PERADA: Parameter-Efficient Federated Learning Personalization with Generalization Guarantees

Chulin Xie[†,‡], De-An Huang[♠], Wenda Chu[♡], Daguang Xu[♠],
Chaowei Xiao[♠,¶,*], Bo Li[†,§,*], Anima Anandkumar[♡,*]
[†]UIUC [♠]NVIDIA [♡]Caltech [¶]UW-Madison [§]UChicago

Abstract

Personalized Federated Learning (pFL) has emerged as a promising solution to tackle data heterogeneity across clients in FL. However, existing pFL methods either (1) introduce high computation and communication costs or (2) overfit to local data, which can be limited in scope and vulnerable to evolved test samples with natural distribution shifts. In this paper, we propose PERADA*, a parameter-efficient pFL framework that reduces communication and computational costs and exhibits superior generalization performance, especially under test-time distribution shifts.* PERADA *reduces the costs by leveraging the power of pretrained models and only updates and communicates a small number of additional parameters from adapters.* PERADA *achieves high generalization by regularizing each client's personalized adapter with a global adapter, while the global adapter uses knowledge distillation to aggregate generalized information from all clients. Theoretically, we provide generalization bounds of* PERADA*, and we prove its convergence to stationary points under non-convex settings. Empirically,* PERADA *demonstrates higher personalized performance (+4.85% on CheXpert) and enables better out-of-distribution generalization (+5.23% on CIFAR-10-C) on different datasets across natural and medical domains compared with baselines, while only updating 12.6% of parameters per model. Our code is available at https://github.com/NVlabs/PerAda.*

1. Introduction

Federated Learning (FL) allows clients to collaboratively train machine learning models without direct access to their data, especially for privacy-sensitive tasks [45]. FL was initially designed to train a single global model for all clients. However, such a one-model-fits-all paradigm is not effective when there is *client heterogeneity*, i.e., the local data are non-IID across clients with heterogeneous features or label distributions [35]. Personalized Federated Learning (pFL) [43] has emerged as an effective solution to tackle client heterogeneity. In pFL, each client trains a personalized model on its local data to ensure personalized performance, while leveraging the aggregated knowledge from other clients to improve its generalization.

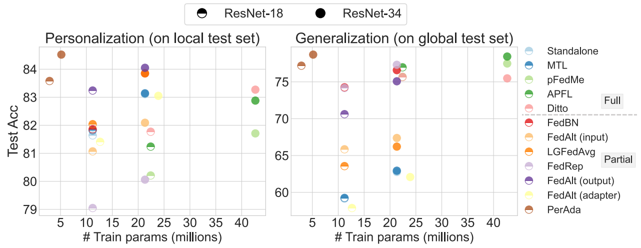


Figure 1. Accuracy of personalized models on Office-Home. "Full"/"Partial" denotes full/partial model personalization. PERADA achieves the highest personalized performance and generalization by updating the smallest number of model parameters.

Existing works in pFL commonly use *full model personalization*, where each client trains a personalized model as well as a copy of the global model from the server for regularization [33, 59]. However, these methods are parameter-expensive, leading to high computational and communicational costs, which is impractical for clients with limited computation resources and network bandwidth [26]. Later on, *partial model personalization* alleviates this issue by splitting each client's *one* model into personalized parameters and shared parameters, where only the set of shared parameters would be communicated with the server [48]. Nonetheless, these methods tend to overfit more to the local training samples since the set of shared parameters does not encode generalized knowledge well compared to a full global model. This hurts the performance of partially personalized models in real-world FL deployment, where the incoming local test samples are evolving with natural shifts from the local training distribution [25], e.g., images taken under varying weather or lighting conditions.

Our Approach. In this work, we propose PERADA, a pFL framework that *reduces communication and computation costs for clients while personalizing the model and maintaining its generalization to test-time distribution shifts*, as shown

‡ work done during an internship at NVIDIA; ∗ equal advising.

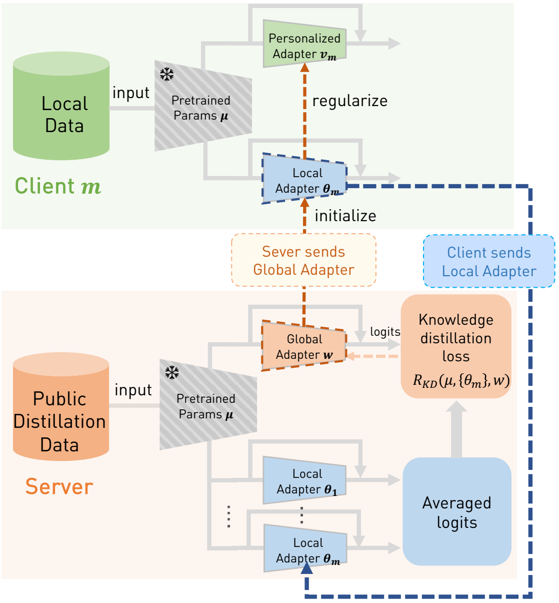


Figure 2. Illustration of PERADA.

in Figure 1. PERADA is a parameter-efficient **per**sonalized FL framework based on **Ada**pter [50] and Knowledge Distillation (KD) [20]. The overview is shown in Figure 2.

Each client has a pretrained model, a personalized adapter, and a local adapter, where each adapter consists of a small number of additional parameters planted in the pretrained model with skip connections. At each training round, *to reduce the computation and communication costs*, PERADA leverages the power of the pretrained model, and *only* updates the personalized adapter and the local adapter using local data, and sends the local adapter to the server. In this way, it limits the number of trainable parameters and only communicates the local adapter, instead of the full model.

Then, to *improve the generalization*, the server aggregates clients' local adapters (i.e., teachers) via knowledge distillation and trains the global adapter (i.e., student). Specifically, it uses the averaged logits from teachers on an unlabeled public distillation dataset as the pseudo-labels to train the student. This avoids directly averaging clients' models trained on heterogeneous local data, while enriching the global adapter with the ensemble knowledge from clients' models and mitigating the potential model aggregation drifts caused by heterogeneity. After that, the server sends the distilled global adapter back to the clients, which is used to initialize the local adapter and regularize the training of the personalized adapter to prevent overfitting and *improve the generalization*. During the testing phase, each client uses the personalized adapter for inference.

To explain why PERADA is effective in improving generalization, we theoretically derive its generalization bounds under FL covariate (or feature) shift non-IID setting [44]. We are the *first* to show that the generalization on a target distribution (e.g., potentially with test-time distribution shift) can be enhanced for both global model and personalized models by KD when the *distillation optimization error is small*, and the distribution of the unlabeled distillation dataset is *close* to the target distribution. We also characterize the role of different components in PERADA on generalization, such as client heterogeneity, pretrained model, and the prediction distance between the global and personalized models.

In addition, we establish convergence guarantees for PERADA in general non-convex settings. The analysis of PERADA is challenging due to the bi-level optimization between server distillation training and local client training. We establish the convergence rates for the global model and personalized models to stationary points and demonstrate the effects of KD and client heterogeneity on the convergence. As far as we know, these are the *first*-known results for FL convergence under *server distillation*.

Empirically, we conduct extensive evaluations on different datasets, including natural and medical images (CIFAR-10, Office-Home, and CheXpert) under both FL covariate-shift and label-shift non-IID settings. We show that PERADA achieves competitive personalized accuracy over state-of-the-art pFL methods with only 12.6% of trainable parameters while obtaining higher generalization, especially when evaluated on out-of-distribution data. We further show that the benefits of PERADA extend to differentially private (DP) FL settings and improve the DP-utility trade-offs compared to full model personalization. In summary,

- We propose PERADA, a lightweight pFL framework with personalized adapters that provides personalization while reducing computation/communication costs. We improve the generalization of PERADA with server-side KD.
- We theoretically analyze the effectiveness of PERADA, and prove the generalization bounds and the convergence rates for both the global model and personalized models under non-convex settings.
- Through extensive experiments, we show that PERADA achieves higher personalized performance and better generalization than state-of-the-art pFL methods with smaller computation and communication costs. Moreover, PERADA retains its benefits under differential privacy.

2. Related Work

Full Model Personalization. Many pFL approaches require each client to train a personalized model and a global model, where the global model is used to prevent the personalized model from overfitting. It includes methods based on meta learning [12], model mixture [10, 16, 43], global regaurlization [33], mean regularization [16, 17, 59] and clustering [15, 54]. However, these methods induce high costs by training two full models in each client and communicating the full model. Another approach is to locally finetune an FL global model (e.g., from FEDAVG [45]). While local finetuning yields promising personalized accuracy [8, 62, 65], it could be prone to catastrophic forgetting and overfitting to its (limited) local data, sacrificing the generalizability [25, 49].

Partial Model Personalization trains one model for each client to reduce the costs, which is partitioned into shared parameters and personalized parameters, such as personalized feature extractors [9], prediction head [3, 7, 38], batch normalization [36], adapters [48], and adaptively selected parameters [58]. Nevertheless, the shared parameters do not learn generalized information well compared to a full global model, so the partially personalized models can have inferior generalization ability. To further reduce the costs, Shysheya et al. [56] apply parameter-efficient transfer learning techniques to train FedAvg and perform local finetuning. However, it does not specifically address the generalization issues of personalization, which is the focus of our work.

Knowledge Distillation (KD) in FL. KD is a technique that transfers the knowledge from one or multiple teacher models to a student model [20]. *Ensemble distillation* has been used to tackle data heterogeneity in generic FL, by refining the *server* model with ensemble knowledge from clients, rather than directly aggregating their model parameters. Specifically, the ensemble predictions from clients' models on an unlabeled dataset are used to guide the training of the server model, where the unlabeled dataset can be public data [6, 31, 39] or generated data [67]. Another line of work leverages *client*-side *local distillation* to transfer global knowledge to local models in generic FL [29, 68] or personalized models in pFL [46, 66]. To reduce the load for clients, we focus on parameter-efficient ensemble distillation in the server with public data to train a better global model, and study its effects on personalized models with novel convergence guarantees and generalization bounds.

Parameter-efficient fine-tuning techniques applied to pretrained large models [5] have become the prominent practice in transfer learning to save computation costs [14, 30, 40]. Motivated by the success of Adapter, a low-cost plug-in mounted on pre-trained vision models [50] or large language models [21, 37, 41], we investigate Adapter in the context of parameter-efficient personalization. Instead of training both the backbone and adapter for pFL as in [48], we treat the adapter parameters as personal and the rest of the model parameters as frozen, and further leverage sever-side ensemble distillation to improve pFL performance.

3. Preliminaries and Challenges

We consider a typical setting of FL with M clients where each client m has a training dataset $\mathbb{D}_m = \{(x_{m,j}, y_{m,j}), j \in [n_m]\}$ with n_m data samples dawn from its local distribution μ_m. Let $f(W, x)$ represents a model that outputs the logit vector given input x, where $W \in \mathbb{R}^d$, denotes its model parameters. Let the loss function be $\ell(f(W, x), y)$, and the empirical loss on local data $\mathbb{D}_m$ associated with client m be $\mathcal{L}_m(W) := \frac{1}{n_m}\sum_{j=1}^{n_m} \ell\left(f\left(W, x_{m,j}\right), y_{m,j}\right)$.

Generic FL aims to optimize a single global model with all clients' local data with the FL objective: $\min_W \mathcal{L}(W)$ where $\mathcal{L}(W) := \frac{1}{M}\sum_{m=1}^{M} \mathcal{L}_m(W)$. A standard way to solve it is FedAvg, which iterates between local model training and global model aggregation for multiple communication rounds. However, due to the heterogeneous local data distributions among clients, local model would drift away from each other, making the aggregated global model deviate from the optimal solution.

Personalized FL learns a personalized model for each client to perform well on its local data while preventing overfitting by leveraging the knowledge from other clients. However, achieving the goal is non-trivial due to the following challenges: (1) **High costs**: existing full model personalization studies [12, 16, 33, 59], which optimize $\min_{W, \{V_m\}} \frac{1}{M}\sum_{m=1}^{M}(\mathcal{L}_m(V_m) + \frac{\lambda}{2}\|V_m - W\|^2)$, require *twice* the memory footprint of the full model at each client by locally updating personalized model $V_m \in \mathbb{R}^d$ and global model $W \in \mathbb{R}^d$ where λ is the ℓ_2 regularization weight controlling the extent of personalization. (2) **Limited generalization**: partial model personalization [7, 9, 38, 48] is more efficient by training a full model $V_m = (u, v_m)$ at each client and communicating a subset of parameters, where $u \in \mathbb{R}^{d_u}$ are shared parameters and $v_m \in \mathbb{R}^{d_v}$ are personal parameters: $\min_{u, \{v_m\}} \frac{1}{M}\sum_{m=1}^{M} \mathcal{L}_m(u, v_m)$. However, such a partially personalized model can be *dominated by personal knowledge* with v_m and *poor at encoding generalized knowledge* with the remaining u from global distribution, leading to inferior performance under test-time distribution shifts. Figure 3 depicts such challenges in existing studies.

4. Method

Here we introduce the objectives and algorithm for PerAda.

Personalized and Global Objectives of PerAda. We address the challenges discussed in Sec. 3 by proposing PerAda, which improves the efficiency of learning personalized adapters and enhances their generalization with regularization and KD. Specifically, we (1) train the personalized adapter $\{v_m\}$ regularized towards a global adapter w to optimize a personalized objective (Personal Obj), and (2) train a well-generalized w via KD to optimize a global objective (Global Obj) under non-IID data, where we use the *alternative* optimization between client local training of local adapter $\{\theta_m\}$ and server KD training of w.

Concretely, we improve the efficiency of partial model personalization with a pretrained model and personalized adapters. Here the personalized adapter consists of a small number of additional parameters with skip connections (in Figure 2), which can reduce to the identity function when its parameters are zero [50, 66]. Our personalized adapter is trained with regularization to prevent overfitting, yielding the personal objective of each client m:

$$\min_{v_m} P_m(v_m, w) := \mathcal{L}_m(u, v_m) + \frac{\lambda}{2}\|v_m - w\|^2, \tag{Personal Obj}$$

where $u \in \mathbb{R}^{d_u}$ denotes the fixed pretrained parameters, and $v_m, w \in \mathbb{R}^{d_a}$ are **personalized adapter** and **global adapter**, respectively, with $d_a \ll d_u$.

Since the global adapter w is trained with all client data, regularizing v_m with w could potentially boost v_m's generalization power. Thus, enhancing w's generalization capacity is crucial for training a personalized model that demonstrates robust generalization as well. Instead of using FEDAVG [45] to learn w as in regularization-based pFL method [33], we leverage server-side ensemble distillation [39] to enrich the global adapter with ensemble knowledge from clients' models and alleviate model aggregation drifts induced by client heterogeneity, yielding the global objective:

$$\min_{w} \mathcal{R}_{\mathtt{KD}}(u, \{\theta_m\}_{m=1}^{M}, w) \qquad \text{(Global Obj)}$$

$$\text{where} \quad \theta_m = \arg\min_{\theta} \mathcal{L}_m(u, \theta), \text{initialized with } w.$$

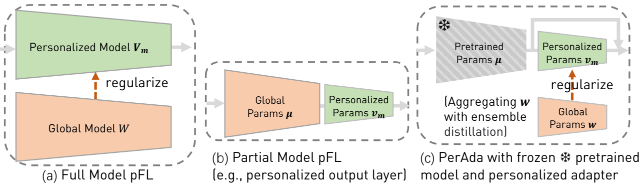
Here $\theta_m \in \mathbb{R}^{d_a}$ is client m's **locally updated global adapter**, and we call it as **local adapter** for distinguishment. The KD loss is defined as: $\mathcal{R}_{\mathtt{KD}}(u, \{\theta_m\}_{m=1}^{M}, w) := \sum_{j=1}^{n_{\mathtt{aux}}} \ell_{\mathtt{KD}}(\sum_{m=1}^{M} \frac{f((u,\theta_m),x_j)}{M}, f((u,w),x_j))$, which is the average distillation loss (between the averaged logits of local models and logits of the global model) on an auxiliary (unlabeled) dataset $\mathbb{D}_{\mathtt{aux}} = \{x_j\}_{j=1}^{n_{\mathtt{aux}}}$ drawn from the distribution $\mu_{\mathtt{aux}}$. Here $\ell_{\mathtt{KD}}(a,b) = \mathtt{KL}(\sigma(a), \sigma(b))$ is Kullback-Leibler divergence loss where σ is softmax function [20]. Compared to server-side KD in generic FL [6, 39, 67], we only update adapters instead of full models, which is more efficient for training and communication.

Algorithm 1 PERADA with client and server training

```
1:  Input: $M$ clients, pretrained model parameters $u$, initialized adapters $w^0$, $\{v_m^0\}$, local datasets $\{\mathbb{D}_m\}$, an unlabeled dataset $\mathbb{D}_{\mathtt{aux}}$
2:  Output: Personalized adapters $v_1^T, \dots, v_M^T$
3:  for communication round $t \in [T]$ do
4:      $\mathcal{S}_t \leftarrow$ Server samples $C$ clients from $M$ clients
5:      Server sends global adapter $w^t$ to the selected clients
6:      for client $m \in \mathcal{S}_t$ do
7:          Client initializes personalized adapter $v_m^{t,0}$ as $v_m^t$
8:          for step $s \in [S]$ do
9:              // update personalized adapter
10:             $v_m^{t,s+1} \leftarrow v_m^{t,s} - \eta_p \left(\widetilde{\nabla}\mathcal{L}_m\left(u, v_m^{t,s}\right) + \lambda\left(v_m^{t,s} - w^t\right)\right)$
11:         Client sets $v_m^{t+1} \leftarrow v_m^{t,S}$
12:         Client initializes local adapter $\theta_m^{t,0}$ as $w^t$
13:         for step $e \in [E]$ do
14:             // update local adapter
15:             $\theta_m^{t,e+1} \leftarrow \theta_m^{t,e} - \eta_l \widetilde{\nabla}\mathcal{L}_m(u, \theta_m^{t,e})$
16:         Client sends local adapter $\theta_m^{t+1} \leftarrow \theta_m^{t,E}$ to server

17:     Server initializes the global adapter $w^{t,0}$ by averaging
18:     $w^{t,0} \leftarrow \sum_{m \in \mathcal{S}_t} \frac{1}{|\mathcal{S}_t|} \theta_m^{t+1}$
19:     for step $r \in [R]$ do
20:         // update global adapter
21:         $w^{t,r+1} \leftarrow w^{t,r} - \eta_g \widetilde{\nabla}_w \mathcal{R}_{\mathtt{KD}}(u, \{\theta_m^{t+1}\}_{m \in \mathcal{S}^t}, w^{t,r})$
22:     Server sets $w^{t+1} \leftarrow w^{t,R}$
```

PERADA Algorithm. Now we introduce the details of iteratively optimizing the personalized objective and the global objective. Algorithm 1 presents our workflow. At each communication round $t \in [T]$, the server selects C clients $\mathcal{S}_t$ and broadcasts the current global adapter w^t. **To optimize personalized objective**, each selected client $m \in \mathcal{S}_t$ initializes personalized adapter as $v_m^{t,0} \leftarrow v_m^t$, and updates it for S steps with learning rate η_p and mini-batches $\{\xi_m^{t,s}\}_{s=0}^{S-1}$ sampled from $\mathbb{D}_m$ (Line 10). The client sets personalized adapter $v_m^{t+1} \leftarrow v_m^{t,S}$ after training. **To optimize global objective**, each selected client m initializes local adapter as the received global adapter $\theta_m^{t,0} \leftarrow w^t$, and makes local updates for E steps with learning rate η_l and mini-batches $\{\xi_m^{t,e}\}_{e=0}^{E-1}$ sampled from $\mathbb{D}_m$ (Line 15). Then client m sends the updated local adapter $\theta_m^{t+1} \leftarrow \theta_m^{t,E}$ to server. After receiving local adapters, the server first initializes the global adapter by parameter-averaging $w^{t,0} \leftarrow \bar{\theta}_m^{t+1}$ where $\bar{\theta}_m^{t+1} := \sum_{m \in \mathcal{S}_t} \frac{1}{|\mathcal{S}_t|} \theta_m^{t+1}$. Then, the server updates global adapter for R steps via knowledge distillation from local adapters (Line 21) with learning rate η_g and batches $\{\xi^{t,r}\}_{r=1}^{R}$ sampled from $\mathbb{D}_{\mathtt{aux}}$. The server will send the updated global adapter as $w^{t+1} \leftarrow w^{t,R}$ to clients at the next communication round.



(a) Full Model pFL

(b) Partial Model pFL (e.g., personalized output layer)

(c) PerAda with frozen ❄ pretrained model and personalized adapter

Figure 3. Current full model personalization incurs high computation costs by training two models, whereas existing partial model personalization often falls short in terms of generalizability. By updating adapter only, PERADA achieves a favorable balance between training/communication costs of clients and their pFL performance.

5. Generalization Bounds of PERADA

In this section, we analyze the generalization bounds for PERADA by answering the questions: *how do the distillation data distribution and KD optimization impact the generalization of the global model? How does the global model impact the generalization of personalized models?*

For notation simplicity, we define $p_1, \cdots, p_M$ as the personalized hypothesis, where each hypothesis $p_m \in \mathcal{P}_m : \mathcal{X} \to [0,1]^k$ maps the input $x \in \mathcal{X}$ to a *probability vector* over the k classes (i.e., softmax outputs). Similarly, we define global hypothesis $g \in \mathcal{G}$ and local hypothesis $h_m(x) \in \mathcal{H}_m, \forall m \in [M]$. We call "hypothesis" as "model" in this section. The local dataset $\mathbb{D}_m$ of each client m is drawn from the local distribution μ_m, and the distillation dataset $\mathbb{D}_{\mathtt{aux}}$ of the server is drawn $\mu_{\mathtt{aux}}$. We study generalization of the global model and personalized models on a **target distribution** μ **of interest** (e.g., with distribution shifts), by analyzing the effect of local distributions $\{\mu_m\}$ and distillation distribution $\mu_{\mathtt{aux}}$ used in FL training. We focus on the generalization bounds under FL covariate shifts following [44] and defer all proofs to Appendix C.

Global Model. Previous KD-based FL generalization bounds [39, 68] simply assume a perfect distillation (i.e., the global model is the ensemble of local models) *which neglects the actual distillation errors and the choice of distillation distribution.* To take them into account, we define the *ensemble distillation distance* on n_{aux} points $\{x_i\}_{i=1}^{n_{\text{aux}}}$ drawn from μ_{aux} as: $\Phi_{\mu_{\text{aux}},n_{\text{aux}}}(h_1,\dots,h_M;g) := \frac{1}{n_{\text{aux}}}\sum_{i=1}^{n_{\text{aux}}}\|g(x_i) - \frac{1}{M}\sum_{m=1}^{M} h_m(x_i)\|_1$ which measures the output difference between the global model and the ensemble of local models. To show g can have good generalization bounds on μ with KD, our main idea is to bound error probabilities of g with the expected distillation distances and errors of local models, and then bound the errors on μ by μ_m based on prior arts from domain adaptation [4]. We defer the preliminaries about learning theory to Appendix C.3.

Theorem 1 (Generalization bound of PERADA global model)**.** *Consider empirical datasets* $\mathbb{D} \sim \mu, \mathbb{D}_{\text{aux}} \sim \mu_{\text{aux}}, \mathbb{D}_m \sim \mu_m$ *with* $|\mathbb{D}| = |\mathbb{D}_m| = n, |\mathbb{D}_{\text{aux}}| = n_{\text{aux}}$. *Let* d_m *be the VC dimension of* $\mathcal{H}_m$, $\text{Rad}_{n_{\text{aux}}}$ *be the empirical Rademacher complexity measured on* n_{aux} *samples. With probability at least* $1-\delta$, *for every* $h_m \in \mathcal{H}_m, \forall m \in [M]$ *and* $g \in \mathcal{G}$, *we have* $\Pr_{(x,y)\sim\mu}\left[\underset{y'}{\arg\max} g(x)_{y'} \neq y\right] \leq 2\mathbb{E}_{(x,y)\sim\mu}[1 - g(x)_y] \leq \mathcal{O}(k^{3/2}[\max_j(\frac{1}{M}\sum_{m=1}^{M}\text{Rad}_{n_{\text{aux}}}(\mathcal{H}_m|_j)) + \max_j \text{Rad}_{n_{\text{aux}}}(\mathcal{G}|_j)]) + \frac{6}{M}\sum_{m=1}^{M}(\frac{4}{3}\sqrt{\frac{2d_m\log(2n)+\log(6M/\delta)}{n}} + \sqrt{\frac{\log(6M/\delta)}{2n}} + \sqrt{\frac{\log(6/\delta)}{2n_{\text{aux}}}} + \mathcal{O}(\text{Rad}_n(\mathcal{H}_m))) + \frac{1}{M}\sum_{m=1}^{M}(\underbrace{2\texttt{ERR}(\mathbb{D}_m,h_m)}_{\textit{local empirical risk}} + \underbrace{\hat{d}_{\mathcal{H}\Delta\mathcal{H}}(\mathbb{D}_m,\mathbb{D})}_{\textit{client heterogeneity}} + \lambda_m) + 2\underbrace{\Phi_{\mu_{\text{aux}},n_{\text{aux}}}(h_1,\dots,h_M;g)}_{\textit{ensemble distillation distance}} + 4\underbrace{\mathbb{TV}(\mu,\mu_{\text{aux}})}_{\textit{TV divergence}}$, *where* $\texttt{ERR}(\mathbb{D}_m,h_m) = \frac{1}{n}\sum_{j=1}^{n}\left[1 - h_m(x_{m,j})_{y_{m,j}}\right], \lambda_m = \varepsilon_{\mu_m}(h^*) + \varepsilon_\mu(h^*), h^* := \arg\min_{h\in\mathcal{H}}\varepsilon_{\mu_m}(h) + \varepsilon_\mu(h)$.

Remark 1. We discuss key implications of Theorem 1: (1) **Ensemble distillation.** $\Phi_{\mu_{\text{aux}},n_{\text{aux}}}$ captures the distillation error measured on the distillation dataset $\mathbb{D}_{\text{aux}}$ as minimized in Line 21. When $\mu_{\text{aux}} = \mu$, e.g., using data from the target distribution as the distillation dataset, KD improves the generalization of g during training by directly minimizing $\Phi_{\mu_{\text{aux}},n_{\text{aux}}}$. The smaller the distillation distance, the better the generalization. When $\mu_{\text{aux}} \neq \mu$, KD on μ_{aux} decreases $\Phi_{\mu_{\text{aux}},n_{\text{aux}}}$ while causing additional generalization gap measured by TV divergence $\mathbb{TV}(\mu_{\text{aux}},\mu)$. Compared to without KD, using a distillation dataset from a domain close to μ with small $\mathbb{TV}(\mu_{\text{aux}},\mu)$ and reducing $\Phi_{\mu_{\text{aux}},n_{\text{aux}}}$ during KD can also improve the generalization (e.g., when $\Phi_{\mu_{\text{aux}},n_{\text{aux}}} + 2\mathbb{TV}(\mu_{\text{aux}},\mu) \leq \Phi_{\mu,n_{\text{aux}}}$). We empirically verify the effect of different distillation datasets in Sec. 7.1. (2) **Quality of local models.** The $\texttt{ERR}(\mathbb{D}_m,h_m)$ term shows that reducing the empirical risk of local models w.r.t local distributions μ_m improves the generalization of the global model. We verify in Sec. 7.1 that a more powerful pretrained model, which results in higher quality local models, leads to better generalization. (3) **Sample complexity.** More empirical samples during training improve the generalization. We further discuss the effect of *client heterogeneity* $\hat{d}_{\mathcal{H}\Delta\mathcal{H}}(\mathbb{D}_m,\mathbb{D})$ (i.e., the empirical $\mathcal{H}$ -divergence between two datasets) and *number of classes* k in Appendix C.1.

Personalized Models. We show that personalized model p_m can generalize well on μ if global model g generalizes well on μ and p_m has small prediction distance with g.

Theorem 2 (Generalization bound of PERADA personalized model)**.** *With probability at least* $1-\delta$, *for every* $p_m \in \mathcal{P}_m, \forall m \in [M]$, *and for every* $g \in \mathcal{G}$, *we have* $\Pr_{(x,y)\sim\mu}\left[\underset{y'}{\arg\max} p_m(x)_{y'} \neq y\right] \leq 2\mathbb{E}_{(x,y)\sim\mu}(1 - g(x)_y) + 2\frac{1}{n}\sum_{i=1}^{n}\min\{1, \|p_m(x) - g(x)\|_1\} + 6\sqrt{\frac{\log(2/\delta)}{2n}} + \mathcal{O}\left(k^{3/2}\left[\max_j \text{Rad}_n(\mathcal{P}|_j) + \max_j \text{Rad}_n(\mathcal{G}|_j)\right]\right)$.

Remark 2. The first term is the population risk of g on μ, which has been upper bounded by Theorem 1. The second term is the prediction difference between g and personalized models. Therefore, the generalization of personalized model is intrinsically related to the performance of global model. In Sec. 7.1, we empirically show that moderately increasing the regularization strength λ in (Personal Obj) could improve the generalization of p_m, by reducing such prediction distance.

6. Convergence Guarantees of PERADA

In this section, we aim to provide the convergence analysis. We outline the analysis challenges for PERADA, arising from the bi-level optimization between server distillation and local training, as well as the personalization regularized by the global model. Then, we present the convergence analysis for PERADA global model and personalized model. For notation simplicity, we will omit the frozen parameters u and use $w/\theta_m/v_m$ to represent corresponding models.

To convey the salient ideas, we consider full client participation (i.e., $|\mathcal{S}_t| = M$) for convergence analysis following [46, 52]; thus, the stochasticity comes from mini-batch samplings during client and server training. Below, we first give several necessary assumptions.

Assumption 1. (Smoothness). $\mathcal{L}_m(\theta)$ is L-Lipschitz smooth $\forall m \in [M]$ and $\mathcal{R}(\{\theta_m\}, w)$ is L_R-Lipschitz smooth.

Assumption 2. (Bounded Variance). The stochastic gradients are unbiased and variance is bounded $\forall m \in [M]$: $\mathbb{E}\|\widetilde{\nabla}\mathcal{L}_m(\theta) - \nabla\mathcal{L}_m(\theta)\|^2 \leq \sigma^2$, $\mathbb{E}\|\widetilde{\nabla}_w\mathcal{R}(\{\theta_m\}, w) - \nabla_w\mathcal{R}(\{\theta_m\}, w)\|^2 \leq \sigma_R^2$.

Assumption 3. (Bounded Diversity). The variance of local gradients to global gradient is bounded $\frac{1}{M}\sum_{m=1}^{M}\|\nabla\mathcal{L}_m(w) - \frac{1}{M}\sum_{i=1}^{M}\nabla\mathcal{L}_i(w)\|^2 \leq \bar{\gamma}$.

Assumption 4. (Bounded Gradients). The functions $\mathcal{L}_m, \mathcal{R}, P_m, \forall m \in [M]$ have bounded gradients: $\|\nabla \mathcal{L}_m(\theta)\| \leq G$, $\|\nabla_w \mathcal{R}(\{\theta_m\}, w)\| \leq G_R$, $\|\nabla_w P_m(v_m, w)\| \leq G_P$.

We defer more discussions on the assumptions to Appendix D.1. Next, we discuss the challenges and present the main results. All proofs are relegated to Appendix D.

Global Model Convergence with Ensemble Distillation. Despite the wide applications of knowledge distillation in FL [29, 66, 68], its convergence analysis is less explored. To the best of our knowledge, there is no convergence guarantee under server-side ensemble distillation [6, 31, 39, 67]. This lack of research is likely because (1) the complexity of bi-level optimization between server distillation for w^t and client training for $\{\theta_m^t\}$, which incorporates two objectives (i.e., minimizing distillation loss and local loss respectively); (2) at each round, the global model is initialized by averaged local models before distillation, and local models are initialized by the global model before local training. Such mutual initializations intervene in the model updating trajectories of w^t and $\{\theta_m^t\}$ w.r.t their training objectives, making the convergence even harder to analyze. On the other hand, it has been empirically shown that ensemble distillation can improve the global model performance by incorporating diverse knowledge from clients (e.g., low $\mathcal{L}(w^t)$ measured on all clients' data) [6, 31, 39, 67]. Therefore, we aim to *understand the global model convergence w.r.t $\mathcal{L}(w^t)$ as a function of ensemble distillation*. To overcome the aforementioned challenges, we regard $\{\theta_m^t\}$ as the intermediate models to update w^{t+1}, and quantify the effects of local client training and server distillation on optimizing FL global objective:

Theorem 3 (Convergence of PerAda global model). *Let Assumptions 1 to 4 hold, and* $\eta_l = \frac{1}{EL\sqrt{T}}$, $\eta_g = \frac{1}{L_R RT}$, *denote* $\bar{w}^{t,e} = \frac{1}{M}\sum_{m=1}^{M} \theta_m^{t,e}$, *then the algorithm satisfies*

$$\sum_{t=0}^{T-1}\sum_{e=0}^{E-1} \frac{\mathbb{E}\|\nabla \mathcal{L}(\bar{w}^{t,e})\|^2}{ET} \leq \mathcal{O}\Big(\frac{L\Delta_{\mathcal{L}} + \psi_1}{\sqrt{T}} + \frac{\bar{\gamma}^2}{T} + \frac{L^2\psi_2}{T\sqrt{T}L_R^2 E}\Big),$$

where $\Delta_{\mathcal{L}} = \mathcal{L}(w^0) - \mathcal{L}(w^T)$, $\psi_1 = \frac{\sigma^2}{EM} + \frac{L(G^2+\psi_2)}{EL_R}$, *and* $\psi_2 = 4\sigma_R^2 + 32(3G_R^2 + \frac{2\sigma_R^2}{R})/T^2 + 2G_R^2$. *In particular,* $\bar{w}^{t+1,0} = w^t$ *and* $\bar{w}^{t+1,E-1} = \bar{\theta}^{t+1}$.

Remark 3. (1) **Convergence rate** is $\mathcal{O}(1/\sqrt{T})$ as it is the dominant term, matching the rate of the general FL non-convex settings of our interest [46, 59]. (2) **Local steps & distillation steps.** With more local updating steps E and distillation steps R, the terms ψ_1 and ψ_2 decrease. It means that a larger E and R can reduce the required communication rounds T to converge, thus lowering communication costs. (3) **Client heterogeneity** is reflected in $\bar{\gamma}$, whose effect can be mitigated by larger T. (4) **Ensemble distillation** is mainly reflected in ψ_2 where σ_R^2 are inherent data sampling noise when using stochastic gradients [12, 59], and G_R is from the bounded gradient assumption for distillation. The distillation gradient can be small when the averaged logits of local models (teacher) and the logits of the global model (student) are close (See Equation (11) and more discussion in Appendix D.1). Notably, the convergence bound remains valid for any distillation data, even if it is *out-of-domain*.

Personalized Model Convergence. Regarding personalization, unlike [59], to preserve generalization, the global model w^t of PerAda is not updated based on the personalized objective $P(v_m^t, w^t)$. Thus, it remains unclear *how the global model w^t learned from the ensemble distillation impacts the convergence of personalized models w.r.t $P(v_m^t, w^t)$*. In Theorem 4 (Appendix D.1), we analyze such impacts and show the convergence rate of personalized models.

7. Experiments

We empirically compare PerAda to existing pFL methods. We defer the details of experiments and hyperparameter as well as the additional experimental results to Appendix A.

Data and Model. We use CIFAR-10 [28], Office-Home [61], and medical image data CheXpert [24]. We simulate pFL setting for (1) *label Non-IID* using Dirichlet distribution $\text{Dir}(\alpha)$ [23] with $\alpha = 0.1/0.3$ on CIFAR-10/CheXpert, creating different local data size and label distributions for M clients; and (2) *feature Non-IID* on Office-Home by distributing the data from 4 domains (Art, Clipart, Product, and Real Word) to 4 clients respectively [58]. We use $M = 20$ for CIFAR-10/CheXpert, and sample 40% clients at every round following [7, 39], and use full client participation for Office-Home following [58]. We use ResNet-18 pretrained on ImageNet-1K [53] for all datasets. For PerAda[1], we use out-of-domain distillation dataset CIFAR-100 for CIFAR-10, and use CIFAR-10 for Office-Home/CheXpert.

Baselines. We evaluate full model pFL methods FedAvg+FT [65], Ditto [33], APFL [10], MTL [57], pFedMe [59], and partial model pFL methods with decoupled personalized/global parameters, including FedBN [36], LG-FedAvg [38], FedRep [9], FedSim [48], FedAlt [48]. We also include PerAda w/o KD, which is PerAda without Line 21 server-side knowledge distillation (i.e., using FedAvg to aggregate global adapter). Note that we use the *same pretrained ResNet as initialization* for all methods for fair comparisons.

Evaluation Metrics. We report the averaged test accuracy (**pFL accuracy**) and standard deviation over all clients' *personalized models*. For CheXpert, we report the AUC score since it is a multi-label classification task. We evaluate pFL accuracy mainly under two metrics: Local-test (i.e., clients' corresponding local test data) and Global-test (i.e., the union of clients' local test data), to study the *personalized performance* and *generalization* (against label or covariate shifts), respectively. In addition, for CIFAR-10, we evaluate pFL generalization against distribution shifts on

[1] We follow [48] to implement Adapter, which includes prediction head.

Table 1. Parameter-efficiency and averaged test accuracy across all clients' personalized models. PERADA achieves higher personalized performance and generalization with a smallest # of trainable parameters. **bold**/Underline fonts highlight the best/runner-up approach.

Algorithm	Personalized Params	# Trained Params	# Comm. Params	CIFAR-10				Office-Home		CheXpert	
				Local-test	Global-test	CIFAR-10.1	CIFAR-10-C	Local-test	Global-test	Local-test	Global-test
STANDALONE	Full model	11.18M	**0**M	85.94± 8.82	29.77± 8.09	25.82± 6.27	26.67 ± 7.07	81.64± 6.08	59.15± 3.32	65.06± 1.88	65.45± 2.3
MTL [57]	Full model	11.18M	11.18M	86.24± 8.45	29.46± 8.33	25.64± 6.42	26.4 ± 7.29	81.82± 5.53	59.25± 2.84	65.15± 1.95	65.48± 2.3
FEDAVG+FT [65]	Full model	11.18M	11.18M*	88.91± 5.71	43.99± 9.57	35.49± 8.02	36.51 ± 8.36	79.42± 5.62	77.19± 0.56	70.16± 0.78	70.6 ± 0.31
pFEDME [59]	Full model	22.36M	11.18M	90.73± 4.67	45.06± 8.65	36.51± 7.2	37.65 ± 7.6	80.21± 5.32	75.69± 0.69	65.07± 1.2	64.86± 1.22
APFL [10]	Full model	22.36M	11.18M	90.74± 4.75	43.92± 9.18	35.83± 7.5	36.51 ± 7.94	81.24± 4.51	76.98± 1.39	68.98± 1.04	68.96± 1.1
DITTO [33]	Full model	22.36M	11.18M	90.21± 4.61	53.82± 6.35	42.72± 5.68	44.32 ± 5.73	81.77± 4.31	75.66± 1.01	68.79± 1.4	68.86± 1.22
FEDBN [36]	Batch norm.	11.18M	11.17M	90.37± 5.19	43.18± 8.67	35.01± 7.24	36.29 ± 7.43	81.86± 5.13	74.26± 0.52	68.74± 1.17	68.83± 1.08
FEDALT [48]	Input layer	11.18M	6.45M	87.07± 6.54	32.23± 8.23	27.49± 6.41	28.51 ± 7.11	81.07± 5.59	65.85± 0.9	67.63± 1.18	67.74± 1.1
FEDSIM [48]	Input layer	11.18M	6.45M	87.93± 6.25	33.07± 8.16	28.21± 6.41	29.15 ± 7.16	82.45± 5.03	67.66± 0.82	67.49± 1.32	67.54± 1.24
LG-FEDAVG [38]	Feat. extractor	11.18M	0.005M	86.7 ± 8.01	29.96± 8	25.97± 6.21	26.83 ± 6.95	82.04± 5.96	63.57± 2.32	65.78± 1.62	66.23± 1.75
FEDREP [9]	Output layer	11.18M	11.17M	87.76± 6.46	35.19± 6.97	30.15± 5.89	30.68 ± 6.31	79.05± 5.88	74.17± 2.02	66.66± 1.82	66.52± 1.47
FEDALT [48]	Output layer	11.18M	11.17M	89.68± 5.4	40.68± 7.3	33.61± 6.12	34.3 ± 6.5	83.24± 3.96	70.62± 1.46	68.27± 1.3	68.36± 1.31
FEDSIM [48]	Output layer	11.18M	11.17M	89.75± 5.51	41.98± 7.66	34.21± 6.22	35.31 ± 6.79	82.91± 4.46	72.34± 0.51	68.22± 1.34	68.12± 1.24
FEDALT [48]	Adapter	12.59M	11.18M	87.26± 7.78	31.51± 8.55	27.38± 6.65	27.77 ± 7.19	81.41± 6.5	57.88± 3.57	72.13± 1.34	74.67± 1.57
FEDSIM [48]	Adapter	12.59M	11.18M	87.76± 7.57	31.97± 7.44	27.76± 5.78	28.1 ± 6.46	82.14± 5.46	58.62± 3.24	71.75± 1.4	74.09± 1.55
PERADA w/o KD	Adapter	2.82M	1.41M	91.27± 5.15	53.81± 6.27	42.5 ± 5.06	44.45 ± 5.48	83.31± 5.54	76.55± 2.47	76.77± 2.24	77.59± 2.18
PERADA	Adapter	**2.82**M	1.41M	**91.82**± 4.43	**59.05**± 5.24	**47.25**± 4.48	**48.53**± 4.74	**83.58**± 4.74	**77.2** ± 1.63	**76.98**± 3.87	**77.88**± 1.55

*FEDAVG+FT requires full model communciation during FEDAVG training and there is no communciation during local finetuning.

CIFAR-10.1 [51] and common image corruptions (e.g. Blur, Gaussian Noise) on CIFAR-10-C [19].

7.1. Evaluation Results

PERADA is parameter-efficient. ResNet-18 model consists of 11.18 million (M) parameters, and the adapter has 1.41M (12.6%) parameters. Tab. 1 reports each client's # trainable parameters and # communicated parameters to the server. We see that PERADA is most parameter-efficient by locally training two adapters and communicating one adapter. Most full model pFL requires training two full models (PFEDME, APFL, DITTO), and sends one full model to the server. Partial model pFL requires training one full model and communicating its shared parameter. Note that adapter-based partial model pFL in FEDALT and FEDSIM are more expensive than PERADA because they still need to train both a personalized adapter plus a shared full model (12.59M), and communicate the full model. Additional comparison under ResNet-34 shows similar conclusions in Figure 1.

PERADA achieves competitive personalized performance and better generalization than baselines. Tab. 1 shows that even with the smallest number of trainable parameters, PERADA achieves the comparable personalized performance (+1.08%, 0.34%, 4.85% on CIFAR-10, Office-Home, CheXpert) and better generalization (+5.23%, 4.53%, 4.21%, 0.22%, 3.21% on CIFAR-10, CIFAR-10.1, CIFAR-10-C, Office-Home, CheXpert). Specifically, **(a)** PERADA W/O KD already achieves favorable performance compared to the best baseline, which shows that the plug-in module adapter can adapt the pretrained model to FL data distributions, and personalized adapter can successfully encode both local knowledges (with local empirical risk) and generalized knowledge (with regularization). **(b)** PERADA outperforms PERADA W/O KD, which shows that KD improves the generalization of personalized models (Theorem 2). We present the convergence curves in Figure 6 (Appendix B) to show the learning performance from the convergence perspective, where PERADA achieves the best convergence speed.

Table 2. Generalization comparison of the *global* model from different generic FL and pFL methods on CIFAR-10.

Algorithm	Algorithm Type	Trained Params	Global-test	CIFAR-10.1	CIFAR-10-C
FEDAVG [45]	generic FL	Full	69.34	54.95	57.07
FEDPROX [32]	generic FL	Full	69.64	54.75	56.84
FEDDYN [2]	generic FL	Full	70.36	56.3	55.91
FEDDF [39] (w/ KD)	generic FL	Full	74.83	60.95	61.23
pFEDME [59]	pFL	Full	68.25	52.55	56.33
APFL [10]	pFL	Full	69.79	53.6	57.06
DITTO [33]	pFL	Full	69.95	55.25	57.33
PERADA W/O KD	pFL	Adapter	74.22	57.6	61.40
PERADA	pFL	Adapter	**76.77**	**62.5**	**64.47**

To verify that such improvement of pFL is due to an improved global model (Theorem 1), we compare the performance of the *global model* of PERADA to the global model of state-of-the-art methods in pFL (PFEDME, APFL, DITTO) and generic FL (FEDAVG, FEDPROX [32], FEDDYN [2], FEDDF [39]). Note that FEDDF [39] also uses ensemble knowledge distillation for global model aggregation, but updates the full model. Tab. 2 shows that the generalization of PERADA *global* model with adapter also outperforms baselines, and KD indeed improves our global model.

Existing partial model pFL can have poor generalization to out-of-distribution shifts. As shown in Tab. 1, these methods, while showing promising personalized accuracy on CIFAR-10 and sometimes outperform full model pFL on Office-Home and CheXpert by personalizing the right model component, they significantly lag in generalizing to test-time distribution shifts. **(a)** Compared to full model pFL, the root causes of this inferior generalization in existing partial model pFL methods are twofold: **(i)** a smaller number of shared parameters prevents them from effectively learning global information; **(ii)** personalized parameters can predominately encode local information for the partially personalized model. PERADA circumvents such issues by regularization, which enforces personalized adapters to learn *both* local and global information. **(b)** Moreover, the fact that PERADA even w/o KD has better generalization than existing partial pFL methods suggests that updating the shared parameters globally via FL on heterogeneous data can compromise the pretrained feature exactor. Our findings indicate

Table 3. Averaged test accuracy across personalized models with data heterogeneity degrees $\mathrm{Dir}(1)$ and $\mathrm{Dir}(0.3)$ on CheXpert. PerAda achieves best personalized performance and generalization.

Algorithm	Personalization	Local-test		Global-test	
		Dir(1)	Dir(0.3)	Dir(1)	Dir(0.3)
Standalone	Full	64.69 ± 1.63	65.06 ± 1.88	65.32 ± 1.7	65.45 ± 2.3
MTL	Full	65.18 ± 1.95	65.15 ± 1.95	65.67 ± 1.72	65.48 ± 2.3
pFedMe	Full	64.8 ± 1.4	65.07 ± 1.2	64.85 ± 1.25	64.86 ± 1.22
APFL	Full	69.21 ± 1.23	68.98 ± 1.04	69.21 ± 1.05	68.96 ± 1.1
Ditto	Full	68.65 ± 0.82	68.79 ± 1.4	68.72 ± 0.58	75.55 ± 0.34
FedBN	BN	69.09 ± 0.79	68.74 ± 1.17	69.03 ± 0.57	68.83 ± 1.08
FedAlt	Input	67.74 ± 0.85	67.63 ± 1.18	67.88 ± 0.6	67.74 ± 1.1
FedSim	Input	67.65 ± 0.88	67.49 ± 1.32	67.82 ± 0.61	67.54 ± 1.24
LG-FedAvg	Feat. extractor	65.77 ± 1.48	65.78 ± 1.62	66.33 ± 1.38	66.23 ± 1.75
FedRep	Output	66.42 ± 1.62	66.66 ± 1.82	66.49 ± 1.53	66.52 ± 1.47
FedAlt	Output	68.31 ± 0.79	68.27 ± 1.3	68.41 ± 0.47	68.36 ± 1.31
FedSim	Output	68.51 ± 0.82	68.22 ± 1.34	68.63 ± 0.57	68.12 ± 1.24
FedAlt	Adapter	72.52 ± 0.99	72.13 ± 1.34	74.79 ± 1.21	74.67 ± 1.57
FedSim	Adapter	72 ± 1.26	71.75 ± 1.4	74.3 ± 1.51	74.09 ± 1.55
PerAda w/o KD	Adapter	77.45 ± 1.21	76.77 ± 2.24	**78.02** ± 1.36	77.59 ± 2.18
PerAda	Adapter	**77.47** ± 1.54	**76.98** ± 1.81	**78.02** ± 1.55	**77.88** ± 1.55

that maintaining frozen parameters, as done in PerAda without KD, is more effective in preserving the capabilities of the pre-trained model.

Adapter-based personalization methods are generally effective on CheXpert. Tab. 1 shows that adapter-based personalization, including FedAlt, FedSim, PerAda, are especially effective on the X-ray data CheXpert. This conclusion holds under different degrees of data heterogeneity $\mathrm{Dir}(0.3)$ and $\mathrm{Dir}(1)$ in Tab. 3. It indicates that when adapting to FL domains that have a large domain gap for ImageNet pre-trained models, e.g., medical domains, adapter personalization may be preferable to input/output/batch-norm pFL.

Effects of KD. We use CIFAR-100 as the distillation dataset on CIFAR-10, and Figure 4 shows that more distillation steps and distillation data samples are better for pFL generalization. These results echo our theoretical analysis in Theorem 1 that smaller KD optimization error $\Phi_{\mu_{\mathrm{aux}}, n_{\mathrm{aux}}}$ and a larger number of samples can tighten the generalization bounds. We also evaluate different distillation datasets, and Figure 4 shows that out-of-domain datasets (STL-10, CIFAR100) can improve generalization compared to the one without KD (None) by a margin, and achieve comparable performance compared to in-domain CIFAR10 validation data. *The flexibility of choosing distillation* datasets makes it practical for the server to leverage public data for KD.

Another potential way to improve generalization is by moderately increasing regularization strength λ for less personalization. However, Figure 7 (Appendix B) show that an overly large λ degrades the personalized performance, which matches the observation for ℓ_2 regularization-based pFL methods in [48]. Notably, KD does not have such a negative impact on personalized performance (in Figure 4).

Effects of pretrained models. Starting personalization from a pretrained model, such as FedAvg global model, is commonly considered in pFL [44, 48]. Therefore, we first train a ResNet-18 global model on FL data from scratch using FedAvg and utilize it as initialization for pFL. Results in Figure 5 show that PerAda also achieves comparable personalized performance and higher generalization than baselines with FedAvg pretrained model. Moreover,

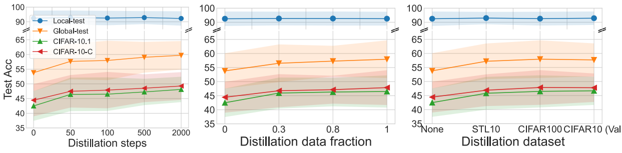


Figure 4. Effect of KD on PerAda evaluated on CIFAR-10. More distillation steps and data samples lead to better generalization and out-of-domain distillation data (STL-10, CIFAR-100) achieve similar performance as in-domain (validation) data.

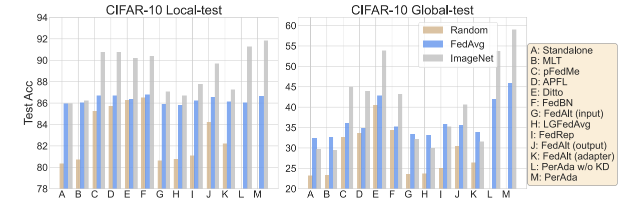


Figure 5. Effect of different initializations (Random, FedAvg model, and ImageNet pretrained model).

ImageNet-pretraining leads to better generalization than FedAvg-pretraining for PerAda, which echos Theorem 1 that high-quality local models (enabled by good pretrained model) can further improve generalization.

Utility under differential privacy guarantees. To further protect local data privacy, we train our method under *sample-level* (ϵ, δ) -differential privacy (DP) [11] on CIFAR-10 with a ViT-S/16-224 model [2]. Following [42], we consider full client participation and perform local training with DP-SGD [1] for *both* personalized models and the global model (see experimental details in Appendix A); We set $\delta = 10^{-5}$ and report averaged ϵ across all clients and averaged pFL accuracy under $\mathrm{Local\text{-}test}$. Tab. 4 shows that (1) PerAda w/o KD retains higher utility than full model personalization Ditto under reasonable privacy guarantees due to a smaller number of trainable parameters and the whole model is less impacted by DP noise. (2) KD with unlabeled *public* data in PerAda can further improve the utility without consuming additional privacy budgets.

Table 4. PerAda retains high personalized utility under DP guarantee on CIFAR-10 with ViT-S/16-224 model.

Algorithm	Personalization	$\epsilon = \infty$	$\epsilon = 5.99 \pm 3.03$	$\epsilon = 3.7 \pm 2.12$	$\epsilon = 1.81 \pm 1.12$
Ditto	Full	**98.59** ± 1.63	76.76 ± 24.14	76.75 ± 24.13	76.67 ± 24.12
PerAda w/o KD	Adapter	97.69 ± 1.79	77.49 ± 21.21	77.32 ± 21.16	76.68 ± 21
PerAda	Adapter	98.08 ± 1.28	**80.33**± 20.76	**79.79** ±20.45	**77.83**± 19.58

8. Conclusion

We propose a pFL framework PerAda based on global/personalized adapter and knowledge distillation with convergence and generalization guarantees, and show that it reduces computation and communication costs and achieves higher personalized performance and generalization.

[2] As batch normalization layer in ResNet creates dependencies between samples and violates DP, we use ViT model [64] for DP experiments.

References

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016. 8, 13

[2] Durmus Alp Emre Acar, Yue Zhao, Ramon Matas, Matthew Mattina, Paul Whatmough, and Venkatesh Saligrama. Federated learning based on dynamic regularization. In *International Conference on Learning Representations*, 2020. 7, 14

[3] Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. Federated learning with personalization layers. *arXiv preprint arXiv:1912.00818*, 2019. 3

[4] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1): 151–175, 2010. 5, 16, 18

[5] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021. 3, 15

[6] Hong-You Chen and Wei-Lun Chao. Fedbe: Making bayesian model ensemble applicable to federated learning. In *International Conference on Learning Representations*, 2020. 3, 4, 6

[7] Hong-You Chen and Wei-Lun Chao. On bridging generic and personalized federated learning for image classification. In *International Conference on Learning Representations*, 2022. 3, 6

[8] Hong-You Chen and Wei-Lun Chao. On bridging generic and personalized federated learning for image classification. In *International Conference on Learning Representations*, 2022. 2

[9] 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. 3, 6, 7

[10] Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Adaptive personalized federated learning. *arXiv preprint arXiv:2003.13461*, 2020. 2, 6, 7

[11] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014. 8

[12] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning: A meta-learning approach. *NeurIPS*, 2020. 2, 3, 6, 25

[13] Dylan J Foster and Alexander Rakhlin. ℓ_∞ vector contraction for rademacher complexity. *arXiv preprint arXiv:1911.06468*, 6, 2019. 17

[14] Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. Clip-adapter: Better vision-language models with feature adapters. *arXiv preprint arXiv:2110.04544*, 2021. 3

[15] Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An efficient framework for clustered federated learning. *Advances in Neural Information Processing Systems*, 33:19586–19597, 2020. 2

[16] Filip Hanzely and Peter Richtárik. Federated learning of a mixture of global and local models. *arXiv preprint arXiv:2002.05516*, 2020. 2, 3

[17] Filip Hanzely, Slavomír Hanzely, Samuel Horváth, and Peter Richtárik. Lower bounds and optimal algorithms for personalized federated learning. *Advances in Neural Information Processing Systems*, 33:2304–2315, 2020. 2

[18] 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*, pages 770–778, 2016. 13

[19] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *Proceedings of the International Conference on Learning Representations*, 2019. 7, 12, 13

[20] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015. 2, 3, 4

[21] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019. 3

[22] Daniel Hsu, Ziwei Ji, Matus Telgarsky, and Lan Wang. Generalization bounds via distillation. In *International Conference on Learning Representations*, 2021. 17, 20

[23] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019. 6, 12

[24] Jeremy Irvin, Pranav Rajpurkar, Michael Ko, Yifan Yu, Silviana Ciurea-Ilcus, Chris Chute, Henrik Marklund, Behzad Haghgoo, Robyn Ball, Katie Shpanskaya, et al. Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison. In *Proceedings of the AAAI conference on artificial intelligence*, pages 590–597, 2019. 6, 12

[25] Liangze Jiang and Tao Lin. Test-time robust personalization for federated learning. *International Conference on Learning Representations*, 2023. 1, 3

[26] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2): 1–210, 2021. 1

[27] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pages 5132–5143. PMLR, 2020. 25

[28] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 6, 12

[29] Gihun Lee, Yongjin Shin, Minchan Jeong, and Se-Young Yun. Preservation of the global knowledge by not-true self

knowledge distillation in federated learning. *arXiv preprint arXiv:2106.03097*, 2021. 3, 6

[30] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021. 3

[31] Daliang Li and Junpu Wang. Fedmd: Heterogenous federated learning via model distillation. *arXiv preprint arXiv:1910.03581*, 2019. 3, 6

[32] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2:429–450, 2020. 7, 14

[33] Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through personalization. In *International Conference on Machine Learning*, pages 6357–6368. PMLR, 2021. 1, 2, 3, 4, 6, 7, 13, 25

[34] Xiang Li, Wenhao Yang, Shusen Wang, and Zhihua Zhang. Communication-efficient local decentralized sgd methods. *arXiv preprint arXiv:1910.09126*, 2019. 30

[35] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. In *International Conference on Learning Representations*, 2020. 1, 25, 26

[36] Xiaoxiao Li, Meirui Jiang, Xiaofei Zhang, Michael Kamp, and Qi Dou. Fedbn: Federated learning on non-iid features via local batch normalization. *arXiv preprint arXiv:2102.07623*, 2021. 3, 6, 7

[37] Xuechen Li, Florian Tramer, Percy Liang, and Tatsunori Hashimoto. Large language models can be strong differentially private learners. In *International Conference on Learning Representations*, 2022. 3

[38] 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. 3, 6, 7

[39] Tao Lin, Lingjing Kong, Sebastian U Stich, and Martin Jaggi. Ensemble distillation for robust model fusion in federated learning. *Advances in Neural Information Processing Systems*, 33:2351–2363, 2020. 3, 4, 5, 6, 7, 14, 16

[40] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*, 2021. 3

[41] Yi Liu, Xiaohan Bi, Lei Li, Sishuo Chen, Wenkai Yang, and Xu Sun. Communication efficient federated learning for multilingual neural machine translation with adapter. *ACL Findings*, 2023. 3

[42] Ziyu Liu, Shengyuan Hu, Zhiwei Steven Wu, and Virginia Smith. On privacy and personalization in cross-silo federated learning. *Advances in Neural Information Processing Systems*, 2022. 8, 13

[43] Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020. 1, 2

[44] Othmane Marfoq, Giovanni Neglia, Richard Vidal, and Laetitia Kameni. Personalized federated learning through local memorization. In *International Conference on Machine Learning*, pages 15070–15092. PMLR, 2022. 2, 4, 8, 15, 16

[45] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, 2017. 1, 2, 4, 7

[46] Kaan Ozkara, Navjot Singh, Deepesh Data, and Suhas Diggavi. Quped: Quantized personalization via distillation with applications to federated learning. *Advances in Neural Information Processing Systems*, 34, 2021. 3, 5, 6, 25

[47] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 13

[48] Krishna Pillutla, Kshitiz Malik, Abdelrahman Mohamed, Michael Rabbat, Maziar Sanjabi, and Lin Xiao. Federated learning with partial model personalization. *ICML*, 2022. 1, 3, 6, 7, 8, 15

[49] Vinay Venkatesh Ramasesh, Aitor Lewkowycz, and Ethan Dyer. Effect of scale on catastrophic forgetting in neural networks. In *International Conference on Learning Representations*, 2022. 3

[50] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. *Advances in neural information processing systems*, 30, 2017. 2, 3

[51] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do cifar-10 classifiers generalize to cifar-10? *arXiv preprint arXiv:1806.00451*, 2018. 7, 12

[52] Sashank J. Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and Hugh Brendan McMahan. Adaptive federated optimization. In *International Conference on Learning Representations*, 2021. 5, 25

[53] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 6, 13

[54] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE transactions on neural networks and learning systems*, 32(8):3710–3722, 2020. 2

[55] Clayton Scott. Rademacher complexity. 2014. 16

[56] Aliaksandra Shysheya, John F Bronskill, Massimiliano Patacchiola, Sebastian Nowozin, and Richard E Turner. Fit: Parameter efficient few-shot transfer learning for personalized and federated image classification. In *The Eleventh International Conference on Learning Representations*, 2023. 3

[57] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. *Advances in neural information processing systems*, 30, 2017. 6, 7

[58] 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, 2021. 3, 6, 12

[59] Canh T Dinh, Nguyen Tran, and Josh Nguyen. Personalized federated learning with moreau envelopes. *Advances in Neural Information Processing Systems*, 33:21394–21405, 2020. 1, 2, 3, 6, 7, 13

[60] Antonio Torralba, Rob Fergus, and William T Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 30(11):1958–1970, 2008. 12

[61] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5018–5027, 2017. 6, 12

[62] Kangkang Wang, Rajiv Mathews, Chloé Kiddon, Hubert Eichner, Françoise Beaufays, and Daniel Ramage. Federated evaluation of on-device personalization. *arXiv preprint arXiv:1910.10252*, 2019. 2

[63] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, 2020. Association for Computational Linguistics. 13

[64] Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Zhicheng Yan, Masayoshi Tomizuka, Joseph Gonzalez, Kurt Keutzer, and Peter Vajda. Visual transformers: Token-based image representation and processing for computer vision, 2020. 8, 13

[65] Tao Yu, Eugene Bagdasaryan, and Vitaly Shmatikov. Salvaging federated learning by local adaptation. *arXiv preprint arXiv:2002.04758*, 2020. 2, 6, 7

[66] Jie Zhang, Song Guo, Xiaosong Ma, Haozhao Wang, Wenchao Xu, and Feijie Wu. Parameterized knowledge transfer for personalized federated learning. *Advances in Neural Information Processing Systems*, 34:10092–10104, 2021. 3, 6

[67] Lin Zhang, Li Shen, Liang Ding, Dacheng Tao, and Ling-Yu Duan. Fine-tuning global model via data-free knowledge distillation for non-iid federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10174–10183, 2022. 3, 4, 6

[68] Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-free knowledge distillation for heterogeneous federated learning. In *International Conference on Machine Learning*, pages 12878–12889. PMLR, 2021. 3, 5, 6, 16