prediction. We generate the unbiased criteria input $\mathbf{x}_{\mathcal{N}} \in \mathbb{R}^{W \times H \times D}$ from the Gaussian normal distribution, such that $x_i \sim \mathcal{N}(0, 1)$ where x_i is its element, and assume that it is located at the server. Since we have

K local models $m_{1:K}$, we equally feed the single criteria input $\mathbf{x}_{\mathcal{N}}$ to K local encoders so that we can generate a set of K multiple personalized transformations $\mathcal{Z}_{\mathcal{N}}$ as follows:

$$\mathcal{Z}_{\mathcal{N}} = \{\mathbf{z}_{\mathcal{N},k} | \mathbf{z}_{\mathcal{N},k} = E_k(\mathbf{z}|\mathbf{x}_{\mathcal{N}})\}_{k=1}^{K} \tag{2}$$

Now our remaining task is to compute the relevancy between the transformations in $\mathcal{Z}_{\mathcal{N}}$ which each local distribution is efficiently encoded. We use the cosine similarity to measure the distances between target model m_k with the rest of models in $\mathcal{M}$, as follows:

$$\Omega(k, \tau, \mathcal{Z}_{\mathcal{N}}, \Theta) = \{(\sigma_i, \theta_i) | \sigma_i = \frac{\mathbf{z}_k \cdot \mathbf{z}_i}{\|\mathbf{z}_k\| \|\mathbf{z}_i\|}, \sigma_i \geq \tau, \mathbf{z}_i \in \mathcal{Z}, \theta_i \in \Theta\}_{i=1}^{K} \tag{3}$$

where Θ is a set of K local weights maintained at the server and $\Omega(\cdot)$ is a similarity function that returns a set of pairs of the similarity score σ_i and the corresponding local weights θ_i for all $i \in \{1, 2, \ldots, K\}$, while satisfying the similarity scores are greater or equal to the given threshold τ.

4.2 Kernel Factorization

In most federated learning approaches, merging local knowledge from multiple clients is typically computed as $\theta^{(r+1)} \leftarrow \frac{N_k}{N} \sum_{i=1}^{K^{(r)}} \theta_k^{(r)}$. This naive aggregation might work if each local updates share common features explicitly, for example, in cases of learning from the same single domain across all clients. However, when each local model learns different domains that are heterogeneous each other in certain degrees, such simple mixture of knowledge can be highly problematic, which even could lead to *catastrophic forgetting* in the worst case. This is because the parameter space is significantly high-dimensional even for the simple neural networks, and thus the meaningful element-wise neural matching is not guaranteed, causing the critical loss of essential information.

We assume that such problem can be alleviated if the dimensionality of parameter space to compute is significantly reduced as well as knowledge is separately captured to merge them more effectively while minimizing information loss. In order to do that, our novel idea is to factorize the high order model parameters into lower rank components, hopefully rank 1 vectors. Formally, local model m_k has local weights θ_k that is a set of weights of its layers, such that $\theta_k = \{\mathbf{w}_k^i\}_{i=1}^{L}$. The type of each layer $\mathbf{w}_k^i$ then typically split into fully-connected or convolutional layers.

Factorization of Fully-connected Layers The dimensionality of the dense weight $\mathbf{w}_k^i$ for the fully-connected layers is drastically increased by multiplication of input and output dimensions, such that $\mathbf{w}_k^i \in \mathbb{R}^{I \times O}$, where I and O indicate respective input and output dimensions. We can reduce the $I \times O$ complexity into rank 1 complexity via factorizing the high order matrix as follows:

$$\mathbf{w}_k^i = \mathbf{u} \times \mathbf{v}, \text{where } \mathbf{u} \in \mathbb{R}^{I \times 1}, \mathbf{v} \in \mathbb{R}^{1 \times O} \tag{4}$$

This way, we can induce the basis vector parameters $\mathbf{u}$ to capture the I-length input-related knowledge and $\mathbf{v}$ to learn the O-length output-related knowledge while significantly reducing the complexity. However, the multiplication of the two vectors interact only in a linear manner, it might be hard to capture some non-linear knowledge with just two of them. As such, we add *the light-weighted sparse mask* $\boldsymbol{\mu}$ to the weights to additionally capture non-linearity effectively and efficiently, as follows:

$$\mathbf{w}_k^i = \mathbf{u} \times \mathbf{v} \oplus \boldsymbol{\mu}, \text{where } \mathbf{u} \in \mathbb{R}^{I \times 1}, \mathbf{v} \in \mathbb{R}^{1 \times O}, \boldsymbol{\mu} \in \mathbb{R}^{I \times O} \tag{5}$$

Factorization of Convolutional Layers The difference between fully-connected layers and convolutional layers is that the convolutional layers have multiple kernels (or filters) such that $\mathbf{w}_k^i \in \mathbb{R}^{F \times F \times I \times O}$, where F is a size of filters. Similarly to the factorization of fully-connected layer, we intend to capture input- and output-related knowledge separately and additional non-linearity as well for the convolutional layers, as follows:

$$\begin{aligned} \mathbf{w}_k^i = \pi(\mathbf{u}' \times \mathbf{v}' \oplus \boldsymbol{\mu}'), \text{where } \mathbf{u}' \in \mathbb{R}^{I \cdot F \times 1}, \mathbf{v}' \in \mathbb{R}^{1 \times O \cdot F}, \boldsymbol{\mu}' \in \mathbb{R}^{I \cdot F \times O \cdot F}, \\ \pi(\cdot) : \mathbb{R}^{I \cdot F \times O \cdot F} \rightarrow \mathbb{R}^{F \times F \times I \times O} \end{aligned} \tag{6}$$

By doing so, we can still factorize the weights of convolutional layers into rank 1 vector parameters with the highly sparse masks to support the non-linearity. Note that we factorize our parameters *before training* and thus the weights of each layer $\mathbf{w}_k^i$ act like the regular weights during training, while learning the basis vectors and the masks separately. Then we handle each components respectively when sharing them with others, which will be describe in the next subsection.

4.3 SIMFED ALGORITHM

Putting the novel methods altogether, we now describe our objective functions with the factorized parameters, personalized knowledge reflection, and overall learning procedure in this subsection.

Learning Objective Instead of utilizing the single term θ_k for local weights of neural network f_k, now let ζ_k, ξ_k, and ϕ_k be sets of $\mathbf{u}_k$, $\mathbf{v}_k$, and $\boldsymbol{\mu}_k$ of all layers in f_k, s.t. $\zeta_k = \{\mathbf{u}_k^i\}_{i=1}^L$, $\xi_k = \{\mathbf{v}_k^i\}_{i=1}^L$, and $\phi_k = \{\boldsymbol{\mu}_k^i\}_{i=1}^L$, then our local objective function can be defined as follows:

$$\min_{\zeta_k,\xi_k,\phi_k} \sum_{\mathcal{B}\in\mathcal{D}_k} \mathcal{L}(\mathcal{B};\zeta_k,\xi_k,\phi_k) + \lambda_1||\boldsymbol{\phi}_k||_1, \tag{7}$$

where $\mathcal{L}$ is the standard cross-entropy loss performed on all minibatch $\mathcal{B} \in \mathcal{D}_k$. We add the L_1 sparsity inducing regularization term to make the mask parameters highly sparse yet to efficiently capture the non-linearity, whose intensity can be controlled by the hyper-parameter λ_1.

Personalized Knowledge Reflection We avoid direct averaging the knowledge amongst different clients. We instead individually reflect only difference between target local model and the other clients to preserve local reliability more effectively. First, given local model m_k, we select the other beneficial knowledge that are relevant to m_k via $\Omega(\cdot)$ described in Eq. 3, returning a set of J pairs of $(\sigma_j, \zeta_j, \xi_j, \phi_j)$. Second, we separately update each factorized parameters ζ_j, ξ_j, and ϕ_j while minimizing the collapse of information as follows:

$$\zeta_k \leftarrow \zeta_k + \frac{exp(\alpha\cdot\sigma_j)}{\sum_{1:J} exp(\alpha\cdot\sigma_j)} \cdot \sum_{j=1}^{J}(\zeta_j - \zeta_k), \tag{8}$$

where α is scalar value for scaling the exponential reflection ratio of similarity score σ_j. This personalized knowledge reflection is also equally applied to ξ_j and ϕ_j, respectively (see Algorithm 1).

Algorithm 1 SimFed Algorithm

Function RunServer()
  initialize $\mathcal{M}, \mathbf{x}_\mathcal{N}, \tau$
  for each round $r = 1, 2, \ldots, R$ **do**
    $\mathcal{M}^{(r)} \leftarrow$ select $K^{(r)}$ clients from $\mathcal{M}$
    for each client $m_k^{(r)} \in \mathcal{M}^{(r)}$ **in parallel do**
      if $r > 1$ **then**
        $(\sigma_{1:J}, \zeta_{1:J}, \xi_{1:J}, \phi_{1:J}) \leftarrow \Omega(k, \tau, \mathcal{Z}_\mathcal{N}, \zeta, \xi, \phi)$
        $\zeta_k^{(r)} \leftarrow \zeta_k^{(r)} + \frac{exp(\alpha\cdot\sigma_j)}{\sum_{1:J} exp(\alpha\cdot\sigma_j)} \cdot \sum_{j=1}^{J}(\zeta_j^{(r)} - \zeta_k^{(r)})$
        $\xi_k^{(r)} \leftarrow \xi_k^{(r)} + \frac{exp(\alpha\cdot\sigma_j)}{\sum_{1:J} exp(\alpha\cdot\sigma_j)} \cdot \sum_{j=1}^{J}(\xi_j^{(r)} - \xi_k^{(r)})$
        $\phi_k^{(r)} \leftarrow \phi_k^{(r)} + \frac{exp(\alpha\cdot\sigma_j)}{\sum_{1:J} exp(\alpha\cdot\sigma_j)} \cdot \sum_{j=1}^{J}(\phi_j^{(r)} - \phi_k^{(r)})$
      end if
      $\zeta_k^{(r+1)}, \xi_k^{(r+1)},$
        $\phi_k^{(r+1)}, \mathbf{z}_{\mathcal{N},k} \leftarrow$ RunClient$(\zeta_k^{(r)}, \xi_k^{(r)}, \phi_k^{(r)}, \mathbf{x}_\mathcal{N})$
    end for
  end for
Function RunClient$(\zeta_k, \xi_k, \phi_k, \mathbf{x}_\mathcal{N})$
  for each local epoch e from 1 to E **do**
    for minibatch $\mathcal{B} \in \mathcal{D}_k$ **do**
      $\theta_{\zeta\times\xi\oplus\phi} \leftarrow \theta_{\zeta\times\xi\oplus\phi} - \eta\nabla\mathcal{L}(\mathcal{B};\theta_{\zeta\times\xi\oplus\phi})$
    end for
  end for
  $\mathbf{z}_{\mathcal{N},k} \leftarrow E_k(\mathbf{x}_\mathcal{N};\theta_{\zeta\times\xi\oplus\phi})$
  return $\zeta_k, \xi_k, \phi_k, \mathbf{z}_{\mathcal{N},k}$

We describe step-by-step pseudo-coded algorithm for overall learning procedure of our similarity matching and personalized knowledge reflection with factorized parameters in Algorithm 1.

5 EXPERIMENT

5.1 EXPERIMENTAL SETUP

Baseline Models & Architecture We compare our method, *Similarity Matching and Kernel Factorization (SimFed)*, with the following eight baseline algorithms, such as (1) Stand-Alone: training on local data without sharing local weights, (2) FedAvg (McMahan et al., 2017) performing weighted averaging, (3) FedProx (Li et al., 2018) using proximal regularization, (4) Agnostic-FL (Mohri et al., 2019) learning generalized central model over domains, (5) Clustered-FL (Sattler et al., 2019) clustering relevant clients, (6) FedPer (Arivazhagan et al., 2019) using personalized layers, (7) Per-FedAvg (Fallah et al., 2020) adopting MAML framework, and (8) FedFOMO (Zhang et al., 2021) using the first-order model optimization. We use ResNet-9 architecture for all base algorithms and our method. We ensure that all training configurations, such as learning rate, batch-size, etc., are set equally for a fairness, unless otherwise we state clearly in our paper.

Datasets (1) *Single-domained Dataset*: we use three datasets for three respective experiments, which are MNIST, CIFAR-10, and CIFAR-100, for the single-domain scenario. For the first step, we split each dataset into train, validation, and test sets, such that $55,990$/$7,000$/$7,000$ for MNIST, $47,985$/$6,000$/$6,000$ for CIFAR-10 and 100. We then split the train set into K local partitions $\mathcal{P}_{1:20}$

Table 1: **Performance Comparison of Single-domain Scenario** We train 20 clients on each dataset for 100 rounds respectively. We measure local accuracy at the final round and report averaged scores over three trials with different seeds. Note that the classifier layers are not aggregated in the permuted settings for fairness.

Single-domain Federated Learning	**MNIST**		**CIFAR-10**		**CIFAR-100**	
Method	Conventnl. IID	Permuted IID	Conventnl. IID	Permuted IID	Conventnl. IID	Permuted IID
Stand-Alone	96.91 (± 0.13)	97.26 (± 0.06)	59.62 (± 0.41)	62.01 (± 0.27)	25.12 (± 0.24)	22.92 (± 0.12)
FedAvg (McMahan et al., 2017)	97.87 (± 0.12)	95.26 (± 0.23)	65.52 (± 0.55)	37.73 (± 0.34)	45.36 (± 0.17)	11.48 (± 0.11)
FedProx (Li et al., 2018)	97.91 (± 0.04)	95.41 (± 0.19)	65.06 (± 0.37)	38.07 (± 0.26)	44.94 (± 0.22)	11.58 (± 0.17)
Agnostic-FL (Mohri et al., 2019)	97.87 (± 0.10)	95.13 (± 0.13)	64.69 (± 0.32)	39.37 (± 0.18)	44.63 (± 0.21)	11.73 (± 0.12)
Clustered-FL (Sattler et al., 2019)	97.57 (± 0.08)	96.41 (± 0.39)	65.39 (± 0.53)	48.31 (± 0.23)	42.83 (± 0.39)	17.94 (± 0.19)
FedPer (Arivazhagan et al., 2019)	98.11 (± 0.11)	94.81 (± 0.27)	68.70 (± 0.29)	38.12 (± 0.19)	49.12 (± 0.33)	11.26 (± 0.16)
Per-FedAvg (Fallah et al., 2020)	97.91 (± 0.14)	94.02 (± 0.11)	66.12 (± 0.37)	40.23 (± 0.31)	50.58 (± 0.13)	12.08 (± 0.22)
FedFOMO (Zhang et al., 2021)	97.53 (± 0.05)	94.97 (± 0.38)	67.08 (± 0.28)	44.12 (± 0.33)	48.13 (± 0.49)	11.08 (± 0.31)
SimFed (Ours)	**98.28 (± 0.12)**	**97.83 (± 0.36)**	**71.01 (± 0.19)**	**64.21 (± 0.29)**	**52.01 (± 0.18)**	**24.11 (± 0.23)**
Method	Non-IID (C)	Non-IID (P)	Non-IID (C)	Non-IID (P)	Non-IID (C)	Non-IID (P)
Stand-Alone	81.52 (± 0.43)	81.73 (± 0.45)	39.77 (± 0.11)	44.46 (± 0.22)	18.53 (± xx.xx)	17.36 (± 0.35)
FedAvg (McMahan et al., 2017)	91.89 (± 0.47)	72.34 (± 0.19)	44.37 (± 0.31)	26.74 (± 0.83)	26.32 (± 0.34)	8.87 (± 0.36)
FedProx (Li et al., 2018)	89.84 (± 0.32)	66.32 (± 0.22)	42.63 (± 0.25)	26.85 (± 0.34)	26.46 (± 0.33)	5.81 (± 0.83)
Agnostic-FL (Mohri et al., 2019)	91.51 (± 0.48)	70.97 (± 0.31)	43.33 (± 0.22)	27.11 (± 0.21)	25.14 (± 0.31)	8.21 (± 0.73)
Clustered-FL (Sattler et al., 2019)	90.59 (± 0.29)	77.78 (± 0.37)	49.68 (± 0.xx)	34.80 (± 0.41)	25.73 (± 0.xx)	11.61 (± 0.19)
FedPer (Arivazhagan et al., 2019)	90.32 (± 0.41)	70.45 (± 0.18)	50.43 (± 0.xx)	24.65 (± 0.38)	33.58 (± 0.xx)	5.68 (± 0.17)
Per-FedAvg (Fallah et al., 2020)	90.66 (± 0.35)	73.37 (± 0.43)	45.91 (± 0.73)	26.60 (± 0.52)	27.39 (± 0.86)	6.49 (± 0.33)
FedFOMO (Zhang et al., 2021)	91.74 (± 0.33)	68.94 (± 0.29)	50.62 (± 0.93)	25.50 (± 0.35)	28.29 (± 0.23)	8.93 (± 0.19)
SimFed (Ours)	**92.13 (± 0.27)**	**84.94 (± xx.xx)**	**52.13 (± xx.xx)**	**45.37 (± 0.23)**	**34.11 (± xx.xx)**	**18.38 (± 0.11)**

(K=20) in an identical manner (all instances in each class are evenly distributed to all clients) or a non-identical way (instances in each class are sampled from Dirichlet distribution with α=0.5). We further permute the labels for each class per local partition $\mathcal{P}_k$ for *Permuted IID* and *Permuted Non-IID* setting to simulate the *Label-Heterogeneity*. (2) *Multi-domained Dataset*: we use CIFAR-100 datasets $(60,000)$ and split it into 20 datasets $\mathcal{D}_{1:20}$ grouped by 20 superclasses which are also introduced as "coarse" labels, such as mammals, household furniture, vehicles, etc. In each domain $\mathcal{D}_k$ $(3,000)$, non-overlapping 5 subclasses are assigned to each client. Then we split each dataset into train $(2,400)$, validation(150), and test (450) sets. Please see further details for the dataset configuration in Section 3.2.

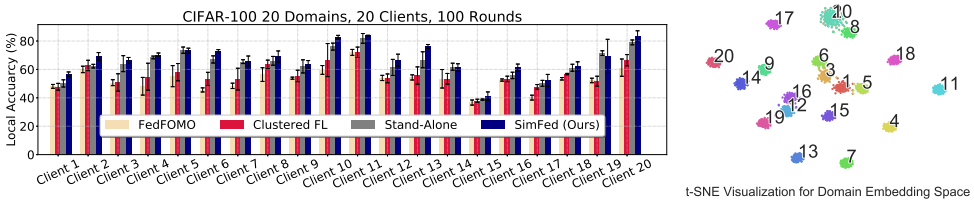
5.2 Experimental Result

Label Heterogeneity As shown in Table 1, for the conventional IID and Non-IID settings, all federated learning methods show higher performance than the pure local training baseline (Stand-Alone), meaning that the locally learned knowledge is helpful each other and sharing them certainly improves each local performance. Interestingly, however, when the labels are not synchronized for each clients (the permuted IID and Non-IID settings), we observed that all existing federated learning methods are degenerated and show even lower than the Stand-Alone models for both IID and Non-IID partitions. Again, note that we do not share the classifier layers for fairness for all algorithms in this permuted settings. The permutation of labels makes your model heterogeneously learn even if the given data is from the same class. Our method, however, shows strongest performance particularly for the permuted settings and actually improves over Stand-Alone models which all other existing models failed. This clearly shows that our method can effectively reflect the other knowledge while minimizing the information collapse via kernel factorization as well as similarity matching.

Domain Heterogeneity Table 2 shows the experimental results for the multi-domain scenario with non-overlapping (disjoint) 20 domains from CIFAR-100 dataset. In this extremely heterogeneous setting, similarly to the permuted labels scenarios, we observe that the prior federated learning algorithms cannot properly share local knowledge amongst clients and fail to achieve the better performance over pure local training baseline (Stand-Alone), showing that the domain heterogeneity is critical challenges of federated learning. Even though it seems that federated learning does not work in this domain heterogeneous settings, our method, however, shows the strong possibility. SimFed (Similarity Matching) achieves much higher scores over the existing federated learning algorithms, meaning that it properly utilizes other clients' knowledge. However, it still shows slightly lower performance compared to the Stand-Alone model. Even though we choose the beneficial knowledge,

Table 2: **Performance Comparison of Multi-domain Scenario** We train 20 clients on 20 heterogeneous datasets simultaneously for 100 rounds. We measure local accuracy at the final round and report averaged scores over three trials. Note that in the multi-domained settings the classifier layer is not merged for fairness.

Multi-domain Federated Learning	**CIFAR-100, 20 Domains, 5 independent sub-classes for each domain**						
Method	Aqu. Mammals	Fish	Flowers	Food	Fruits & Vege.	Elec. Devices	Furniture
Stand-Alone	50.15 (± 2.54)	62.30 (± 1.29)	64.07 (± 5.63)	68.44 (± 0.96)	**73.56 (± 2.20)**	67.04 (± 2.52)	65.48 (± 1.36)
FedAvg (McMahan et al., 2017)	48.07 (± 1.76)	60.37 (± 1.85)	53.63 (± 2.52)	45.48 (± 1.56)	58.00 (± 1.55)	48.15 (± 2.16)	51.63 (± 3.96)
FedProx (Li et al., 2018)	45.63 (± 1.00)	59.48 (± 1.36)	42.96 (± 4.79)	46.07 (± 5.31)	58.67 (± 2.45)	51.78 (± 2.02)	51.93 (± 1.79)
Agnostic-FL (Mohri et al., 2019)	48.52 (± 1.27)	59.78 (± 1.79)	52.59 (± 0.91)	46.81 (± 3.37)	54.44 (± 3.54)	49.48 (± 1.74)	51.48 (± 3.10)
Clustered-FL (Sattler et al., 2019)	47.78 (± 2.40)	62.81 (± 3.63)	50.81 (± 6.14)	54.89 (± 9.47)	58.07 (± 6.08)	53.33 (± 4.63)	53.19 (± 7.64)
FedPer (Arivazhagan et al., 2019)	48.11 (± 1.22)	60.67 (± 0.44)	57.67 (± 0.78)	41.11 (± 0.67)	60.22 (± 2.44)	49.22 (± 0.56)	48.22 (± 5.56)
Per-FedAvg (Fallah et al., 2020)	46.78 (± 2.11)	60.00 (± 1.11)	48.33 (± 1.67)	49.78 (± 0.00)	53.00 (± 5.44)	47.44 (± 2.33)	50.89 (± 6.67)
FedFOMO (Zhang et al., 2021)	48.07 (± 1.36)	60.00 (± 2.27)	50.67 (± 2.09)	48.15 (± 6.23)	55.26 (± 7.42)	45.56 (± 1.45)	48.44 (± 1.89)
Similarity Matching (Ours)	51.56 (± 1.45)	64.67 (± 1.89)	64.52 (± 1.00)	70.67 (± 0.63)	67.41 (± 5.87)	67.78 (± 0.65)	**66.96 (± 0.69)**
+ Factorization (Ours)	**56.59 (± 1.63)**	**69.11 (± 2.75)**	**66.30 (± 2.13)**	**70.37 (± 1.21)**	73.26 (± 1.71)	**72.59 (± 1.37)**	65.93 (± 3.52)
	Insects	Carnivores	Objects	Outdr. Scenes	Om./Herbivores	Med. Mammals	Invertebrates
Stand-Alone	66.07 (± 3.09)	62.15 (± 4.63)	76.07 (± 2.46)	81.85 (± 3.43)	61.41 (± 5.66)	66.81 (± 5.62)	**61.93 (± 2.46)**
FedAvg (McMahan et al., 2017)	55.48 (± 2.72)	51.41 (± 1.39)	59.19 (± 2.47)	70.59 (± 5.26)	52.44 (± 3.62)	52.81 (± 1.37)	56.22 (± 1.75)
FedProx (Li et al., 2018)	55.26 (± 2.15)	49.04 (± 2.72)	61.04 (± 3.54)	74.89 (± 3.15)	51.41 (± 3.29)	55.04 (± 0.86)	52.59 (± 3.82)
Agnostic-FL (Mohri et al., 2019)	54.00 (± 0.31)	49.70 (± 4.73)	57.78 (± 6.95)	73.56 (± 1.84)	50.74 (± 3.63)	54.37 (± 3.99)	49.85 (± 2.31)
Clustered-FL (Sattler et al., 2019)	63.56 (± 3.00)	55.41 (± 4.03)	66.59 (± 11.63)	72.07 (± 3.67)	53.70 (± 3.12)	55.70 (± 6.09)	53.41 (± 3.75)
FedPer (Arivazhagan et al., 2019)	56.33 (± 3.67)	51.89 (± 4.56)	62.44 (± 1.56)	72.67 (± 2.22)	51.78 (± 4.00)	54.33 (± 3.22)	51.44 (± 4.33)
Per-FedAvg (Fallah et al., 2020)	57.89 (± 2.56)	55.22 (± 1.00)	53.78 (± 11.78)	73.44 (± 0.33)	53.56 (± 0.89)	52.44 (± 1.11)	48.89 (± 0.44)
FedFOMO (Zhang et al., 2021)	56.37 (± 4.81)	53.85 (± 0.73)	59.63 (± 3.09)	72.00 (± 2.09)	54.00 (± 1.84)	54.37 (± 1.82)	53.41 (± 6.52)
Similarity Matching (Ours)	64.81 (± 3.98)	58.67 (± 4.91)	76.44 (± 3.21)	**84.44 (± 0.36)**	58.30 (± 9.29)	66.52 (± 0.76)	55.04 (± 6.22)
+ Factorization (Ours)	**69.11 (± 3.90)**	**63.56 (± 2.67)**	**82.81 (± 1.18)**	83.85 (± 0.21)	**66.52 (± 4.21)**	**76.15 (± 1.41)**	61.41 (± 2.58)
	Peopole	Reptiles	Small Mammals	Trees	Vehicles 1	Vehicles 2	**AVERAGE**
Stand-Alone	38.74 (± 0.46)	55.78 (± 2.27)	50.22 (± 2.09)	61.19 (± 2.46)	**71.48 (± 1.65)**	79.04 (± 1.71)	64.19 (± 0.79)
FedAvg (McMahan et al., 2017)	36.07 (± 4.46)	50.00 (± 3.19)	46.37 (± 2.42)	54.96 (± 1.48)	52.22 (± 2.18)	63.85 (± 2.16)	53.35 (± 0.76)
FedProx (Li et al., 2018)	34.52 (± 1.41)	49.48 (± 1.84)	44.15 (± 1.71)	52.52 (± 2.88)	48.07 (± 1.99)	63.33 (± 0.73)	52.39 (± 0.47)
Agnostic-FL (Mohri et al., 2019)	34.89 (± 0.36)	48.30 (± 2.04)	45.85 (± 3.27)	55.41 (± 1.17)	49.93 (± 1.21)	65.41 (± 4.08)	52.64 (± 0.91)
Clustered-FL (Sattler et al., 2019)	37.93 (± 1.09)	53.33 (± 1.92)	47.70 (± 1.65)	56.74 (± 0.46)	51.70 (± 3.52)	66.52 (± 3.94)	55.76 (± 2.37)
FedPer (Arivazhagan et al., 2019)	34.33 (± 3.67)	46.44 (± 0.44)	46.33 (± 0.78)	53.78 (± 2.00)	50.33 (± 4.11)	64.44 (± 1.78)	53.09 (± 1.56)
Per-FedAvg (Fallah et al., 2020)	38.67 (± 0.22)	52.22 (± 0.22)	46.67 (± 0.44)	52.89 (± 4.89)	53.00 (± 1.67)	63.56 (± 5.11)	52.92 (± 1.50)
FedFOMO (Zhang et al., 2021)	36.67 (± 1.79)	52.52 (± 0.76)	40.15 (± 1.71)	53.41 (± 0.86)	52.22 (± 1.58)	61.26 (± 6.08)	53.41 (± 6.52)
Similarity Matching (Ours)	36.07 (± 1.76)	58.44 (± 1.44)	50.67 (± 1.55)	59.11 (± 1.84)	68.00 (± 5.70)	76.07 (± 2.50)	63.31 (± 0.55)
+ Factorization (Ours)	**41.26 (± 2.89)**	**61.33 (± 2.44)**	**52.30 (± 4.18)**	**62.00 (± 3.36)**	69.19 (± 12.02)	**83.41 (± 1.41)**	**67.35 (± 0.66)**



(a) Bar plot for the Per-Client Performance (b) Latent Space Visualization

Figure 2: **Per-client Performance and Latent Space Visualization for Multi-domain Scenario:** (a) we show bar plot for the per-client performance comparison. (b) we visualize labent embedding space for domain embeddings. The labeled number is corresponding to the dataset and also client number.

the reflection of the knowledge should be properly handled, which is an another critical challenges. With kernel factorization, SimFed finally improves ($4\%p$ higher in average) over Stand-Alone model, meaning that ours can discriminate which knowledge from others are beneficial or harmful and successfully reflects the knowledge to adapt to the given local doamain while effectively alleviating the risks caused by domain heterogeneity. In Figure 2 Left, we visualize per-client performance. All clients of our method shows way higher performance over all federated learning methods.

Table 3: Similarity Matching

Method	Acc.(%)
Worst Match	$xx.xx\%$
Random Match	$xx.xx\%$
SimFed (Ours)	67.35%

Effectiveness of Similarity Matching Our robustness on label and domain heterogeneity firstly comes from correctly discriminating which knowledge is helpful or not. To see the effectiveness, we compare performance between our similarity matching algorithms and Random Matching & Worst Matching baselines on Multi-domain Scenario with the exactly same configurations as described above. Random matching baseline randomly pick two arbitrary models to aggregate while worst matching model choose the most two dissimilar models to reflect for each round. As shown in Table 3, both random and worst

matching models significantly suffer from the performance degeneration compared to ours, meaning that we properly choose the beneficial knowledge for each round. In Figure 2 Right, we visualize our latent space for domain embeddings. We utilize 100 Gaussian criteria inputs to transform them for each local clients after 100 rounds training. As shown, our method successfully embed 20 domains while minimizing overlaps between the domains. This succesful encoding helps our similarity matching to be more robust.

Effectiveness of Kernel Factorization Similarity matching might not be sufficient as we observed in multi-domain scenario. Successfully reflecting the other heterogeneous knowledge is the critical points in the domain heterogeneity setting. As we explored already in Table 1 and 2, the kernel factorization improves performance over when only similarity matching is applied. In Table 4, we compare one more baseline which we eliminate the spare mask ϕ on the multi-domain settings with the same configurations described above. ϕ is designed to capture non-linearity which might be hard to capture with only two rank 1 basis vector parameters ζ and ξ. As shown, we observe that the performance is dropped without the spare mask, meaning that the sparse mask efficiently and effectively captures the non-linearity.

Table 4: Kernel Factorization

Method	Acc.(%)
w/o ϕ	$xx.xx\%$
w/o Factor.	63.31%
SimFed (Ours)	67.35%

6 Conclusion

In this work, we discussed the realistic scenarios of federated learning at worldwide scale, where (1) the labeling schemes are possibly not synchronized across all participants and (2) local clients can be from any multiple domains, which we named such challenges as *Label Heterogeneity* and *Domain Heterogeneity*. To tackle these problems, we proposed a novel method, *Similarity Matching and Kernel Factorization (SimFed)*, which measures semantic similarity/dissimilarity between local knowledge and reflects the relevant clients that are beneficial to each other. Furthermore, SimFed factorizes the model parameters into two rank 1 basis vectors and the light-weighted sparse masks to effectively capture permutation-robust representations and reduce information loss when reflecting the heterogeneous knowledge. We extensively validate our method on single-domained and multi-domained datasets, and show that our method outperforms the current state-of-the-art federtaed learning baselines.

References

Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. Federated learning with personalization layers, 2019.

Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Adaptive personalized federated learning, 2020.

Enmao Diao, Jie Ding, and Vahid Tarokh. Heterofl: Computation and communication efficient federated learning for heterogeneous clients, 2021.

Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning: A meta-learning approach, 2020.

Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An efficient framework for clustered federated learning, 2021.

Filip Hanzely and Peter Richtárik. Federated learning of a mixture of global and local models, 2021.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank J. Reddi, Sebastian U. Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning, 2021.

Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning, 2021.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127*, 2018.

Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Federated multi-task learning for competing constraints. *CoRR*, abs/2012.04221, 2020a. URL https://arxiv.org/abs/2012.04221.

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

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, 2020.

Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning, 2020.

H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.

Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. Agnostic federated learning, 2019.

Xingchao Peng, Zijun Huang, Yizhe Zhu, and Kate Saenko. Federated adversarial domain adaptation. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HJezF3VYPB.

Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. Clustered federated learning: Model-agnostic distributed multi-task optimization under privacy constraints, 2019.

Hyowoon Seo, Jihong Park, Seungeun Oh, Mehdi Bennis, and Seong-Lyun Kim. Federated knowledge distillation, 2020.

Aviv Shamsian, Aviv Navon, Ethan Fetaya, and Gal Chechik. Personalized federated learning using hypernetworks. *arXiv preprint arXiv:2103.04628*, 2021.

Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet Talwalkar. Federated multi-task learning, 2018.

Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BkluqlSFDS.

Chun-Han Yao, Boqing Gong, Yin Cui, Hang Qi, Yukun Zhu, and Ming-Hsuan Yang. Federated multi-target domain adaptation, 2021.

Michael Zhang, Karan Sapra, Sanja Fidler, Serena Yeung, and Jose M. Alvarez. Personalized federated learning with first order model optimization. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=ehJqJQk9cw.

Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-free knowledge distillation for heterogeneous federated learning, 2021.