Supplementary Material for FedSelect: Personalized Federated Learning with Customized Selection of Parameters for Fine-Tuning

Rishub Tamirisa[†], Chulin Xie[†,‡], Wenxuan Bao[†,‡], Andy Zhou[†], Ron Arel[†], Aviv Shamsian[§]
[†]Lapis Labs [‡]University of Illinois Urbana-Champaign [§]Bar-Ilan University
{rishubt2,chulinx2,wbao4,andyz3,ronarel2}@illinois.edu aviv.shamsian@live.biu.ac.il

A. Experimental Setup

In this section, we describe the details of our experimental setup used for the results in the main text. We include information about our ablation study as well as hyperparameters used for training both FEDSELECT and the compared baselines.

A.1. Implementation Details

Models & Datasets. We conduct experiments on 4 datasets: CIFAR-10 [4], CIFAR10-C [3], Mini-ImageNet [14], and OfficeHome [13]. CIFAR10 contains 60,000 image samples across 10 labeled image categories, from which 50,000 are selected for training and the remaining 10,000 for testing; CIFAR10-C contains the same set of training samples as CIFAR10, with different types of image corruptions (severity $= 5$) for each client. Mini-ImageNet contains 50,000 training samples and 10,000 testing samples, across 100 image categories. Finally, the OfficeHome dataset contains data across 4 domain shifts and 65 image categories with 2162, 3909, 3969, 3892 training samples for each of the 4 domains, respectively. The corresponding test set sizes are 265, 456, 470, and 465 testing samples.

Data Partioning For all experiments, training samples were distributed to each client in a non-IID fashion. For experiments involving either CIFAR10 or CIFAR10-C, training samples for each client were additionally truncated to a fixed size N_k; however the test set size for each client remained fixed at 200. We follow the learning setups presented in [5, 15] and limit the number of data samples per client to ensure the necessity of federated learning to achieve optimal performance as opposed to pure local training. We define the shard s as the number of classes assigned to a client, which is set to $s = 2$ in all CIFAR10 & CIFAR10-C experiments and $s = 10$ for the Mini-ImageNet experiment. For the OfficeHome experiment, there are 4 clients for each domain shift, such that each client is allocated to the full dataset of its domain as in [12].

Types of Distributional Shift. For the CIFAR10 and Mini-ImageNet experiments, we use a label-based partition; therefore there is label shift among the clients' data distributions. For the CIFAR10-C experiments, while using a labeled-based partition, we also apply a random type of image corruption to each client, which introduces feature shift. Finally for the OfficeHome experiments, there is primarily feature shift since the images from each client are from different domains.

Compared Algorithms. In addition to FedAvg [8], we compare our method to several *full model personalization* methods: local-only training, FedAvg with local fine-tuning (FedAvg + FT), and Ditto [6] which personalizes clients via a multi-task learning objective. We also compare against *partial model personalization* methods: FedBABU [9] which only updates and aggregates the feature extractor; FedRep [2] and FedPer [1] which aggregate client feature extractors and personalize classifier heads; LG-FedAvg [7] which personalizes the clients' feature extractors; FedPAC [15] which performs local-global feature alignment via classifier collaboration.

Hyperparameters. For all methods, we use 3 local training epochs using stochastic gradient descent (SGD) optimizer momentum. We tuned the learning rate over $\{0.1, 0.01, 0.001\}$ for all of the compared baselines. For the parameter decoupling methods [1, 2, 7, 9], this entails setting the learning rates for the respective feature extractors and classifier heads that are locally updated. As a result, we used a learning rate of 0.01 for the CIFAR-10 and CIFAR10-C experiments, and a learning rate of 0.001 for the OfficeHome and Mini-ImageNet experiments. For FEDSELECT, we follow the recommendation

	$\alpha = 0.05$	$\alpha = 0.30$	$\alpha = 0.50$	$\alpha = 0.80$
$p = 0.01$	81.35	81.35	81.85	80.65
$p = 0.05$	80.10	82.25	82.20	81.40
$p = 0.20$	80.55	81.70	81.65	82.15
$p = 0.50$	82.05	82.05	81.60	81.20

Table 1. Performance (% mean client test accuracy) of FEDSELECT on CIFAR-10 when varying both the personalization limit α and the personalization rate p.

α	CIFAR10	CIFAR10-C
0.05	80.10	68.30
0.10	81.25	69.05
0.20	81.15	70.65
0.30	**82.25**	70.60
0.40	81.10	70.55
0.50	82.20	**72.05**
0.60	81.60	70.90
0.80	81.40	71.40

Table 2. Performance (% mean client test accuracy) of FEDSELECT on both CIFAR-10 and CIFAR-10C when varying the personalization limit α.

of choosing different learning rates for global and personal parameters given in the partial model personalization framework [11]. Specifically, we use a learning rate of 0.1 for the personal parameters and 0.001 for the global parameters for the CIFAR-10 experiments. Our reported experiments (excluding Table 1) use a personalization rate of $p = 0.05$. Extended results for a grid search across a selection of values for p and α are described in Appendix B.2. We set the personalization hyperparameter λ of Ditto to 0.75 after tuning over $\{0.25, 0.50, 0.75\}$, which follows the setting used by FedBABU [9].

B. Additional Experimental Results

In this section we provide further experimental results and clarify additional details of the ablation study described in the main text.

B.1. Ablation Study Details

One of the components of our ablation study of the key design choices involved in FEDSELECT involves choosing 4 different layers (denoted as Layer A/B/C/D in Section 5.2 of the main text) for personalization. We selected Layers A, B, C, and D from ResNet18 given by their PyTorch [10] layer-names in the library-provided ResNet18 implementation: Layer A refers to 'conv1.weight'; Layer B refers to 'layer1.0.conv1.weight'; Layer C refers to 'layer2.0.conv1.weight'; Layer D refers to 'layer4.1.conv1.weight'.

B.2. Effect of p and α on personalization

For the following described experiments in this subsection, we use the same data partition split as in the CIFAR-10 experiments presented in the main text; 100 training samples and 200 testing samples are allocated for each client with 10 clients undergoing full participation FL, and each client is allocated 2 image category classes.

Varying the personalization rate p. We present additional experimental results for varying the personalization rate p alongside the personalization limit α for CIFAR-10 in Table 1. We observe the best performance occurs when $p = 0.05$ for a limit of $\alpha = 0.30$. We also note that when $p > \alpha$, the personalization limit is reached after the first round. In general, we suggest using a smaller p with a larger α when attempting to personalize many client parameters, rather than personalizing with a very high rate ($p > 0.20$).

Varying the personalization limit α. We showcase the performance of FEDSELECT on CIFAR-10 under a wide range of values for $\alpha \in [0, 1]$ in Table 2 for a fixed personalization rate $p = 0.05$. From Section 4.4 of the main text, we have that FEDSELECT intuitively performs an interpolation of the two extremes of personalization: pure federated averaging ($\alpha = 0.0$) and eventual pure local training $\alpha = 1.0$. We generally recommend using middle-ground values of $\alpha \in [0.3, 0.5]$.

Method	$N_k = 20$	$N_k = 40$	$N_k = 100$	$N_k = 200$
FedAvg	23.55	24.70	27.70	26.90
FedAvg + FT	37.65	72.35	75.30	83.50
FedPAC	68.30	67.00	77.20	81.65
FedRep	70.10	73.00	67.60	78.15
FedPer	55.70	75.15	75.40	83.15
FedBABU	33.10	66.10	75.45	81.80
Ditto	37.25	71.55	72.75	81.65
LG-FedAvg	71.45	74.70	77.65	83.90
FedSelect	**72.25**	**78.20**	**82.25**	**84.85**

Table 3. Comparison of performance (% mean client test accuracy) on CIFAR-10 when varying the training data size N_k per client c_k. Each client was assigned with 2 classes.

B.3. Robustness to sample size.

We present the mean client performance of FEDSELECT and the compared baselines for client training data sizes $N_k \in \{20, 40, 100, 200\}$ in Table 3, where k represents the k-th client c_k. To ensure a fair comparison of performance across each setting of N_k, we designated the same test sets for each client, which had 200 testing samples each. Notably, we observe that FEDSELECT maintains superior performance across each data size setting, particularly for low client training dataset sizes $N_k \leq 40$, where algorithms such as FedBABU, Ditto, and FedPer suffer a significant performance decrease.

C. Convergence Analysis

In this section, we analyze the convergence of FEDSELECT. Specifically, we first prove in Theorem 1 that the mask for each client is guaranteed to converge in finite communication rounds. We then prove in Theorem 2 that once the masks for all clients converge, the model parameters have the same convergence guarantee as block stochastic gradient descent.

Theorem 1 (Convergence of masks)**.** Under full client participation, the masks for all clients converge in finite communication rounds.

Proof. For each client c_k, the set of personalized parameter indices are monotonically increasing, i.e., $\text{Idx}(v_k^0) \subset \text{Idx}(v_k^1) \subset \cdots \subset \text{Idx}(v_k^T)$. Meanwhile, the cardinality of this set is upper bounded by αd, where α is the personalization limit and d is the dimensionality of the parameter. Therefore, the masks for all clients converge in finite communication rounds. □

Theorem 2 (Convergence of parameters)**.** Under full client participation, when the number of local steps $\tau = 1$ and participation, after the masks for all clients converge, FEDSELECT has the identical convergence guarantee as centralized block stochastic gradient descent (block SGD).

Proof. We let $\theta_k[i]$ denote the i-th parameter on client c_k and $m_k[i]$ denote its corresponding mask, where $m_k[i] = 0$ means the parameter is globally shared and $m_k[i] = 1$ means the parameter is personalized (on client c_k). If a parameter is global, i.e., shared across several clients, we use $\theta[i]$ to denote such parameter. After the masks for all clients converge, FEDSELECT becomes an optimization problem over the union of the global parameters and each client's personalized parameters:

- Global parameters: $U = \{\theta[i] : \forall i, \exists k, \text{ s.t. } m_k[i] = 0\}$
- Personalized parameters: $V = \{\theta_k[i] : \forall k, i, \text{ s.t. } m_k[i] = 1\}$

and the optimization objective is

$$F(U, V) = \frac{1}{N} \sum_{k=1}^{N} f_k(u_k, v_k)$$

where $u_k \subset U$ and $\cup_{k=1}^{N} v_k = V$.

For example, consider a system with 3 clients and corresponding masks

$$m_1 = [1, 1, 0, 0], m_2 = [1, 0, 1, 0], m_3 = [1, 0, 0, 1]$$

then, we have

$$\begin{aligned} u_1 &= [\theta_1[3], \theta_1[4]] = [\theta[3], \theta[4]], \quad v_1 = [\theta_1[1], \theta_1[2]], \\ u_2 &= [\theta_2[2], \theta_1[4]] = [\theta[2], \theta[4]], \quad v_2 = [\theta_2[1], \theta_2[3]], \\ u_3 &= [\theta_3[2], \theta_3[3]] = [\theta[2], \theta[3]], \quad v_3 = [\theta_3[1], \theta_3[4]], \end{aligned}$$

and

$$\begin{aligned} U &= [\theta[2], \theta[3], \theta[4]], \\ V &= [\theta_1[1], \theta_1[2], \theta_2[1], \theta_2[3], \theta_3[1], \theta_3[4]] \end{aligned}$$

For clarification, we use the original client and parameter indices k and i as the indices of U and V.

After the masks converge, in each communication round, each client conducts LocalAlt (see Algorithm 2), which is SGD on personalized parameters v followed by SGD on global parameters u. After LocalAlt, the global parameters are aggregated on the server. Next, we show that each communication round is equivalent to central block SGD on U and V alternatively.

- Update personalized parameters: Each client independently optimize its personalized parameter. This is equivalent to centralized SGD w.r.t. V, since

$$\frac{\partial F}{\partial V_k[i]} = \frac{\partial F}{\partial \theta_k[i]}$$

- Update global parameters: Eech client first optimize its local copy global parameter. Then, each client's updated local copy will be uploaded to the server for aggregation, i.e., for the i-th parameter,

$$u_k^+[i] \leftarrow u_k[i] - \gamma_u \frac{\partial f_k(u_k, v_k)}{\partial u_k[i]}, \forall k \qquad \text{(local update)}$$

$$u_k[i] \leftarrow \frac{1}{\sum_{k=1}^N (1 - m_k[i])} \sum_{k=1}^N (1 - m_k[i]) u_k^+[i] \qquad \text{(aggregation)}$$

  This is equivalent to

$$u_k[i] \leftarrow u_k[i] - \left(\gamma_u \frac{N}{\sum_{k=1}^N (1 - m_k[i])} \right) \cdot \frac{\partial F}{\partial U_k[i]}$$

  which is centralized SGD w.r.t. U.

Therefore, the optimization process is numerically equivalent to block SGD with U, V alternatively. □

References

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

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

[3] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *Proceedings of the International Conference on Learning Representations*, 2019. 1

[4] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009. 1

[5] Ang Li, Jingwei Sun, Binghui Wang, Lin Duan, Sicheng Li, Yiran Chen, and Hai Li. Lotteryfl: Personalized and communication-efficient federated learning with lottery ticket hypothesis on non-iid datasets, 2020. 1

[6] T. Li, S. Hu, A. Beirami, and V. Smith. Ditto: Fair and robust federated learning through personalization. In *International Conference on Machine Learning*, pages 6357–6368. PMLR, 2021. 1

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

[8] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proc. of Int'l Conf. Artificial Intelligence and Statistics (AISTATS)*, 2017. 1

[9] Jaehoon Oh, SangMook Kim, and Se-Young Yun. FedBABU: Toward enhanced representation for federated image classification. In *International Conference on Learning Representations*, 2022. 1, 2

[10] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019. 2

[11] K. Pillutla, K. Malik, A. Mohamed, M. Rabbat, M. Sanjabi, and L. Xiao. Federated learning with partial model personalization. In *International Conference on Machine Learning*, 2022. 2

[12] Benyuan Sun, Hongxing Huo, Yi Yang, and Bo Bai. Partialfed: Cross-domain personalized federated learning via partial initialization. *Advances in Neural Information Processing Systems*, 34:23309–23320, 2021. 1

[13] Hemanth Venkateswara, José Eusébio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1

[14] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *Neural Information Processing Systems*, 2017. 1

[15] Jian Xu, Xinyi Tong, and Shao-Lun Huang. Personalized federated learning with feature alignment and classifier collaboration. In *The Eleventh International Conference on Learning Representations*, 2023. 1