Supplementary Material for MetaVers: Meta-Learned Versatile Representations for Personalized Federated Learning

We summarize the contents of the Supplementary material as follows. Section 1 provides the impelemntation details of the experiments, including the dataset splitting and the hyperparameter optimizations. Table 1 in Section 1 fully shows the result of Standard PFL Benchmarks with SEM (Standard Error of the Mean). Additional experiments are presented in Section 2. Finally, Section 5 provides the proofs of Lemma 1 and Theorem 1 in the main paper.

1. Implementation Details

Simulation environment: We implement `MetaVers` and the baseline methods using the PyTorch framework. The experiments were conducted on NVIDIA Quadro RTX 8000, having 48GB of memory with python 3.8.5, PyTorch 1.7.1, Torchvision 0.8.2, and CUDA 11.0.

Re-implementation of prior methods: Here we describe the re-implementation of the existing methods. As stated in the main paper, we tested all related methods in Standard PFL Benchmarks that is used in [1, 18], i.e., Standard PFL Benchmarks. For **FedProto** in Standard PFL Benchmarks, we follow the same configurations in the original paper of FedProto [19]. We search for the best hyperparameter λ, which is the weighting of the prototype-based regularization loss. It is grid searched in {0, 0.1, 1, 2}. In the original paper, FedProto assumes the full participation of clients at every round, which is impractical. Therefore, we allow full participation of all clients only for the first round of communication, then restrict the number of active clients to 5 as we did for other methods. For **Per-FedAvg**, experiments are conducted with the first-order (FO) MAML [7], and we grid search for the best β value in {0.01, 0.005, 0.001}, which is the inner-update learning rate. For **kNN-Per**, we tune the interpolation hyperparameter λ_m via grid search on {0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9} and set the number of nearest neighbors k as 10 by following the setting of the original paper [13]. For **FedRep**, the number of local updates for the representation is set to 1 for CIFAR-10 and 5 for CIFAR-100 by following the settings of the original paper [4]. In the case of CINIC-10, the corresponding update number is searched over the range of {1, 5}. For the case of **FedBABU**, 10 fine-tuning steps are allowed in the testing for all cases.

1.1. Episode Construction

Number of shots in support and query: For the experiments in the main paper, we construct the training episode $\mathcal{E}$ with the support set with K samples per class, and the query set with Q samples per class. For the Generalization on Novel Classes experiment in subsection 4.2, episodes are composed of $(K, Q) = (5, 15)$ for all datasets, i.e., CIFAR-100 and miniImageNet. Therefore, each training episode is set to have 20 samples per class. For Standard PFL Benchmarks, the total number of samples that are distributed to each client is different for different datasets due to the widely-varying number of samples per client. (K, Q) value differs depending on the dataset because the number of samples that each local client can use to construct the episode is different. We set (K, Q) of each client to be $(20, 30)$, $(10, 20)$, and $(25, 35)$ for CIFAR-10, CINIC-10, and CIFAR-100, respectively.

Episode for extremely-small local clients: For the experiment on CIFAR-100 with 500 clients, some clients are insufficient data samples to create such episodes because the average number of samples per class in each client is just 8. Consequently, we compose the support set with five or fewer samples and let the query set contain the remaining samples. Table 2 shows the test accuracies of over 500 clients on the CIFAR-100 according to the value of K. In the case of $K = 3$ is shown to be the best, so the support set of the training episode is constructed with 3 samples for the case of Client 500 in CIFAR-100. Due to the same reason, $(K, Q) = (20, 30)$ is used for 500 clients on the CINIC-10, which is a slightly smaller episode than 50 and 100 client cases with $(K, Q) = (25, 35)$.

1.2. Hyperparameters for `MetaVers`

For `MetaVers`, we use the validation set for hyperparameter tuning and early stopping. We adopt the Adam optimizer and search for the optimal learning rates over the range of {0.1, 0.05, 0.01, 0.005, 0.001}. Also, we search the best hyperparameter γ over the range of {0.1, 0.2, 0.3,

Table 1. Test accuracy (± SEM) on CIFAR-10, CIFAR-100, and CINIC-10.

	CIFAR-10			CIFAR-100			CINIC-10		
# clients	50	100	500	50	100	500	50	100	500
# samples/client	800	400	80	800	400	80	1800	900	180
Local	84.8 ± 0.1	82.1 ± 0.2	76.2 ± 0.2	49.8 ± 0.2	44.5 ± 0.4	31.0 ± 0.3	58.4 ± 0.2	57.4 ± 0.4	50.3 ± 0.0
FedAvg [14]	56.4 ± 0.5	59.7 ± 0.5	54.0 ± 0.5	23.6 ± 0.2	24.0 ± 0.2	20.4 ± 0.0	45.6 ± 0.4	44.7 ± 0.5	45.7 ± 0.5
LG-FedAvg [11]	87.9 ± 0.3	83.6 ± 0.7	64.7 ± 0.7	43.6 ± 0.2	37.5 ± 0.9	20.3 ± 0.5	59.5 ± 1.1	59.9 ± 2.1	52.5 ± 0.8
pFedMe [5]	86.4 ± 0.8	85.0 ± 0.3	80.3 ± 0.5	49.8 ± 0.5	47.7 ± 0.4	32.5 ± 0.8	69.9 ± 0.5	68.9 ± 0.7	58.8 ± 0.1
FedProto [19]	85.9 ± 0.7	79.0 ± 0.4	51.0 ± 0.0	47.8 ± 0.5	17.8 ± 0.1	10.9 ± 0.1	58.2 ± 0.7	40.3 ± 0.8	26.0 ± 0.0
Per-FedAvg [2]	71.1 ± 1.5	79.1 ± 3.7	67.7 ± 1.9	38.2 ± 2.0	34.1 ± 0.4	32.8 ± 1.7	53.8 ± 0.8	53.5 ± 0.6	59.6 ± 0.7
pFedHN [18]	90.2 ± 0.6	87.4 ± 0.2	83.2 ± 0.8	60.0 ± 1.0	52.3 ± 0.5	34.1 ± 0.1	70.4 ± 0.4	69.4 ± 0.5	64.2 ± 0.1
pFedGP [1]	89.2 ± 0.3	88.8 ± 0.2	87.6 ± 0.4	63.3 ± 0.1	61.3 ± 0.2	50.6 ± 0.2	71.8 ± 0.3	71.3 ± 0.4	68.1 ± 0.3
FedPer [2]	83.8 ± 0.8	81.5 ± 0.5	76.8 ± 1.2	48.3 ± 0.6	43.6 ± 0.2	25.6 ± 0.3	70.6 ± 0.2	68.4 ± 0.5	62.2 ± 0.1
FedRep [4]	82.4 ± 1.5	80.7 ± 1.0	77.3 ± 0.8	45.1 ± 2.8	38.8 ± 1.1	30.2 ± 0.4	67.1 ± 1.1	64.7 ± 0.0	61.5 ± 0.5
kNN-Per [13]	89.6 ± 0.6	89.5 ± 0.4	84.8 ± 0.4	61.8 ± 0.3	56.0 ± 0.3	38.7 ± 0.7	71.8 ± 0.2	72.0 ± 0.2	69.2 ± 0.6
FedBABU [15]	87.2 ± 1.0	86.2 ± 0.6	85.5 ± 0.5	53.4 ± 0.4	52.3 ± 0.7	49.0 ± 0.6	68.7 ± 0.4	66.5 ± 0.2	67.8 ± 1.3
Ours									
(K, Q)	$(20, 30)$	$(20, 30)$	$(20, R^*)$	$(10, 20)$	$(10, 20)$	$(3, R^*)$	$(25, 35)$	$(25, 35)$	$(20, R^*)$
`MetaVers` (only $\mathcal{L}_T$)	90.3 ± 0.2	89.7 ± 0.2	89.6 ± 0.1	64.7 ± 0.2	62.9 ± 0.3	46.6 ± 0.2	72.6 ± 0.4	72.7 ± 0.2	71.9 ± 0.2
`MetaVers` (only $\mathcal{L}_S$)	89.9 ± 0.3	88.6 ± 0.1	88.1 ± 0.1	$\mathbf{66.7 \pm 0.1}$	64.6 ± 0.2	54.4 ± 0.3	72.8 ± 0.3	72.5 ± 0.2	71.7 ± 0.1
`MetaVers`	$\mathbf{90.8 \pm 0.3}$	$\mathbf{90.2 \pm 0.2}$	$\mathbf{89.9 \pm 0.3}$	$\mathbf{66.7 \pm 0.0}$	$\mathbf{64.8 \pm 0.1}$	$\mathbf{55.8 \pm 0.1}$	$\mathbf{73.2 \pm 0.4}$	$\mathbf{73.2 \pm 0.3}$	$\mathbf{72.5 \pm 0.1}$

Samples/client indicates the mean number of samples per local client in each case.
(K, Q) indicates the number of samples per class corresponding to the support set and the query set constituting the training episode.
R^* indicates the remaining data samples per class after securing K samples by prioritizing the support set configuration.

Table 2. Searching for the best support set size for the Client-500 case of the CIFAR-100 benchmark.

K (Samples per Class)	1	2	**3**	4	5
Accuracy (%)	53.4	55.5	**55.8**	54.6	54.6

0.4, 0.5, 0.6, 0.7, 0.8, 0.9}, which balances the cross-entropy loss term and the triplet loss term. Table 3 shows the effect of the balancing hyparparmeter γ.

1.3. Interval Value W

The server considers the past global margin values with a fixed interval of W rounds to stabilize the global margin value. We set that W to be 50, so the moving average of the past global margin values over 50 rounds is used. Also, for the first 50 rounds of training, where not enough margin values are collected to calculate the global margin value, we use only the cross-entropy loss term. After the 50 rounds, we adopt triplet loss to find the large-margin representation, as stated in the main paper.

1.4. Details for Standard PFL Benchmarks

Accuracy table: Table 1 fully shows the test accuracy with SEM (Standard Error of the Mean) over three random seeds of all cases. Also, in the bottom part of the table, we show the full accuracies with SEM of loss term ablation studies. As shown in the main paper, i.e., the combined loss increases accuracies in the CINIC-10 cases, consistent gains are shown for all cases from three benchmarks.

Data split: As aforementioned, we follow the same data splits in [1, 18]. The numbers of training samples of the datasets are 40K, 40K, and 90K for CIFAR-10, CIFAR-100, and CINIC-10, respectively. In each local client, the training and test sets have the same class configuration, but the different clients can have different class configurations, which means the heterogeneous setting. To construct the setting, for each client i, and class k, a uniform probability $p_{i,k} \sim U(.4, .6)$ is sampled to assign $p_{i,k} / \sum_j p_{j,k}$ samples of the class k to the client i. Therefore, local clients have samples that do not overlap, and the number of samples per client is not the same but slightly different.

1.5. Generalization on Novel Classes

Implementation details: Following the same model architecture introduced in [7, 16], we use a CNN model with four 3x3 convolutional layers where batch normalization,

Table 3. Test accuracy (± SEM) of MetaVers under varting hyparparmeter γ on CIFAR-10, CIFAR-100, and CINIC-10

	CIFAR-10			CIFAR-100			CINIC-10		
# clients	50	100	500	50	100	500	50	100	500
balancing parameter γ									
0.1	90.6 ± 0.3	90.0 ± 0.2	89.8 ± 0.2	65.1 ± 0.2	64.7 ± 0.1	52.1 ± 0.2	72.0 ± 0.6	72.4 ± 0.4	71.7 ± 0.1
0.2	90.7 ± 0.3	90.1 ± 0.2	**89.9 ± 0.3**	65.7 ± 0.1	63.8 ± 0.2	53.6 ± 0.3	72.4 ± 0.6	72.8 ± 0.4	72.1 ± 0.1
0.3	90.6 ± 0.3	**90.2 ± 0.2**	89.7 ± 0.2	66.2 ± 0.3	64.4 ± 0.0	54.1 ± 0.4	72.8 ± 0.4	72.9 ± 0.3	72.2 ± 0.1
0.4	**90.8 ± 0.3**	89.8 ± 0.2	89.6 ± 0.2	66.4 ± 0.2	64.5 ± 0.2	55.2 ± 0.2	**73.2 ± 0.5**	73.1 ± 0.3	72.3 ± 0.1
0.5	**90.8 ± 0.4**	89.9 ± 0.2	89.6 ± 0.1	**66.7 ± 0.1**	**64.8 ± 0.1**	54.2 ± 0.2	**73.2 ± 0.4**	**73.2 ± 0.3**	**72.5 ± 0.1**
0.6	90.7 ± 0.4	89.9 ± 0.1	89.4 ± 0.3	**66.7 ± 0.0**	**64.8 ± 0.1**	**55.8 ± 0.1**	**73.2 ± 0.4**	**73.2 ± 0.4**	72.4 ± 0.1
0.7	90.6 ± 0.4	89.8 ± 0.2	89.5 ± 0.2	66.2 ± 0.1	**64.8 ± 0.1**	55.7 ± 0.3	73.1 ± 0.4	**73.2 ± 0.3**	**72.5 ± 0.1**
0.8	90.1 ± 0.5	89.9 ± 0.1	89.1 ± 0.4	**66.7 ± 0.1**	64.5 ± 0.3	55.6 ± 0.3	73.0 ± 0.5	**73.2 ± 0.3**	72.3 ± 0.0
0.9	89.9 ± 0.5	89.2 ± 0.1	**89.9 ± 0.3**	66.6 ± 0.1	**64.8 ± 0.1**	55.7 ± 0.3	**73.2 ± 0.6**	**73.2 ± 0.4**	**72.5 ± 0.2**

ReLU, and a 2 x 2 max-pooling layer follow each one. In the case of CIFAR-100, we omit the last two max-pooling layers for the feature map size as done in [16]. We optimize the model using Adam optimizer with a learning rate of 0.001. Also, we use the learning rate decaying at every 5,000 epochs with a decaying factor of 0.5.

2. Additional Ablation Study

2.1. Fixed Margin

We additionally explore the optimal margin value with fixed-margin value in the range of {0.25, 0.5, 0.75, 1.0, 1.25, 1.5, 3.0} and adaptive margin when the learning is done only with the triplet loss term. As shown in Table 4, when the margin value is 0.75, it generally shows the best or near-best performance in all cases. Therefore, we use the fixed-margin value 0.75 when comparing MetaVers and only-triplet-loss cases of Table 4 in the main paper and Table 1 in Supplementary material.

2.2. Effect of the Way in the Few-Shot Episode

We examine the impact of the way in the episodic training, which is the number of different classes that are selected in each episode. 'Way' indicates the number of classes to be classified in each episode. Therefore, in the case of CIFAR-10, there are only two classes in each local dataset, so the ablation experiments are conducted on CINIC-10 and CIFAR-100. First, we evaluate the model by changing the way values from 2 to 10 for CIFAR-100 and 2 to 4 for CINIC-10. We want to point out that the maximum way can be adopted for CIFAR-100 and CINIC-10 are 10 and 4, respectively. In addition, we tested the scenario in that each client creates an episode in a randomly selected way in the possible ranges of ways. To be specific, in CIFAR-100, we randomly assign the number of ways from 2 to 10 for each client, and we select ways from 2 to 4 for CINIC-10. Table 5 shows that the way value significantly impacts performance, and keeping it as large as possible is the best choice for achieving better performance. However, even when the way is reduced to 50% of the maximum (refer to way 5 for CIFAR-100 and way 2 for CINIC-10), it is still comparable to other runner-up algorithms or achieves state-of-the-art performance. Besides, the performance does not decrease significantly in the random way setting. We emphasize that MetaVers successfully achieves a generalized representation even when each local client is allowed to construct its own episode with a randomly-varying number of target classes.

2.3. Fine-tuning in Testing

Prior methods that aim to train a global representation, such as FedBABU of [15], acquire further performance gain via fine-tuning in testing. The reasons for the gain are the further personalization of the head, i.e., classifiers, for well-fitted classification weights, and the further training of the body, i.e., a feature extractor, for better representation ability. However, MetaVers trains a representation via distance-based meta-learning, so we conjecture that the representation of MetaVers does not require any further fine-tuning steps. Also, MetaVers utilizes prototypes as classifiers. When the fine-tuning is done in testing, i.e., the learned representation is further trained via few-shot episodes at each local client, MetaVers shows performance degradation due to the overfitting to the local training data. To be specific, the performance degrades from 64.8% to 59.8% when a single step of fine-tuning is used for the case of CIFAR-100 with 100 clients. It implies that MetaVers achieves a sufficiently trained global representation that performs well across heterogeneous clients via federation.

Table 4. `MetaVers` with only $\mathcal{L}_T$

	CIFAR-10			CIFAR-100			CINIC-10		
# clients	50	100	500	50	100	500	50	100	500
$\mathcal{L}_T$ with fixed margin $\mathfrak{m}^*$									
0.25	90.2 ± 0.2	89.4 ± 0.2	89.0 ± 0.1	$\mathbf{64.7 \pm 0.2}$	62.6 ± 0.2	46.4 ± 0.2	72.4 ± 0.4	72.5 ± 0.2	71.6 ± 0.0
0.50	$\mathbf{90.3 \pm 0.3}$	89.6 ± 0.0	$\mathbf{89.6 \pm 0.1}$	64.5 ± 0.2	$\mathbf{62.9 \pm 0.3}$	46.2 ± 0.3	72.5 ± 0.4	$\mathbf{72.7 \pm 0.2}$	$\mathbf{71.9 \pm 0.2}$
0.75	90.2 ± 0.3	$\mathbf{89.7 \pm 0.1}$	$\mathbf{89.6 \pm 0.1}$	64.6 ± 0.1	62.7 ± 0.2	46.5 ± 0.3	$\mathbf{72.6 \pm 0.5}$	$\mathbf{72.7 \pm 0.2}$	$\mathbf{71.9 \pm 0.2}$
1.00	$\mathbf{90.3 \pm 0.2}$	89.5 ± 0.1	89.4 ± 0.2	64.4 ± 0.3	62.4 ± 0.1	$\mathbf{46.6 \pm 0.3}$	72.4 ± 0.2	$\mathbf{72.7 \pm 0.2}$	71.7 ± 0.0
1.25	90.2 ± 0.2	89.6 ± 0.2	$\mathbf{89.6 \pm 0.1}$	64.2 ± 0.1	62.6 ± 0.2	46.2 ± 0.2	72.5 ± 0.3	$\mathbf{72.7 \pm 0.2}$	71.5 ± 0.1
1.50	90.2 ± 0.2	$\mathbf{89.7 \pm 0.2}$	89.5 ± 0.1	64.2 ± 0.2	62.6 ± 0.2	$\mathbf{46.6 \pm 0.2}$	$\mathbf{72.6 \pm 0.4}$	72.5 ± 0.1	71.5 ± 0.0
3.00	90.2 ± 0.3	89.6 ± 0.1	89.4 ± 0.2	64.0 ± 0.1	62.5 ± 0.1	46.5 ± 0.3	72.4 ± 0.5	72.4 ± 0.2	71.7 ± 0.0
$\mathcal{L}_T$ with adaptive margin	$\mathbf{90.3 \pm 0.3}$	$\mathbf{90.0 \pm 0.3}$	$\mathbf{89.7 \pm 0.1}$	64.3 ± 0.1	62.5 ± 0.2	$\mathbf{46.7 \pm 0.2}$	71.5 ± 0.3	71.4 ± 0.1	70.7 ± 0.1
`MetaVers`	$\mathbf{90.8 \pm 0.3}$	$\mathbf{90.2 \pm 0.2}$	$\mathbf{89.9 \pm 0.3}$	$\mathbf{66.7 \pm 0.0}$	$\mathbf{64.8 \pm 0.1}$	$\mathbf{55.8 \pm 0.1}$	$\mathbf{73.2 \pm 0.4}$	$\mathbf{73.2 \pm 0.3}$	$\mathbf{72.5 \pm 0.1}$

Table 5. `MetaVers` with the lower way setting.

	CIFAR-100		
	# clients		
Way	50	100	500
10	$\mathbf{66.7 \pm 0.0}$	$\mathbf{64.8 \pm 0.1}$	$\mathbf{55.8 \pm 0.1}$
5	64.7 ± 0.2	63.5 ± 0.2	51.6 ± 0.2
2	57.5 ± 0.4	56.2 ± 0.1	41.8 ± 0.4
Random way	64.7 ± 0.2	63.0 ± 0.4	51.4 ± 0.2

	CINIC-10		
	# clients		
Way	50	100	500
4	$\mathbf{73.2 \pm 0.4}$	$\mathbf{73.2 \pm 0.3}$	$\mathbf{72.5 \pm 0.1}$
3	72.4 ± 0.4	72.2 ± 0.1	71.6 ± 0.1
2	70.6 ± 0.6	70.7 ± 0.2	69.9 ± 0.2
Random way	71.6 ± 0.5	72.1 ± 0.1	71.4 ± 0.1

2.4. Discussions on Additional Communications Burden

The additional communication burden from the margin-value sharing between clients and the server is negligible because the margin value is a simple scalar that is extremely smaller than the model parameter size.

3. Additional Experiments

3.1. Comparison with Centralized Meta-Learning

We try to compare `MetaVers` with the centralized version of meta-learning. We conjecture that when the data is *not* decentralized, then the meta-learning can explore more diverse episodes in the whole dataset so that the generalization performance can be improved further. We want to show how much `MetaVers` can converge to the *oracle* centralized setting. For the centralized version, we set all settings identically to `MetaVers`, including hyperparameters and inference procedures. The only difference is that it can create the training episode by accessing the full dataset. But surprisingly, according to the results in Table 6, the performance difference between our decentralized framework, i.e., `MetaVers` and the centralized version is not significant except in the case of Client 500 in CIFAR-100, where the extremely small number of samples are used for constructing local episodes. We can conclude that our method is an effective model that generalizes well on decentralized data samples, even with the limited diversity of episodes due to the decentralized regime.

4. Evaluation on Other Benchmarks and Larger Architecture

4.1. Testing on PFL Benchmarks based on Dirichlet Allocation

Recently, a work of [3] suggests a PFL benchmark with CIFAR-10 by imposing heterogeneity through Dirichlet allocation. We borrow the same settings in [3]: CIFAR10-$\alpha 0.1$, CIFAR10-$\alpha 0.5$, and CIFAR10-$\alpha 5$ cases are considered with 100 clients. In each round, 20 active clients participate in the federation. For the model architecture, a convolutional neural network with two 5x5 convolutional layers followed by a max-pooling layer, batch normalization, ReLU, and two dense layers. In Table 7, test accuracies on three CIFAR-10-based benchmarks are shown. The accuracies of prior works are borrowed from the evaluation results in [3]. We show the simplest form of each algorithm without any fine-tuning. As aforementioned, `MetaVers` also does not adopt the fine-tuning steps in testing. `MetaVers` shows the outstanding performance on CIFAR10-$\alpha 0.1$, which imposes the strongest heterogeneity across clients. When the data distri-

Table 6. Comparison with Centralized Learning on CIFAR-10, CIFAR-100, and CINIC-10.

	CIFAR-10			CIFAR-100			CINIC-10		
# clients	50	100	500	50	100	500	50	100	500
Centralized	90.9 ± 0.4	90.7 ± 0.3	90.7 ± 0.1	67.1 ± 0.1	66.5 ± 0.1	61.0 ± 0.3	73.1 ± 0.5	73.5 ± 0.2	72.9 ± 0.2
MetaVers (Decentralized)	90.8 ± 0.3	90.2 ± 0.2	89.9 ± 0.3	66.7 ± 0.0	64.8 ± 0.1	55.8 ± 0.1	73.2 ± 0.4	73.2 ± 0.3	72.5 ± 0.1

bution becomes homogeneous by increasing α, the performance gain diminishes so that MetaVers shows worse performance than others. We emphasize that the meta-learning framework used by MetaVers is shown to be effective in heterogeneous cases where the task and data distribution diverges across different mini-batches. We conjecture that the nature of meta-learning is the reason for the performance trend across different α values.

4.2. Results for ResNet architecture

Standard PFL Benchmarks are based on the LeNet architecture, which is quite small than cutting-edge backbone architecture. We want to point out that many prior works for PFL are still based on quite small architectures. To show the scalability of MetaVers, we conduct experiments using ResNet18 [8], which is a popular large architecture. All experimental settings, including the number of communication rounds and the active clients, are the same as in the LeNet-based experiments. Table 8 is the result of the experiment. MetaVers shows the highest performance in most cases, as in the experiment for the LeNet architecture. We clearly confirm MetaVers is also effective in training larger backbone architectures. Also, we want to emphasize that MetaVers achieves the highest performance in every case in CINIC-10 with significant margins, which is consistent with the LeNet-based experiments.

5. Convergence Analysis of MetaVers

Notations: $\mathcal{E}_i$ represents a training episode at client i. $\theta^{(\tau)}$ indicates the global model parameter at the beginning of the round τ. $\theta_i^{(\tau)}$ is the locally updated model parameter after an episodic-training at client i in the round τ. $\mathcal{L}_i(\theta; \mathcal{E}_i)$ is the local loss value based on the model parameter θ and the given training episode $\mathcal{E}_i$ from client i. With these notations, let us introduce some definitions as follow.

Definition 1. (*Expectation of Local Loss and Gradients*)

$$\mathcal{L}_i(\theta^{(\tau)}) \triangleq \mathbb{E}_{\mathcal{E}_i \sim \mathcal{D}_i}\big[\mathcal{L}_i(\theta^{(\tau)}; \mathcal{E}_i)\big] \tag{1}$$

$$\nabla\mathcal{L}_i(\theta^{(\tau)}) \triangleq \mathbb{E}_{\mathcal{E}_i \sim \mathcal{D}_i}\big[\nabla\mathcal{L}_i(\theta^{(\tau)}; \mathcal{E}_i)\big] \tag{2}$$

Definition 2. (*Aggregated Gradient Across Clients*)

$$\nabla\mathcal{L}(\theta^{(\tau)}; \mathscr{E}) \triangleq \frac{1}{n}\sum_{i=1}^{n} \nabla\mathcal{L}_i(\theta^{(\tau)}; \mathcal{E}_i), \tag{3}$$

where $\mathscr{E} = \{\mathcal{E}_i\}_{i=1}^n$ is the set of training episodes of clients.

Definition 3. (Expectation of aggregated gradient)

$$\nabla\mathcal{L}(\theta^{(\tau)}) \triangleq \mathbb{E}_{\mathscr{E}}\big[\nabla\mathcal{L}(\theta^{(\tau)}; \mathscr{E})\big] = \frac{1}{n}\sum_{i=1}^{n} \nabla\mathcal{L}_i(\theta^{(\tau)}) \tag{4}$$

We use the following assumptions, which are similar to the settings introduced by the work of [19].

Assumption 1. *For any client $i \in [1, n]$, the loss function is L-Lipschitz smooth:*

$$\|\nabla\mathcal{L}_i(\theta) - \nabla\mathcal{L}_i(\phi)\| \leq L\|\theta - \phi\|. \tag{5}$$

Assumption 2. *The inner-product between local gradient with an episode $\mathcal{E}_i$ and the averaged gradient is bounded:*

$$\alpha\|\nabla\mathcal{L}(\theta^{(\tau)}; \mathscr{E})\|^2 \leq \nabla\mathcal{L}_i(\theta^{(\tau)}; \mathcal{E}_i) \cdot \nabla\mathcal{L}(\theta^{(\tau)}; \mathscr{E}) \tag{6}$$

where $\alpha \in \mathbb{R}$ and $\alpha \in (0, 1]$.

Assumption 3. *For any set of episodes $\mathscr{E}$, the variance of the global gradients is bounded:*

$$\mathbb{E}_{\mathscr{E}}\big[\|\nabla\mathcal{L}(\theta^{(\tau)}; \mathscr{E}) - \nabla\mathcal{L}(\theta^{(\tau)})\|^2\big] \leq \sigma^2. \tag{7}$$

Lemma 1. *For every client $i \in [1, n]$, the difference of local loss values at round $\tau + 1$ and τ is bounded:*

$$\mathcal{L}_i(\theta^{(\tau+1)}) - \mathcal{L}_i(\theta^{(\tau)}) \leq (-\eta\alpha + \frac{1}{2}L\eta^2)\Big(\|\nabla\mathcal{L}(\theta^{(\tau)})\|^2 + \sigma^2\Big), \tag{8}$$

where η is the learning rate of local update.

Theorem 1. (Convergence) *For any client $i \in [1, n]$ with a learning rate $\eta^* < \frac{2\alpha}{L}$, the local loss is a decreasing function in the number of rounds:*

$$\mathcal{L}_i(\theta^{(\tau+t)}) < \mathcal{L}_i(\theta^{(\tau)}). \tag{9}$$

Table 7. Test accuracy on CIFAR10-α0.1, CIFAR10-α0.5, and CIFAR10-α5 benchmarks from [3].

	CIFAR10-α0.1	CIFAR10-α0.5	CIFAR10-α5
FedAvg [14]	57.71	70.89	73.87
FedOpt [17]	37.30	49.61	54.66
pFedMe [5]	56.46	70.63	73.74
FedBN [10]	57.59	68.28	72.17
Ditto [9]	49.77	67.65	71.97
FedEM [12]	54.52	70.56	73.36
`MetaVers` (Ours)	**81.41**	**73.06**	**67.67**

Table 8. Test accuracy ($\pm$ SEM) over 50, 100, 500 clients on CIFAR-10, CIFAR-100, and CINIC-10 using ResNet18.

	CIFAR-10			CIFAR-100			CINIC-10		
# clients	50	100	500	50	100	500	50	100	500
# samples/client	800	400	80	800	400	80	1800	900	180
Local	83.7 ± 0.6	82.7 ± 1.0	83.0 ± 0.5	50.2 ± 0.3	45.7 ± 0.5	31.4 ± 0.7	60.7 ± 1.4	60.4 ± 1.3	60.2 ± 0.4
`FedAvg` [14]	32.4 ± 1.8	31.6 ± 0.7	32.0 ± 1.6	35.6 ± 0.1	34.9 ± 0.1	28.6 ± 0.3	47.7 ± 1.0	47.5 ± 0.3	45.7 ± 0.2
FedProto [19]	83.9 ± 0.3	76.1 ± 1.1	54.1 ± 0.0	47.4 ± 1.0	21.9 ± 0.4	11.5 ± 0.2	57.6 ± 0.5	46.7 ± 0.3	27.5 ± 0.2
Per-FedAvg [6]	54.7 ± 2.8	55.3 ± 0.4	63.4 ± 0.7	32.5 ± 0.1	28.3 ± 0.1	27.7 ± 0.1	46.7 ± 0.3	47.5 ± 0.1	45.9 ± 0.0
pFedGP [1]	84.2 ± 0.6	84.7 ± 0.4	$\mathbf{84.8 \pm 0.4}$	54.3 ± 0.3	52.2 ± 0.1	$\mathbf{45.7 \pm 0.4}$	63.2 ± 0.0	64.9 ± 0.5	67.7 ± 0.1
FedRep [4]	79.4 ± 1.2	76.5 ± 1.1	65.9 ± 1.6	59.8 ± 0.8	57.6 ± 0.3	34.7 ± 0.7	67.7 ± 0.0	66.8 ± 0.4	64.6 ± 0.6
`MetaVers`	$\mathbf{86.7 \pm 0.5}$	$\mathbf{85.3 \pm 0.3}$	$\mathbf{84.7 \pm 0.2}$	$\mathbf{60.8 \pm 1.0}$	$\mathbf{57.9 \pm 0.2}$	40.8 ± 0.5	$\mathbf{70.1 \pm 0.6}$	$\mathbf{70.7 \pm 0.3}$	$\mathbf{68.4 \pm 0.6}$

5.1. Proof of Lemma 1

Assumption 1 implies the following inequality.

$$\mathcal{L}_i(\theta^{(\tau+1)};\mathcal{E}_i) - \mathcal{L}_i(\theta^{(\tau)};\mathcal{E}_i) \leq \nabla\mathcal{L}_i(\theta^{(\tau)};\mathcal{E}_i)\cdot(\theta^{(\tau+1)}-\theta^{(\tau)}) + \frac{1}{2}L\|\theta^{(\tau+1)}-\theta^{(\tau)}\|^2. \tag{10}$$

By using the fact that

$$\begin{aligned}\theta^{(\tau+1)} &= \theta^{(\tau)} - \frac{1}{n}\eta\sum_{j=1}^{n}\nabla\mathcal{L}_i(\theta^{(\tau)};\mathcal{E}_i)\\ &= \theta^{(\tau)} - \eta\nabla\mathcal{L}(\theta^{(\tau)};\mathscr{E}),\end{aligned} \tag{11}$$

the inequality (10) becomes

$$\mathcal{L}_i(\theta^{(\tau+1)};\mathcal{E}_i) - \mathcal{L}_i(\theta^{(\tau)};\mathcal{E}_i) \leq \nabla\mathcal{L}_i(\theta^{(\tau)};\mathcal{E}_i)\cdot(-\eta\nabla\mathcal{L}(\theta^{(\tau)};\mathscr{E})) + \frac{1}{2}L\|-\eta\nabla\mathcal{L}(\theta^{(\tau)};\mathscr{E})\|^2. \tag{12}$$

By following Assumption 4, we have

$$\begin{aligned}&\mathcal{L}_i(\theta^{(\tau+1)};\mathcal{E}_i) - \mathcal{L}_i(\theta^{(\tau)};\mathcal{E}_i) && (13)\\ &\leq \nabla\mathcal{L}_i(\theta^{(\tau)};\mathcal{E}_i)\cdot(-\eta\nabla\mathcal{L}(\theta^{(\tau)};\mathscr{E}))\\ &+ \frac{1}{2}L\|-\eta\nabla\mathcal{L}(\theta^{(\tau)};\mathscr{E})\|^2 && (14)\\ &= -\eta\alpha\|\nabla\mathcal{L}(\theta^{(\tau)};\mathscr{E})\|^2 + \frac{1}{2}L\eta^2\|\nabla\mathcal{L}(\theta^{(\tau)};\mathscr{E})\|^2 && (15)\\ &= (-\eta\alpha + \frac{1}{2}L\eta^2)\|\nabla\mathcal{L}(\theta^{(\tau)};\mathscr{E})\|^2. && (16)\end{aligned}$$

Let us take the expectation over the given episodes $\mathscr{E} = \{\mathcal{E}_j\}_{j=1}^n$, then we have

$$\begin{aligned}&\mathbb{E}_{\mathscr{E}}\Big[\mathcal{L}_i(\theta_i^{(\tau+1)};\mathcal{E}_i)\Big] - \mathbb{E}_{\mathscr{E}}\Big[\mathcal{L}_i(\theta^{(\tau)};\mathcal{E}_i)\Big] && (17)\\ &\overset{(a)}{=} \mathcal{L}_i(\theta^{(\tau+1)}) - \mathcal{L}_i(\theta^{(\tau)}) && (18)\\ &\leq (-\eta\alpha + \frac{1}{2}L\eta^2)\mathbb{E}_{\mathscr{E}}\Big[\|\nabla\mathcal{L}(\theta^{(\tau)};\mathscr{E})\|^2\Big] && (19)\\ &\overset{(b)}{\leq} (-\eta\alpha + \frac{1}{2}L\eta^2)\Big(\|\nabla\mathcal{L}(\theta^{(\tau)})\|^2\\ &+ \mathbb{E}_{\mathscr{E}}\big[\|\nabla\mathcal{L}(\theta^{(\tau)};\mathscr{E}) - \nabla\mathcal{L}(\theta^{(\tau)})\|^2\big]\Big) && (20)\\ &\overset{(c)}{\leq} (-\eta\alpha + \frac{1}{2}L\eta^2)\Big(\|\nabla\mathcal{L}(\theta^{(\tau)})\|^2 + \sigma^2\Big). && (21)\end{aligned}$$

The equality (a) follows Definition 1. The inequality (b) is base on Definition 1 and the fact that $\mathbb{E}[\|X\|^2] \leq \|\mathbb{E}[X]\|^2 + \mathbb{E}[\|X - \mathbb{E}[X]\|^2]$. Also, the inequality (c) follows Assumption 3.

5.2. Proof of Theorem 1

Based on Lemma 1, when the model is updated after a single communication round, the loss at any local client decreases. Then for any $\tau, t > 0$,

$$\mathcal{L}_i(\theta^{(\tau+t)}) - \mathcal{L}_i(\theta^{(\tau)})$$
$$= \sum_{j=1}^{t} \left(\mathcal{L}_i(\theta^{(\tau+j)}) - \mathcal{L}_i(\theta^{(\tau+j-1)})\right) \tag{22}$$
$$\overset{(d)}{\leq} \sum_{j=1}^{t} \Big((-\eta\alpha + \frac{1}{2}L\eta^2)\big(\|\nabla\mathcal{L}(\theta^{(\tau)})\|^2 + \sigma^2\big)\Big) \tag{23}$$
$$= (-\eta\alpha + \frac{1}{2}L\eta^2)\sum_{j=1}^{t}\big(\|\nabla\mathcal{L}(\theta^{(\tau)})\|^2 + \sigma^2\big). \tag{24}$$

With a learning rate $\eta^* < \frac{2\alpha}{L}$, then the loss function at any local client is a decreasing function:

$$\mathcal{L}_i(\theta^{(\tau+t)}) - \mathcal{L}_i(\theta^{(\tau)})$$
$$= (-\eta^*\alpha + \frac{1}{2}L\eta^2)\sum_{j=1}^{t}\big(\|\nabla\mathcal{L}(\theta^{(\tau)})\|^2 + \sigma^2\big) < 0. \tag{25}$$

References

[1] Idan Achituve, Aviv Shamsian, Aviv Navon, Gal Chechik, and Ethan Fetaya. Personalized federated learning with gaussian processes. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 8392–8406, 2021. 1, 2, 6

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

[3] Daoyuan Chen, Dawei Gao, Weirui Kuang, Yaliang Li, and Bolin Ding. pfl-bench: A comprehensive benchmark for personalized federated learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 9344–9360, 2022. 4, 6

[4] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. In *International Conference on Machine Learning (ICML)*, pages 2089–2099. PMLR, 2021. 1, 2, 6

[5] Canh T. Dinh, Nguyen H. Tran, and Tuan Dung Nguyen. Personalized federated learning with moreau envelopes. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 21394–21405, 2020. 2, 6

[6] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 3557–3568, 2020. 6

[7] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning (ICML)*, pages 1126–1135. PMLR, 2017. 1, 2

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 5

[9] Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through personalization. In *International Conference on Machine Learning (ICML)*, volume 139, pages 6357–6368. PMLR, 2021. 6

[10] Xiaoxiao Li, Xiaofei Zhang, Michael Kamp, and Qi Dou. Fedbn: Federated learning on non-iid features via local batch normalization. In *International Conference on Learning Representations (ICLR)*, 2021. 6

[11] 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. 2

[12] Othmane Marfoq, Giovanni Neglia, Aurélien Bellet, Laetitia Kameni, and Richard Vida. Federated multi-task learning under a mixture of distributions. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 6

[13] Othmane Marfoq, Giovanni Neglia, Laetita Kameni, and Richard Vida. Personalized federated learning through local memorization. In *International Conference on Machine Learning (ICML)*, pages 15070–15092. PMLR, 2022. 1, 2

[14] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017. 2, 6

[15] Jaehoon Oh, Sangmook Kim, and Se-Young Yun. Fedbabu: Towards enhanced representation for rederated image classification. In *International Conference on Learning Representations (ICLR)*, 2022. 2, 3

[16] Younghyun Park, Dong-Jun Han, Do-Yeon Kim amd Jun Seo, and Jaekyun Moon. Few-round learning for

federated learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 28612–28622, 2021. 2, 3

[17] Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and H Brendan McMahan. Adaptive federated optimization. In *International Conference on Learning Representations (ICLR)*, 2020. 6

[18] Aviv Shamsian, Aviv Navon, Ethan Fetaya, and Gal Chechik. Personalized federated learning using hypernetworks. In *International Conference on Machine Learning (ICML)*, pages 9489–9502. PMLR, 2021. 1, 2

[19] Yue Tan, Guodong Long, Lu Liu, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. Fedproto: Federated prototype learning over heterogeneous devices. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 36, pages 8432–8440, 2022. 1, 2, 5, 6