Subspace Learning for Personalized Federated Optimization

Seok-Ju Hahn, Minwoo Jeong, Junghye Lee

1 Department of Industrial Engineering, Ulsan National Institute of Science and Technology  
2 AI Lab, Kakao Enterprise  
3 Artificial Intelligence Graduate School, Ulsan National Institute of Science and Technology

seokjuhahn@unist.ac.kr, lloyd.ai@kakaoenterprise.com, junghyelee@unist.ac.kr

Abstract

As data is generated and stored almost everywhere, learning a model from a data-decentralized setting is a task of interest for many AI-driven service providers. Although federated learning is settled down as the main solution in such situations, there still exists room for improvement in terms of personalization. Training federated learning systems usually focuses on optimizing a global model that is identically deployed to all client devices. However, a single global model is not sufficient for each client to be personalized on their performance as local data assumes to be not identically distributed across clients. We propose a method to address this situation through the lens of ensemble learning based on the construction of a low-loss subspace continuum that generates a high-accuracy ensemble of two endpoints (i.e., global model and local model). We demonstrate that our method achieves consistent gains both in personalized and unseen client evaluation settings through extensive experiments on several standard benchmark datasets.

Introduction

For the blessing of advanced communication and computation technologies, individuals and institutions are nowadays data producers as well as data keepers. Thereby, training a machine learning model under a data-centralized setting is sometimes not viable. If there exists too many clients generating their own data in real-time on their devices, it is difficult to collect them. Moreover, such data may contain private information that is not possible to share or transfer to other places. Federated learning (FL) (Konečný et al. 2016; McMahan et al. 2017) is a solution to solve this problematic situation as it enables massive parallel training of a machine learning model collaboratively across clients or institutions without sharing their private data, under the orchestration of the central server (e.g., service providers). In the most common FL setting, such as FedAvg (McMahan et al. 2017), each participant trains a local model with its own data and transmits resulting model parameters to the central server. Then, the central server aggregates (i.e., weighted averaging) local parameters to update a global model and then broadcast this new global model to participating clients. This process is repeated until convergence.

However, a single global model is not enough to provide satisfiable experiences to all clients, due to the curse of data heterogeneity. Since there exists an inherent difference among clients (e.g., culture, country, age, etc.), the data resides in each client is not statistically identical, in other words, the data is not independent and identically distributed (non-IID). In this situation, applying a simple FL method like FedAvg cannot avoid a pitfall of the high generalization error, or even divergence of the global model (Karimireddy et al. 2019). When this occurs, clients barely have the motivation on participating in federated optimization to train a single global model, as it is far better to train and exploit a local model within themselves using its own data.

As training a single global model via FL cannot be a panacea for all clients, it is natural to rethink the scheme of FL. As a result, a personalized FL (PFL) becomes an essential component of FL, which has yielded many related studies. Many approaches are proposed for PFL, such as multitask learning (Smith et al. 2017; Marfoq et al. 2021; Dinh et al. 2021), modified optimization scheme based on a regularization (Hanzely and Richtárik 2020; Dinh, Tran, and Nguyen 2020; Huang et al. 2021), local fine-tuning (Pallah, Mokhtari, and Ozdaglar 2020; Jiang et al. 2019; Kodak, Balcan, and Talwalkar 2019), clustering (Ghosh et al. 2020; Sattler, Müller, and Samek 2020; Mansour et al. 2020; Duan et al. 2020), and a model mixture (Arivazhagan et al. 2019; Liang et al. 2020; Mansour et al. 2020; Deng, Kamani, and Mahdavi 2020). Among them, the model mixture-based method is attracting attention for its flexibility and extensibility of the model structure along with promising performance. It separates the model into the local part and global part, expecting that the former captures heterogeneous local data distribution while the latter focuses on the information common across clients. Thereby, each participating client ends up having an explicit personalized model (i.e., local model) as a result of FL.

Meanwhile, this model mixture approach can be basically viewed as a type of ensemble learning, as the core idea is to interpolate two different models. However, a slight difference exists: a local model can only access the local data belong to a specific client, while a global model can implicitly gain diverse information from different client’s local data.
through a model aggregation at the central server. While this is typically not true in ensemble learning, since models participating in the ensemble can usually access identical data for training. As many ensemble strategies have been proposed in the context of deep learning (Lakshminarayanan, Pritzel, and Blundell 2016; Maddox et al. 2019; Izmairlov et al. 2018; Blundell et al. 2015; Huang et al. 2017), analyses on the behavior of ensemble learning are an active research area (Garipov et al. 2018; Port, Hu, and Lakshminarayanan 2019; Port and Jastrzebski 2019; Draxler et al. 2018; Frankle et al. 2020; Wortsman et al. 2021). Merely interpolating two different models (i.e., local and global models) for FL as in previous works might be a valid approach, but explicitly promoting the mixture to be well-performed under the scheme of ensemble learning can be a more attractive choice in the context of PFL.

In this study, we propose SuperFed, a novel PFL method that leverages the power of ensemble learning, which is accomplished through subspace learning. Since subspace learning provides the optimal space of the objective function instead of a single optimum point, many possible model combinations can be generated, which would be useful for handling heterogeneous local data from multiple clients. Specifically, our method constructs a model mixture consisting of two models and involves two sequential optimization phases in order for each model of the mixture to be specialized in federation and personalization, respectively, and at the same time to lie in the harmonious subspace customized to the heterogeneous data distribution of each client. Thereby, without loss of overall performance of a global model as well as each local model per client, both models can be interpolated with each other more elaborately than existing model mixture-based PFL methods, which merely mixed or connected them. Under the general framework, two different model mixing schemes are presented to find better subspaces. The proposed method as model interpolation-based ensemble learning can be reduced to existing benchmark FL methods by adjusting model hyperparameters, and thus it can guarantee to have at least their performances. With extensive experiments on standard benchmark datasets in a realistic federated setting with non-IID local data, we empirically demonstrate the efficacy and robustness of our method. The method as an improved model mixture-based PFL method shows a decent performance gain compared with other comparable PFL methods by achieving better personalization in FL.

Related Works

FL with Non-IID Data After (McMahan et al. 2017) proposed the vanilla FL algorithm (FedAvg), handling non-IID data across clients is one of the major points to be resolved in the FL field. (Zhao et al. 2018) showed that sharing a subset of each client’s local data to the central server is effective to prevent divergence of a global model. However, this is an unrealistic assumption as the original purpose of FL is to train a model in a data-decentralized setting. (Hsu, Qi, and Brown 2019) suggested that accumulating model updates with momentum at the central server can mitigate the harmful oscillations caused by non-IID data. Modifying a model aggre-
network models have been proposed (Blundell et al. 2015; Huang et al. 2017; Izmailov et al. 2018; Maddox et al. 2019; Lakshminarayanan, Pritzel, and Blundell 2016). Follow-up studies are proceeded for analyzing traits of ensemble models. In (Fort, Hu, and Lakshminarayanan 2019; Fort and Jastrzebski 2019; Frankle et al. 2020; Garipov et al. 2018; Draxler et al. 2018), it is observed that there exists a low-loss subspace connecting two or more different networks separately trained on the same dataset, which is named as a connector in (Wortsman et al. 2021). Models consisting of a connector have a low cosine similarity with each other in the weight space when the two models are trained from different initializations. Though there exist some studies on constructing a low-loss subspace (connector) in (Izmailov et al. 2018; Maddox et al. 2019; Izmailov et al. 2020), a recent study conducted by (Wortsman et al. 2021) proposed a simple and efficient approach to finding a connector by learning a parameterized subspace. Our method adopts this technique to construct a subspace between each local model and a global model as an explicit ensemble learning method for PFL.

**Proposed Method**

**Overview** We describe our proposed PFL method, SuPerFed, in the perspective of model ensemble via subspace learning. We adopt a common federated optimization scheme which was proposed in (McMahan et al. 2017), where the central server orchestrates federated optimization across participating clients with iterative communication of model parameters. Since our method is essentially a model mixture-based PFL method, we need two models per client (one for a global model, the other for a local model), of which initialization is different from each other, but the structure is the same. SuPerFed is composed of two phases: federation phase (PHASE I) and personalization phase (PHASE II).

**Notations** We first define arguments required for the federated optimization as follows: a total number of rounds ($R$), a local batch size ($B$), a number of local epochs ($E$), a total number of clients ($K$), a fraction of clients selected at each round ($C$), an indicator of starting round for personalization ($L$), and a local learning rate ($η$). Next, there are three hyper-parameters required for the optimization of our proposed method: a mixing ratio $α$ sampled from the uniform distribution $U(0, 1)$, a constant $β$ for the construction of a low-loss subspace between local and global models by inducing orthogonality, and a constant $γ$ for a proximity term to avoid divergence of global models that are locally updated from the aggregated global model in the previous round.

**Problem Statement** Consider each client $c_i$ ($i \in [K]$) has its own dataset $D_i = \{(x_i^j, y_i^j)\}_{j=1}^{n_i}$ composed of training and validation sets $D_i^{tr}$ and $D_i^{v}$, and a model $\theta_i$ resides in a client. In PFL, we assume that datasets are non-IID across clients, i.e., each dataset $D_i$ is sampled independently from a corresponding distribution $P_i$ on $X \times Y$. Denote $\mathcal{H}$ as a class of hypotheses, and a hypothesis $h \in \mathcal{H}$ is learned by the loss function $l : \mathcal{H} × \mathcal{Z} \rightarrow \mathbb{R}^+$ in the form of $l(h(x), y; \theta)$. We denote the expected loss by $\mathcal{L}_P(h, \theta) = \mathbb{E}_{(x, y) \sim P_l}[l(h(x), y; \theta)]$, and the empirical loss by $\hat{\mathcal{L}}_P(h, \theta) = \frac{1}{n_i} \sum_{j=1}^{n_i} l_i(h(x_i^j), y_i^j; \theta)$. Then, the main objective of PFL is to optimize

$$\min_{h \in \mathcal{H}} \frac{1}{K} \sum_{i=1}^{K} \mathcal{L}_P(h, \theta_i),$$

(1)

and we can minimize this objective through the empirical loss minimization:

$$\min_{\theta_1, \ldots, \theta_K} \frac{1}{K} \sum_{i=1}^{K} \hat{\mathcal{L}}_P(h, \theta_i) = \frac{1}{K} \sum_{i=1}^{K} \sum_{j=1}^{n_i} l_i(h(x_i^j), y_i^j; \theta_i).$$

(2)

Under this setting, each client $c_i$ has a set of paired model parameters $\theta_i = \{w_i^{(r)}, w_i^{l(r)}\}$ at round $r$ ($r \in [R]$), each of which is initialized with a different random seed. Each parameter is for training a global model and a personalized model within each client, using a local dataset $D_i$. We suppose both models have the same structure, i.e., number of layers, type of activation function, etc. On the client side, two models are mixed in the form of convex combination using $\alpha \sim U(0, 1)$: $w_i^{α(r)} = (1 - \alpha)w_i^{tr(r)} + \alpha w_i^{l(r)}$. The mixing process is done at every training batch update in the local epoch, either in the form of model-wise (Model Mixing) or layer-wise (Layer Mixing) manner (See PHASE II: Personalization for details). Note that the mixing is achieved in the weight space, not the output space. Then, it is optimized using a local training dataset $D_i^{tr}$ with a modified local loss function $\hat{l}_i$, which is composed of canonical loss function $l_i$ with additional regularization terms (suppose we are client $i$ at round $r$):

$$\bar{l}_i(h(x_i), y_i; w_i^{α(r)}) := l_i(h(x_i), y_i; w_i^{α(r)}) + \beta \cos^2(w_i^{g(r)}, w_i^{l(r)}) + γ \left\| w_i^{g(r)} - w_i^{l(r)} \right\|^2,$$

(3)

where $w_i^g$ is a copy of a global model constructed from the previous round, and $\cos(\cdot, \cdot)$ stands for cosine similarity defined as: $\cos(w_i^{g(r)}, w_i^{l(r)}) = \langle w_i^{g(r)}, w_i^{l(r)} \rangle / \left\| w_i^{g(r)} \right\| \left\| w_i^{l(r)} \right\|$. The first regularization term adjusted by $\beta$ induces orthogonality between two weight vectors, which thereby yields a low-loss manifold between two weight spaces. The second regularization term controls the proximity of locally updated global model $w_i^{g(r)}$ from the aggregated global model $w_i^*$ at the previous round. By doing so, we expect each local update of the global model not to be derailed from each other, which prevents a divergence of aggregated global model at the center server.

On the server-side, only a global model $w_i^{g(r)}$ of each client is transferred every round, and the central server aggregates them to make an updated global model: $w_i^{g(r+1)} = \frac{1}{\sum_{i=1}^{K} n_i} \sum_{i=1}^{K} n_i w_i^{g(r)}$. This updated global model is again broadcast to selected clients as a local version of a global model: $w_i^{g(r+1) \leftarrow w_i^{g+1}}$. By adjusting a set of hyper-parameters, $α, β$ and $γ$, we define and describe PHASE I.
Figure 1: Overview of the learning dynamics of SuPerFed. It consists of two phases: PHASE I (federation phase) and PHASE II (personalization phase).

and PHASE II respectively in following subsections for detailed explanations of SuPerFed. See Figure 1 for simplified view of the proposed method.

**Phase I: Federation** The federation phase (PHASE I) aims to train a global model to have general knowledge that is common to participating clients. It is a natural choice since training a personalized model at the early stage of FL may fail to achieve convergence of a global model. When setting $\alpha = \beta = 0$ and $\gamma > 0$, the objective (3) becomes equivalent to optimize the objective proposed in (Li et al. 2018; Dinh, Tran, and Nguyen 2020), which introduced a proximity regularization term for preventing a local update of a global model from diverging toward harmful directions. Until the federated optimization round reaches $L$, which is an indicator of starting the personalization phase, clients contribute to update only a downloaded global model to boost up a performance of a global model $w^g$ aggregated at the central server.

**Phase II: Personalization** After PHASE I is completed at $L$, the personalization phase (PHASE II) begins. The main goal of this phase is to build a personalized model for each client on top of the federated global model trained during PHASE I. From this phase, a mixing process (Model Mixing or Layer Mixing) using $\alpha$ is applied between $w^{g(r)}_i$ and $w^{l(r)}_i$ at every batch update during training. In detail, Model Mixing treats each model $w^{g(r)}_i$, $w^{l(r)}_i$ as a weight vector and combines them ignoring compositions and the number of layers, while Layer Mixing combines the two in a layer-wise manner, applying different values $\alpha \sim U(0, 1)$ per layer. Since merely introducing the mixing process does not induce the benefit of ensemble learning, it is required to set $\beta > 0$. In (Fort, Hu, and Lakshminarayanan 2019; Wortsman et al. 2021), it is observed that independently trained models show dissimilarity between parameters in terms of cosine similarity; and thereby enforcing two differently initialized models to maintain dissimilar parameters with each other results in finding a low-loss manifold continuum between two models. Here, “between two models” is equal to every possible realization of a mixed model $w^{\alpha(r)}_i = (1 - \alpha)w^{g(r)}_i + \alpha w^{l(r)}_i$ by varying $\alpha$. In (Larrazabal et al. 2021; Ayinde, Inanc, and Zurada 2019; Mashhadi, Nowaczyk, and Pashami 2021), inducing orthogonality (i.e., forcing cosine similarity to zero) between parameters prevents a model learning redundant features given learning capacity. In this context, we expect the local model of each client $w^{l(r)}_i$ to learn client-specific knowledge that cannot be captured by training of the global model $w^{g(r)}_i$ in PHASE I. The proximity regularization on the global model $w^{g(r)}_i$ with $\gamma > 0$ is still valid in PHASE II. See Algorithm 1 for understanding a whole optimization process of SuPerFed.

**Experiments**

**Datasets and Settings for PFL** We used four datasets: CIFAR10, CIFAR100 (Krizhevsky 2009), EMNIST (Cohen et al. 2017) and TinyImageNet (Le and Yang 2015). All of each dataset is for a multi-class image classification task, containing 60,000 images with 10 classes, 60,000 images with 100 classes, 814,255 images with 62 classes, and 110,000 images with 200 classes, respectively.

As model mixture-based PFL provides both a single global model and multiple personalized models, we at first divide the test set from each dataset (CIFAR10, CIFAR100, EMNIST, TinyImageNet in order) for the evaluation of a global model (named as global test set), each of which contains 10,000, 10,000, 82,587 and 10,000 samples from all
having their own dataset

Central server orchestrates orthogonality, $\gamma$ rate

20 with momentum 0.9 throughout all experiments. A detailed structure of the split is described in Appendix. For all experiments, we used balanced sample sizes. A detailed description of the dataset datasets by splitting them semantically by labels, with unbalanced sample sizes. As a consequence, each client has 500 samples when the dataset is CIFAR10 or CIFAR100, 452 samples on average (with the standard deviation of 203) when the dataset is EMNIST, and 252 samples on average (with the standard deviation of 88) when the dataset is TinyImageNet. We further make each client have 80% of total samples for its local training set, and remaining 20% of total samples for its local test set. Thereby, the performance of the personalized model is reported by averaging the values of an evaluation metric (i.e., accuracy) on the local test set from all clients.

Main Results In this section, we empirically observe behaviors of SuPerFed through extensive experiments and evaluate its performance on personalization in several benchmark datasets. As SuPerFed inherits the scheme of model mixture-based PFL, we compared our method with other model mixture-based PFL methods (FedPer (Arivazhagan et al. 2019), APFL (Deng, Kamani, and Mahdavi 2020), LG-FedAvg (Liang et al. 2020)) and two basic FL methods sharing a similar optimization strategy with ours (i.e., model averaging and proximity regularization) (FedAvg (McMahan et al. 2017), FedProx (Li et al. 2018)). We also considered variants of FedAvg and FedProx for fair comparison on the personalization performance by adopting the strategy proposed in (Wang et al. 2019). They presented a simple personalization method after finishing the update of a single global model through FL: locally updating a global model with small steps using local data of the clients. We denote the variants as FedAvg-LU and FedProx-LU, where LU stands for the local update. We report the best accuracy of each method by tuning its hyperparameters. Our method (SuPerFed-MM and SuPerFed-LM) achieves the best mean local accuracy (i.e., personalization performance) on all datasets. Note that as LG-FedAvg and FedProx only exchange parts of a model, their global model accuracy cannot be evaluated on the global test set at the central server.

Results of the pathological non-IID setting (CIFAR10 and CIFAR100) are shown in Table 1 and those of the realistic scenario (EMNIST and TinyImageNet) are shown in Table 2. Our method outperforms existing model mixture-based PFL methods as well as basic FL methods and its variants in terms of personalization performance.

We implemented all methods using PyTorch (Paszke et al. 2019) version 1.8.1, CUDA version 11.1, and all of our experiments are conducted on ubuntu LTS 18.04 equipped with Intel(R) Xeon(R) Gold 5120 CPU, one NVIDIA Tesla A100 GPU, and 122GB memory.

When to Start Personalization We conducted an experiment on selecting L, which determines the initiation round of PHASE II, the personalization phase. As our proposed method can choose a mixing scheme from either Model Mixing or Layer Mixing, we designed separate experiments per scheme. We denote each scheme by SuPerFed-MM and SuPerFed-LM. By adjusting L by 0.1 increments from 0 to 0.9 of R, we evaluated the mean accuracy of personalized models and the accuracy of the global model. This experiment was done only using the CIFAR10 dataset with

for $i = 1, ... E$ do
    $B_i \leftarrow \text{Split } D_i \text{ into batches of size } B$
    for local batch $(x, y) \in B_i$ do
        $\alpha = \begin{cases} 0 & : r < L \text{(PHASE I)} \\ \text{Sample from } U(0, 1) & : r \geq L \text{(PHASE II)} \end{cases}$
        $w_i^\alpha(r) = (1 - \alpha)w_i^\alpha(r) + \alpha w_i^r$
        $w_i^\alpha(r+1) \leftarrow w_i^\alpha(r) - \eta \nabla (\hat{L}_i(h(x), y; w_i^\alpha(r)))$ [3]
    end for
    $w_i^r \leftarrow w_i^\alpha(r+1) \big|_{\alpha=0}$
    $w_i^l(r+1) \leftarrow w_i^\alpha(r+1) \big|_{\alpha\in[0,1]}$
end for

13: upload locally updated version of global model $w_i^\alpha(r+1)$ to the server

and 390 for CIFAR10, CIFAR100, EMNIST, and TinyImageNet, respectively. As a consequence, each client has 500 samples when the dataset is CIFAR10 or CIFAR100, 452 samples on average (with the standard deviation of 203) when the dataset is EMNIST, and 252 samples on average (with the standard deviation of 88) when the dataset is TinyImageNet. We further make each client have 80% of total samples for its local training set, and remaining 20% of total samples for its local test set. Thereby, the performance of the personalized model is reported by averaging the values of an evaluation metric (i.e., accuracy) on the local test set from all clients.

Main Results In this section, we empirically observe behaviors of SuPerFed through extensive experiments and evaluate its performance on personalization in several benchmark datasets. As SuPerFed inherits the scheme of model mixture-based PFL, we compared our method with other model mixture-based PFL methods (FedPer (Arivazhagan et al. 2019), APFL (Deng, Kamani, and Mahdavi 2020), LG-FedAvg (Liang et al. 2020)) and two basic FL methods sharing a similar optimization strategy with ours (i.e., model averaging and proximity regularization) (FedAvg (McMahan et al. 2017), FedProx (Li et al. 2018)). We also considered variants of FedAvg and FedProx for fair comparison on the personalization performance by adopting the strategy proposed in (Wang et al. 2019). They presented a simple personalization method after finishing the update of a single global model through FL: locally updating a global model with small steps using local data of the clients. We denote the variants as FedAvg-LU and FedProx-LU, where LU stands for the local update. We report the best accuracy of each method by tuning its hyperparameters. Our method (SuPerFed-MM and SuPerFed-LM) achieves the best mean local accuracy (i.e., personalization performance) on all datasets. Note that as LG-FedAvg and FedProx only exchange parts of a model, their global model accuracy cannot be evaluated on the global test set at the central server.

Results of the pathological non-IID setting (CIFAR10 and CIFAR100) are shown in Table 1 and those of the realistic scenario (EMNIST and TinyImageNet) are shown in Table 2. Our method outperforms existing model mixture-based PFL methods as well as basic FL methods and its variants in terms of personalization performance.

We implemented all methods using PyTorch (Paszke et al. 2019) version 1.8.1, CUDA version 11.1, and all of our experiments are conducted on ubuntu LTS 18.04 equipped with Intel(R) Xeon(R) Gold 5120 CPU, one NVIDIA Tesla A100 GPU, and 122GB memory.

When to Start Personalization We conducted an experiment on selecting L, which determines the initiation round of PHASE II, the personalization phase. As our proposed method can choose a mixing scheme from either Model Mixing or Layer Mixing, we designed separate experiments per scheme. We denote each scheme by SuPerFed-MM and SuPerFed-LM. By adjusting L by 0.1 increments from 0 to 0.9 of R, we evaluated the mean accuracy of personalized models and the accuracy of the global model. This experiment was done only using the CIFAR10 dataset with

classes. Then, the remaining portion of each dataset is split and used for updating and evaluating personalized models.

For CIFAR10 and CIFAR100 datasets, we select the pathological non-IID split setting proposed in (McMahan et al. 2017), where each client has the same number of samples from at most two classes. For a more realistic scenario suited to PFL, we select another split strategy for EMNIST (Cohen et al. 2017) and TinyImageNet (Le and Yang 2015) datasets by splitting them semantically by labels, with unbalanced sample sizes. A detailed description of the dataset split is described in Appendix. For all experiments, we used a simple 5-layer convolutional neural network as a skeleton of both global and local models. A detailed structure of the network is described in Appendix. We used SGD optimizer with momentum 0.9 throughout all experiments.

Throughout all experiments, if not specified, we set $B = 20$, $E = 10$, $C = 0.1$, $\eta = 0.01$, and $K$ to 100, 100, 1,543,
Ablation Study: Effects of Hyperparameters

Our proposed method has hyperparameters, \( \beta \) and \( \gamma \). Each hyper-parameter is expected to act as (1) constructing a good subspace where both a global model and a local personalized model can benefit, by inducing the orthogonality between local and global models, and (2) preventing undesirable divergence of local updates from the direction of global update.

We conducted experiments according to the combination of \( \beta = \{0.0, 1.0, 2.0, 5.0\} \) and \( \gamma = \{0.0, 0.01, 0.1, 1.0\} \) (i.e., 16 combinations in total), under Model Mixing and Layer Mixing settings respectively with \( R = 100 \) and \( L = \lfloor 0.4R \rfloor = 40 \), which shows the best personalization performance, on the CIFAR100 dataset. Experimental results on these combinations are displayed in Figure 2.

There are noticeable observations supporting the efficacy of our method. First, an ensemble learning strategy by finding a low-loss subspace between local and global models yields personalized models with high performance. We can see that in most settings with non-zero \( \beta \) and \( \gamma \), a mixture of global and local models (i.e., models realized by \( \alpha \in (0, 1) \)) is better than two single models from both endpoints (i.e., a global model (\( \alpha = 0 \)) and local model (\( \alpha = 1 \)). Second, moderate degree of orthogonality regularization (adjusted by \( \beta \)) boosts up the personalization performance of the local model, whereas, too large value (i.e., \( \beta = 5 \)) rather could harm the performance. Third, the proximity regularization restricts learning too much when \( \gamma \) is large, but often helps a global model not to diverge when it is small. Therefore, as we expected, we can make by carefully choosing \( \gamma \).

Meanwhile, when ignoring both \( \beta \) and \( \gamma \), SuPerFed is reduced to FedAvg, while only ignoring \( \beta \), it is reduced to FedProx (with an additional condition that there is no PHASE II, or fixing \( \alpha = 0 \)).

**Calibration** While ensemble learning method can boost up the performance of machine learning models [Lakshminarayanan, Pritzel, and Blundell 2016; Fort, Hu, and Lakshminarayanan 2019; Izmilov et al. 2018; Blundell et al. 2015], it also provides the uncertainty estimation, which is closely related to how well-calibrated a model...
Table 3: Experiment on selecting $L$ for determination of the starting round of the personalization phase (PHASE II). Best mean accuracy is reported for local models given $\alpha \in [0, 1]$: † indicates $\alpha = 0.2$, ‡ for $\alpha = 0.3$, ◆ for $\alpha = 0.4$ and * for $\alpha = 0.5$.

Figure 2: Effects of hyperparameters $\beta$ and $\gamma$. Left group of plots colored in orange is the performance of SuPerFed-MM, right group of plots colored in sky-blue is the performance of SuPerFed-LM. Vertical axis of each subplot stands for the accuracy, and horizontal axis is the range of possible $\alpha$ values from 0 to 1. Metrics on each subplot indicate the performance of the global model (located above) evaluated by global test set at the central server, and the best average performance of the local model (located in parentheses) evaluated by local test set at each client. Error bar indicates a standard deviation of each personalized model realized by different values of $\alpha$.

Conclusion and Future Work

In this paper, we propose a simple and scalable personalization method for federated optimization under the scheme of model mixture method by constructing a low-loss subspace between a local and a global model. With extensive experiments, we empirically demonstrated that SuPerFed outperforms existing model mixture-based PFL methods and two basic FL methods with its variants in two different model mixing scheme: (Model Mixing) and (Layer Mixing). As our method stems from constructing an ensemble-friendly subspace, resulting personalized models are well-calibrated at a nominal cost, which could possibly be robust to the label noise in each client. In future work, we plan to do an in-depth analysis on the unusual behavior of a global model (i.e., high accuracy after PHASE II despite increasing of the loss), and on the convergence rate of our method.

Acknowledgement

This work was supported by the National Research Foundation of Korea (NRF) Grant funded by the Korea Government (MSIT) under Grant No. 2020R1C1C1011063, as well as the Kakao I Research Membership Program by the Kakao
References

Acar, D. A. E.; Zhao, Y.; Matas, R.; Mattina, M.; Whatmough, P.; and Saligrama, V. 2020. Federated learning based on dynamic regularization. In International Conference on Learning Representations.

Allen-Zhu, Z.; Li, Y.; and Liang, Y. 2019. Learning and generalization in overparameterized neural networks, going beyond two layers. Advances in neural information processing systems.

Arivazhagan, M. G.; Aggarwal, V.; Singh, A. K.; and Choudhary, S. 2019. Federated learning with personalization layers. arXiv preprint arXiv:1912.00818.

Ayinde, B. O.; Ianc, T.; and Zurada, J. M. 2019. Regularizing deep neural networks by enhancing diversity in feature extraction. IEEE transactions on neural networks and learning systems, 30(9): 2650–2661.

Blundell, C.; Cornebise, J.; Kavukcuoglu, K.; and Wierstra, D. 2015. Weight uncertainty in neural network. In International Conference on Machine Learning, 1613–1622. PMLR.

Chen, H.-Y.; and Chao, W.-L. 2020. Fedbe: Making bayesian model ensemble applicable to federated learning. arXiv preprint arXiv:2009.01974.

Cohen, G.; Afshar, S.; Tapson, J.; and Van Schaik, A. 2017. EMNIST: Extending MNIST to handwritten letters. In 2017 International Joint Conference on Neural Networks (IJCNN), 2921–2926. IEEE.

Deng, Y.; Kamani, M. M.; and Mahdavi, M. 2020. Adaptive personalized federated learning. arXiv preprint arXiv:2003.13461.

Dinh, C. T.; Tran, N. H.; and Nguyen, T. D. 2020. Personalized federated learning with moreau envelopes. arXiv preprint arXiv:2006.08848.

Dinh, C. T.; Vu, T. T.; Tran, N. H.; Dao, M. N.; and Zhang, H. 2021. FedU: A Unified Framework for Federated Multi-Task Learning with Laplacian Regularization. arXiv preprint arXiv:2102.07148.

Draxler, F.; Veschgini, K.; Salmhofer, M.; and Hamprecht, F. 2018. Essentially no barriers in neural network energy landscape. In International conference on machine learning, 1309–1318. PMLR.

Duan, M.; Liu, D.; Ji, X.; Liu, R.; Liang, L.; Chen, X.; and Tan, Y. 2020. FedGroup: Ternary Cosine Similarity-based Clustered Federated Learning Framework toward High Accuracy in Heterogeneous Data. arXiv preprint arXiv:2010.06870.

Fallah, A.; Mokhtari, A.; and Ozdaglar, A. 2020. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. Advances in Neural Information Processing Systems, 33: 3557–3568.

Finn, C.; Abbeel, P.; and Levine, S. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In International Conference on Machine Learning, 1126–1135. PMLR.

Fort, S.; Hu, H.; and Lakshminarayanan, B. 2019. Deep ensembles: A loss landscape perspective. arXiv preprint arXiv:1912.02757.

Fort, S.; and Jastrzabbski, S. 2019. Large scale structure of neural network loss landscapes. Advances in Neural Information Processing Systems, 32: 6709–6717.

Frankle, J.; Dziugaite, G. K.; Roy, D.; and Carbin, M. 2020. Linear mode connectivity and the lottery ticket hypothesis. In International Conference on Machine Learning, 3259–3269. PMLR.

Garipov, T.; Izmailov, P.; Podoprikhin, D.; Vetrov, D.; and Wilson, A. G. 2018. Loss surfaces, mode connectivity, and fast ensembling of dnn. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, 8803–8812.

Ghosh, A.; Chung, J.; Yin, D.; and Ramchandran, K. 2020. An efficient framework for clustered federated learning. arXiv preprint arXiv:2006.04089.

Guo, C.; Pleiss, G.; Sun, Y.; and Weinberger, K. Q. 2017. On calibration of modern neural networks. In International Conference on Machine Learning, 1321–1330. PMLR.

Hanzely, F; and Richtárik, P. 2020. Federated learning of a mixture of global and local models. arXiv preprint arXiv:2002.05516.

Hsu, T.-M. H.; Qi, H.; and Brown, M. 2019. Measuring the effects of non-identical data distribution for federated visual classification. arXiv preprint arXiv:1909.06335.

Huang, G.; Li, Y.; Pleiss, G.; Liu, Z.; Hopcroft, J. E.; and Weinberger, K. Q. 2018. Averaging weights leads to wider optima and better generalization. arXiv preprint arXiv:1704.00109.

Huang, Y.; Chu, L.; Zhou, Z.; Wang, L.; Liu, J.; Pei, J.; and Zhang, Y. 2021. Personalized cross-silo federated learning on non-iid data. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 35, 7865–7873.

Izmailov, P.; Maddox, W. J.; Kirichenko, P.; Garipov, T.; Vetrov, D.; and Wilson, A. G. 2020. Subspace inference for bayesian deep learning. In Uncertainty in Artificial Intelligence, 1169–1179. PMLR.

Izmailov, P.; Podoprikhin, D.; Garipov, T.; Vetrov, D.; and Wilson, A. G. 2018. Averaging weights leads to wider optima and better generalization. arXiv preprint arXiv:1803.05407.

Jiang, Y.; Konečný, J.; Rush, K.; and Kannan, S. 2019. Improving federated learning personalization via model agnostic meta learning. arXiv preprint arXiv:1909.12488.

Karimireddy, S. P.; Kale, S.; Mohri, M.; Reddi, S. J.; Stich, S. U.; and Suresh, A. T. 2019. SCAFFOLD: Stochastic Controlled Averaging for On-Device Federated Learning. CoRR, abs/1910.06378.

Khodak, M.; Balcan, M.-F.; and Talwalkar, A. 2019. Adaptive gradient-based meta-learning methods. arXiv preprint arXiv:1906.02717.

Konečný, J.; McMahan, H. B.; Yu, F. X.; Richtárik, P.; Suresh, A. T.; and Bacon, D. 2016. Federated learning: Strategies for improving communication efficiency. arXiv preprint arXiv:1610.05492.
### Appendix

| Dataset Name (→) | EMNIST | CIFAR10 & CIFAR100 | TinyImageNet |
|------------------|--------|---------------------|--------------|
| Layer Name (↓)   | conv1  | maxpool1            | conv2        |
|                  | in=1,  | k=(2, 2),           | in=32,       |
|                  | out=32,| k=(5, 5),           | in=32,       |
|                  | k=(5, 5),| s=1,               | k=(5, 5),    |
|                  | s=1,   | ps=0                | s=1,         |
|                  | p=0    |                     | ps=0         |
|                  |        |                     |              |
|                  | maxpool1 | k=(2, 2),     | in=32,       |
|                  |         | k=(5, 5),           | in=32,       |
|                  |         | s=1,               | k=(5, 5),    |
|                  |         | ps=0                | s=1,         |
|                  |         |                     | ps=0         |
|                  |        |                     |              |
|                  | maxpool2 | k=(2, 2),     | in=32,       |
|                  |         | k=(5, 5),           | in=32,       |
|                  |         | s=1,               | k=(5, 5),    |
|                  |         | ps=0                | s=1,         |
|                  |         |                     | ps=0         |
|                  |        |                     |              |
|                  | maxpool3 | k=(2, 2),     | in=32,       |
|                  |         | k=(5, 5),           | in=32,       |
|                  |         | s=1,               | k=(5, 5),    |
|                  |         | ps=0                | s=1,         |
|                  |         |                     | ps=0         |
|                  |        |                     |              |
|                  | fc1     | in=32,              | in=32,       |
|                  |         | out=512,            | out=512,     |
|                  |         | k=(1, 1),           | k=(1, 1),    |
|                  |         | s=1,               | s=1,         |
|                  |         | ps=0                | ps=0         |
|                  |         |                     |              |
|                  | fc2     | in=512,            | in=512,      |
|                  |         | out=62,             | out=10 or 100,|
|                  |         | k=(1, 1),           | k=(1, 1),    |
|                  |         | s=1,               | s=1,         |
|                  |         | ps=0                | ps=0         |

Table 4: Model architecture (both a global and a local model) at each client. Abbreviations in, out, k, s, and p stand for the number of input channels, the number of output channels, the size of kernels, the size of strides, and the size of input paddings.

Model Architecture We used a simple 5-layer convolutional neural network for both a global and a local model located in each client. In other words, each local client has two models with the same structure but different initialization values. Detailed description of the model architecture is shown in Table 4. We used ReLU activation function for all layers except the output layer.

Dataset Split Scenario We used four benchmark datasets CIFAR10, & CIFAR100 (Krizhevsky 2009), EMNIST (Cohen et al. 2017), and TinyImageNet (Le and Yang 2015). As these datasets are not designed for FL setting, we curated them to be suitable for FL setting (i.e., non-IID data across clients).

For both CIFAR10 and CIFAR100, we selected a scheme of pathological non-IID setting proposed in McMahan et al. 2017, where each client has samples from at most two classes, thereby some clients have only a partial labels that are mutually exclusive to some other clients, while all clients having the same sample sizes.

On the other hands, for EMNIST and TinyImageNet datasets we split them semantically (i.e., realistic scenario) with unbalanced sample sizes. It is intended to simulate a more challenging scenario closer to a realistic situation where most AI-driven FL service providers may face.

For EMNIST, though it is composed of 62 classes, it is further grouped into three groups semantically: 10 digits, 26 lowercase alphabets, 26 uppercase alphabets. Therefore, we randomly shuffle and split them semantically (i.e., per each label group) into five clusters. As a result, each cluster has 2 digits, 5 to 6 lowercase and uppercase letters, which are not overlapped across clusters. Then, we randomly split samples within each cluster to clients having sample size from 100 to 800. Thereby, we finally have 1,543 clients having 452 samples on average with the standard deviation of 203. For TinyImageNet, as it has totally 200 classes, we make 10 groups having samples from 20 mutually exclusive classes. In resulting 10 groups, we randomly split samples within each group to clients having sample size from 100 to 400. Thereby, we finally have 390 clients having 252 samples on average with the standard deviation of 88.

Detailed Configurations of Experiments Throughout all experiments, if not specified, we set a local batch size B = 20, local epochs E = 10, a fraction of clients C = 0.1, learning rate $\eta = 0.01$ (SGD optimizer with momentum 0.9), and the number of clients K to 100, 100, 1,543, and 390 for CIFAR10, CIFAR100, EMNIST, and TinyImageNet each. For each dataset, we applied different total rounds $R$, which is 200, 300, 100, and 500 for CIFAR10, CIFAR100, EMNIST, and TinyImageNet each. For the ablation study, we used CIFAR10 dataset with only modification on the total rounds, $R = 100$.

Unusual Behavior of a Global Model One interesting observation is the optimization trajectory of a global model aggregated at the central server (see Figure 3). When the
Effects of Local Batch Size and Local Epochs

In FL, the behavior of local updates is of importance in terms of the convergence, communication and computation trade-offs (Li et al. 2019). Thereby, we conducted more experiments on the effects of both local batch size and local epochs by varying them on CIFAR10 dataset with $R = 100$. In Figure 3, we reported results of experiments similar to Figure 2. It can be easily notified that extremely small local updates (i.e., $E = 1$) harm both local and global model’s performance, while longer local updates (i.e., $E = 20$) did better but sometimes failed (i.e., $E = 20$ & $B = 4$). Local batch size also showed the same trend. When the batch size is too small, we cannot achieved an optimal performance, which may be due to too noisy update at each client can cause divergent behaviors across clients. Recall that our proposed method basically pursues an ensemble of a local and a global model at each client by mixing (i.e., combining) different combination of the local and the global model at every training batch update, it is favorable to retain some degree of stochasticity on local updates (i.e., maintaining sufficient number of iterations). Thus, we conclude that selecting a proper configuration of local update (e.g., $B = 10 \sim 20$) is more helpful in constructing an efficient client-specific subspace for a personalization, while considering computation and communication efficiency.