Self-Aware Personalized Federated Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

In the context of personalized federated learning (FL), the critical challenge is to balance local model improvement and global model tuning when the personal and global objectives may not be exactly aligned. Inspired by Bayesian hierarchical models, we develop a self-aware personalized FL method where each client can automatically balance the training of its local personal model and the global model that implicitly contributes to other clients' training. Such a balance is derived from the *inter-client and intra-client uncertainty quantification*. A larger inter-client variation implies more personalization is needed. Correspondingly, our method uses *uncertainty-driven local training steps and aggregation rule* instead of conventional local fine-tuning and sample size-based aggregation. With experimental studies on synthetic data, Amazon Alexa audio data, and public datasets such as MNIST, FEMNIST and Sent140, we show that our proposed method can achieve significantly improved personalization performance compared with the existing counterparts.

1 Introduction

Federated learning (FL) (Konevcny et al., 2016; McMahan et al., 2017) is transforming machine learning (ML) ecosystems from "centralized in-the-cloud" to "distributed across-clients," to potentially leverage the computation and data resources of billions of edge devices (Lim et al., 2020), without raw data leaving the devices. As a distributed ML framework, FL aims to train a global model that aggregates gradients or model updates from the participating edge devices. Recent research in FL has significantly extended its original scope to address the emerging concern of personalization, a broad term that often refers to an FL system that accommodates client-specific data distributions of interest (Ding et al., 2022).

In particular, each client in a personalized FL system holds data that can be potentially non-identically and independently distributed (non-IID). For example, smart edge devices at different houses may collect audio data (Purington et al., 2017) of heterogeneous nature due to, e.g., accents, background noises, and house structures. Each device hopes to improve its on-device model through personalized FL without transmitting sensitive data.

While the practical benefits of personalization have been widely acknowledged, its theoretical understanding remains unclear. Existing works on personalized FL often derive algorithms based on a pre-specified optimization formulation, but explanations regarding the formulation or its tuning parameters rarely go beyond heuristic statements.

In this work, we take a different approach. Instead of specifying an optimization problem to solve, we start with a toy example and develop insights into the nature of personalization from a statistical uncertainty perspective. We aim to answer the following critical questions regarding personalized FL.

(Q1) The lower-bound baselines of personalized FL can be obtained in two cases, i.e., each client performs local training without FL, or all clients participate in conventional FL training. However, the upper-bound for the client is unclear.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

(Q2) Suppose that the goal of each client is to improve its local model performance. How to design an FL training that interpret the global model, suitably aggregate local models and fine-tune each client's local training automatically?

Both questions Q1 and Q2 are quite challenging. Q1 demands a systematic way to characterize the client-specific and globally-shared information. Such a characterization is agnostic to any particular training process being used. To this end, instead of studying personalized data in full generality, we restrict our attention to a simplified and analytically tractable setting: *two-level Bayesian hierarchical models*, where the top and bottom level describes inter-client and intra-client uncertainty, respectively.

The above Q2 requires FL updates to be adaptive to the nature of the underlying discrepancy among client-specific local data, including sample sizes and distributions. The popular aggregation approach that uses a sample size-based weighting mechanism (McMahan et al., 2017) does not account for the distribution discrepancy. Meanwhile, since clients' data is not shared, estimating the underlying data distributions is unrealistic. Consequently, it is highly nontrivial to measure and calibrate such a discrepancy in FL settings. Addressing the above issues is a key motivation of our work.

1.1 Contributions

We make the following technical contributions:

- Interpreting personalization from a *hierarchical model-based* perspective and providing theoretical analyses for FL training.
- Proposing Self-FL, an active personalized FL solution that guides local training and global aggregation via inter- and intra-client *uncertainty quantification*.
- Presenting a novel implementation of Self-FL for deep learning, consisting of automated hyper-parameter tuning for clients and an adaptive aggregation rule.
- Evaluating Self-FL on Sent140 and Amazon Alexa audio data. Empirical results show promising personalization performance compared with existing methods.

To our best knowledge, Self-FL is the first work that connects personalized FL with hierarchical modeling and utilizes uncertainty quantification to drive personalization.

1.2 Related work

Personalized FL. The term *personalization* in FL often refers to the development of client-specific model parameters for a given model architecture. In this context, each client aims to obtain a local model that has desirable test performance on its local data distribution. Personalized FL is critical for applications that involve statistical heterogeneity among clients.

A research trend is to adapt the global model for accommodating personalized local models. To this end, prior works often integrate FL with other frameworks such as multi-task learning (Smith et al., 2017), meta-learning (Jiang et al., 2019; Khodak et al., 2019; Fallah et al., 2020a,b; Al-Shedivat et al., 2021), transfer learning (Wang et al., 2019; Mansour et al., 2020), knowledge distillation (Li and Wang, 2019), and lottery ticket hypothesis (Li et al., 2020a). For example, DITTO (Li et al., 2021) formulates personalized FL as a multi-task learning (MTL) problem and regularizes the discrepancy of the local models to the global model using the ℓ_2 norm. FedEM Marfoq et al. (2021) uses the MTL formulation for personalization under a finite-mixture model assumption and provides federated EM-like algorithms. We refer to Vanhaesebrouck et al. (2017); Hanzely et al. (2020); Huang et al. (2021) for more MTL-based approaches and theoretical bounds on the optimization problem. From the perspective of meta-learning, pFedMe (Dinh et al., 2020) formulates a bi-level optimization problem for personalized FL and introduces Moreau envelopes as clients' regularized loss. PerFedAvg (Fallah et al., 2020b) proposes a method to find a proper initial global model that allows a quick adaptation to local data. pFedHN Shamsian et al. (2021) proposes to train a central hypernetwork for generating personalized models. FedFOMO Zhang et al. (2020) introduces a method to compute the optimally weighted model aggregation for personalization by characterizing the contribution of other models to one client. Later on, we will focus on the experimental comparison of Self-FL with DITTO, pFedMe, and PerFedAvg, since they represent two popular personalized FL formulations, namely multi-task learning and meta-learning.

A limitation of existing personalized FL methods is that they do not provide a clear answer to question Q1. Consequently, it is unclear how to interpret the FL-trained results. For example, in the extreme case where the clients are equipped with irrelevant tasks and data, any personalized FL methods that require a pre-specified global objective (e.g., Ditto (Li et al., 2021), pFedMe (Dinh et al., 2020)) may cause significant biases to local clients.

2 Bayesian Perspective of Personalized FL

We discuss how Self-FL approaches personalized FL with theoretical insights from the Bayesian perspective in this section. The notations are defined as follows. Let $\mathcal{N}(\mu, \sigma^2)$ denote Gaussian distribution with mean μ and variance σ^2. For a positive integer M, let $[M]$ denote the set $\{1, \ldots, M\}$. Let $\sum_{m \neq i}$ denote the summation over all $m \in [M]$ except for $m = i$. We summarize frequently used notation in Table 3 of Appendix and the derivation for general parametric models in Section D.

2.1 Understanding personalized FL through a two-level Gaussian model

To develop insights, we first restrict our attention to the following simplified case. Suppose that there are M clients. From the server's perspective, it is postulated that data $z_1, \ldots, z_M$ are generated from the following two-layer Bayesian hierarchical model:

$$\theta_m \mid \theta_0 \overset{\text{IID}}{\sim} \mathcal{N}(\theta_0, \sigma_0^2), \quad z_m \mid \theta_m \overset{\text{IID}}{\sim} \mathcal{N}(\theta_m, \sigma_m^2), \tag{1}$$

for all clients with indices $m = 1, \ldots, M$. Here, σ_0^2 is a constant, and θ_0 is a hyperparameter with a non-informative flat prior (denoted by π_0). The above model represents both the connections and heterogeneity across clients. In particular, each client's data are distributed according to a client-specific parameter (θ_m), which follows a distribution decided by a parent parameter (θ_0). The parent parameter is interpreted as the root of shared information. In the rest of the section, we often study client 1's local model as parameterized by θ_1 without loss of generality. Under the above model assumption, the parent parameter θ_0 that represents the global model has a posterior distribution $p(\theta_0 \mid z_{1:M}) \sim \mathcal{N}(\theta^{(\text{G})}, v^{(\text{G})})$, where:

$$\theta^{(\text{G})} \triangleq \frac{\sum_{m \in [M]} (\sigma_0^2 + \sigma_m^2)^{-1} z_m}{\sum_{m \in [M]} (\sigma_0^2 + \sigma_m^2)^{-1}}, \quad v^{(\text{G})} \triangleq \frac{1}{\sum_{m \in [M]} (\sigma_0^2 + \sigma_m^2)^{-1}}. \tag{2}$$

The z_m in Eqn. (2) may also be denoted by $\theta_m^{(L)}$, for reasons that will be seen in Eqn. (3). From the perspective of client m, we suppose that the postulated model is Eqn. (1) for $m = 2, \ldots, M$, and $\theta_1 = \theta_0$. It can be verified that the posterior distributions of θ_1 without and with global Bayesian learning are $p(\theta_1 \mid z_1) \sim \mathcal{N}(\theta_1^{(\text{L})}, v_1^{(\text{L})})$ and $p(\theta_1 \mid z_{1:M}) \sim \mathcal{N}(\theta_1^{(\text{FL})}, v_1^{(\text{FL})})$, respectively, which can be computed as:

$$\begin{aligned} &\theta_1^{(\text{L})} \triangleq z_1, \quad v_1^{(\text{L})} \triangleq \sigma_1^2, \\ &\theta_1^{(\text{FL})} \triangleq \frac{\sigma_1^{-2} \theta_1^{(\text{L})} + \sum_{m \neq 1} (\sigma_0^2 + \sigma_m^2)^{-1} \theta_m^{(\text{L})}}{\sigma_1^{-2} + \sum_{m \neq 1} (\sigma_0^2 + \sigma_m^2)^{-1}}, \\ &v_1^{(\text{FL})} \triangleq \frac{1}{\sigma_1^{-2} + \sum_{m \neq 1} (\sigma_0^2 + \sigma_m^2)^{-1}}. \end{aligned} \tag{3}$$

The first and the second distribution above describe the learned result of client 1 from its local data and from all clients' data in hindsight, respectively. Using mean square error as risk, the Bayes estimate of θ_1 or θ_0 is the mean of the posterior distribution, namely $\theta_1^{(\text{L})}$ and $\theta_1^{(\text{FL})}$ as defined above. The flat prior on θ_0 can be replaced with any other distribution to bake prior knowledge into the calculation. We consider the flat prior because the knowledge of the shared model is often vague in practice. The above posterior mean $\theta_1^{(\text{FL})}$ can be regarded as the optimal point estimation of θ_1 given all the clients' data, thus is referred to as "FL-optimal". $\theta^{(\text{G})}$ can be regarded as the "global-optimal." The posterior variance quantifies the reduced uncertainty conditional on other clients' data. Specifically, we define the following *Personalized FL gain* for client 1 as:

$$\text{GAIN}_1 \triangleq \frac{v_1^{(\text{L})}}{v_1^{(\text{FL})}} = 1 + \sigma_1^2 \sum_{m \neq 1} (\sigma_0^2 + \sigma_m^2)^{-1}.$$

Remark 2.1 (**Interpretations of the posterior quantities**). Each client, say client 1, aims to learn θ_1 in the personalized FL context. Its learned information regarding θ_1 is represented by the Bayesian posterior of θ_1 conditional on either its local data z_1 (without communications with others), or the data $z_{1:M}$ in hindsight (with communications). For the former case, the posterior uncertainty described by $v_1^{(\text{L})}$ depends only on the local data quality σ_1^2. For the latter case, the posterior mean $\theta_1^{(\text{FL})}$ is a weighted sum of clients' local posterior means, and the uncertainty will be reduced by a factor of GAIN_1. Since a point estimation of θ_1 is of particular interest in practical implementations, we treat $\theta_1^{(\text{FL})}$ as the theoretical limit in the FL context (recall question Q1). To develop further insights into $\theta_1^{(\text{FL})}$, we consider the following extreme cases.

• As $\sigma_0^2 \to \infty$, meaning that the clients are barely connected, the quantities $\theta_1^{(\text{FL})}$ and $v_1^{(\text{FL})}$ reduce to $\theta_1^{(\text{L})}$ and $v_1^{(\text{L})}$, respectively; meanwhile, the personalized FL gain approaches one, the global parameter mean $\theta^{(\text{G})}$ becomes a simple average of $\theta_m^{(\text{L})}$ (or z_m), and the global parameter has a large variance.

• When $\sigma_0^2 = 0$, meaning that the clients follow the same underlying data-generating process and the personalized FL becomes a standard FL, we have $\theta_1^{(\text{FL})} = \theta^{(\text{G})}$, which is a weighted sum of clients' local optimal solution with weight proportional to σ_m^{-2} (namely client m's precision).

• When σ_1^2 is much smaller than all other σ_m^2's and σ_0^2, and M is not too large, meaning that client 1 has much higher quality data compared with the other clients combined, we have $\theta_1^{(\text{FL})} \approx z_1 = \theta_1^{(\text{L})}$ and $\text{GAIN}_1 \approx 1$. In other words, client 1 almost learns on its own. Meanwhile, client 1 can still contribute to other clients through the globally shared parameter θ_0. For example, the gain for client 2 would be $\text{GAIN}_2 \geq \sigma_2^2/\sigma_1^2$, which is much larger than one.

Remark 2.2 (**Local training steps to achieve** $\theta_1^{(\text{FL})}$). Suppose that client 1 performs ℓ training steps using its local data and negative log-likelihood loss. We show that with a suitable number of steps and initial value, client 1 can obtain the intended $\theta_1^{(\text{FL})}$. The local objective is:

$$\theta \mapsto (\theta - z_1)^2/(2\sigma_1^2) = (\theta - \theta_1^{(\text{L})})^2/(2\sigma_1^2), \tag{4}$$

which coincides with the quadratic loss. Let $\eta \in (0, 1)$ denote the learning rate. By running the gradient descent:

$$\begin{aligned} \theta_1^{\ell} &\leftarrow \theta_1^{\ell-1} - \eta \frac{\partial}{\partial \theta}\left((\theta - \theta_1^{(\text{L})})^2/(2\sigma_1^2)\right)|_{\theta_1^{\ell-1}} \\ &= \theta_1^{\ell-1} - \eta(\theta_1^{\ell-1} - \theta_1^{(\text{L})})/\sigma_1^2 \end{aligned} \tag{5}$$

for ℓ steps with the initial value θ_1^{INIT}, client 1 will obtain:

$$\theta_1^{\ell} = \left(1 - (1 - \sigma_1^{-2}\eta)^{\ell}\right)\theta_1^{(\text{L})} + (1 - \sigma_1^{-2}\eta)^{\ell}\theta_1^{\text{INIT}}. \tag{6}$$

It can be verified that Eqn. (6) becomes $\theta_1^{(\text{FL})}$ in Eqn. (3) if and only if:

$$\theta_1^{\text{INIT}} = \frac{\sum_{m\neq 1}(\sigma_0^2 + \sigma_m^2)^{-1}\theta_m^{(\text{L})}}{\sum_{m\neq 1}(\sigma_0^2 + \sigma_m^2)^{-1}}, \tag{7}$$

$$(1 - \sigma_1^{-2}\eta)^{\ell} = \frac{\sum_{m\neq 1}(\sigma_0^2 + \sigma_m^2)^{-1}}{\sigma_1^{-2} + \sum_{m\neq 1}(\sigma_0^2 + \sigma_m^2)^{-1}}. \tag{8}$$

In other words, with a suitably chosen initial value θ_1^{INIT}, learning rate η, and the number of (early-stop) steps ℓ, client 1 can obtain the desired $\theta_1^{(\text{FL})}$.

3 Proposed Solution for Personalized FL

Our proposed Self-FL framework has three key components as detailed in this section: (i) proper initialization for local clients at each round, (ii) automatic determination of the local training steps, (iii) discrepancy-aware aggregation rule for the global model. These components are interconnected and contribute together to Self-FL's effectiveness. Note that points (i) and (iii) direct Self-FL to the regions that benefit personalization in the optimization space during local training, which is not considered in prior works such as DITTO Li et al. (2021) and pFedMe Dinh et al. (2020). Therefore, Self-FL is more than imposing implicit regularization via early stopping.

3.1 From posterior quantities to FL updating rules

In this section, we show how the posterior quantities of interest in Section 2 can be connected with FL, where clients' parameters suitably deviate from the global parameter, and the global parameter is a proper aggregation of clients' parameters. For notional brevity, we still consider the two-level Gaussian model in Subsection 2.1. For regular parametric models, we mentioned in Subsection D that one may treat z_m as the local optimal solution $\theta_m^{(\text{L})}$, and its variance σ_m^2 as the variance of $\theta_m^{(\text{L})}$ due to finite sample.

Recall that each client m can obtain the FL-optimal solution $\theta_m^{(\text{FL})}$ with the initial value θ_m^{INIT} in Eqn. (7) and tuning parameters η, ℓ in Eqn. (8). Also, it can be shown that θ_m^{INIT} is connected with the

global-optimal $\theta^{(\mathrm{G})}$ defined in Eqn. (2) through:

$$\theta_m^{\mathrm{INIT}} = \theta^{(\mathrm{G})} - \frac{(\sigma_0^2 + \sigma_m^2)^{-1}}{\sum_{k:\, k\neq m}(\sigma_0^2 + \sigma_k^2)^{-1}}(\theta_m^{(\mathrm{L})} - \theta^{(\mathrm{G})}). \tag{9}$$

The initial value θ_m^{INIT} in Eqn. (9) is unknown during training since $\theta_m^{(\mathrm{L})}, \theta^{(\mathrm{G})}$ are unknown. A natural solution is to update θ_m^{INIT}, $\theta_m^{(\mathrm{L})}$, and $\theta^{(\mathrm{G})}$ iteratively, leading to the following personalized FL rule.

Generic Self-FL: At the t-th ($t = 1, 2, \ldots$) round of FL:

• *Each client* m receives the latest global model θ^{t-1} from the server and calculates:

$$\theta_m^{t,\mathrm{INIT}} \triangleq \theta^{t-1} - \frac{(\sigma_0^2 + \sigma_m^2)^{-1}}{\sum_{k:\, k\neq m}(\sigma_0^2 + \sigma_k^2)^{-1}}(\theta_m^{t-1} - \theta^{t-1}), \tag{10}$$

where θ_m^{t-1} is client m's latest personal parameter at round $t-1$, initialized to be θ^0. Starting from the above $\theta_m^{t,\mathrm{INIT}}$, client m performs gradient descent-based local updates with optimization parameters following Eqn. (8) or its approximations, and obtains a personal parameter θ_m^t.

• *Server* collects θ_m^t, $m = 1, \ldots, M$, and calculates:

$$\theta^t \triangleq \frac{\sum_{m\in[M]}(\sigma_0^2 + \sigma_m^2)^{-1}\theta_m^t}{\sum_{m\in[M]}(\sigma_0^2 + \sigma_m^2)^{-1}}. \tag{11}$$

In general, the above σ_0^2, σ_m^2 represent "inter-client uncertainty" and "intra-client uncertainty," respectively. When σ_0^2 and σ_m^2's are unknown, they can be approximated using asymptotics of M-estimators or using practical finite-sample approximations (elaborated in Subsection 3.2).

We provide a theoretical understanding of the convergence. Consider the data-generating process that $\theta_m^{(\mathrm{L})} \sim \mathcal{N}(\theta_m, \sigma_m^2)$ are independent, and $\theta_m \overset{\mathrm{IID}}{\sim} \mathcal{N}(\theta_0, \sigma_0^2)$ where σ_0^2 is a fixed constant and each σ_m^2 may or may not depend on M. Let O_p denote the standard stochastic boundedness. The following result gives an error bound of each client's personalized parameter and the server's parameter.

Proposition 3.1. *Assume that* $\max_{m\in[M]} \sigma_m^2$ *is upper bounded by a constant, and there exists a constant* $q \in (0, 1)$ *such that:*

$$\max_{m\in[M]} \frac{\sum_{k:\, k\neq m}(\sigma_k^2 + \sigma_0^2)^{-1}}{\sigma_m^{-2} + \sum_{k:\, k\neq m}(\sigma_k^2 + \sigma_0^2)^{-1}} \leq q. \tag{12}$$

Suppose that at $t = 0$*, the gap between the initial parameter and each client's FL-optimal value satisfies* $|\hat{\theta}^0 - \theta_m^{(\mathrm{FL})}| \leq C$ *for all* $m \in [M]$*. Then, for every positive integer* t*, the quantities* $\max_{m\in[1:M]} |\theta_m^t - \theta_m^{(\mathrm{FL})}|, |\theta^t - \theta^{(\mathrm{G})}|$ *are both upper bounded by* $C \cdot q^t + O_p(M^{-1/2})$ *as* $M \to \infty$.

Remark 3.2 (*Interpretation of Proposition* 3.1). The proposition shows that the estimation error of each personalized parameter θ_m^t and the server parameter θ^t can uniformly go to zero as the number of clients M and the FL round t go to infinity. The error bound involves two terms. The first term q^t corresponds to the optimization error. Typically, every σ_m is a small value. If each client has a sample size of N, q will be at the order of $M/(N + M)$. The second term $O_p(M^{-1/2})$ is a statistical error that vanishes as more clients participate. Intuitively, the initial model of each client and the global model at each round are averaged over many clients, so a larger pool leads to a smaller bias. The proof is nontrivial because the averaged terms are statistically dependent. For the particular case that each σ_m^2 is at the order of N^{-1}, we can see that the error is tiny when both M and N/M grow.

3.2 SGD-based practical algorithm for deep learning

For the training method proposed in Subsection 3.1, the quantities σ_0^2 and σ_m^2 are crucial as they affect the choice of learning rate η_m and the early-stop rule. For many complex learning models, we do not know σ_0^2 and σ_m^2, and the asymptotic approximation of σ_0^2 and σ_m^2's may not be valid due to a lack of regularity conditions. Furthermore, one often uses SGD instead of GD to perform local updates, so the stop rule depends on the batch size. In this subsection, we consider the above aspects and develop a practical algorithm for deep learning. Following the discussions in Subection D, we can generally treat σ_m^2 as "uncertainty of the local optimal solution $\theta_m^{(\mathrm{L})}$ of client m", and σ_0^2 as "uncertainty of clients' underlying parameters." We propose a way to approximate them.

Assume that for each client m, we had u independent samples of its data and the corresponding local optimal parameter $\theta_{m,1}, \ldots, \theta_{m,u}$. We could then estimate σ_m^2 by their sample variance. In practice, we can treat each round of local training as the optimization from a bootstrapped sample. In other words, at the round t, let client m run sufficiently many steps (not limited by our tuning parameter ℓ) until it approximately converges to a local minimum, denoted by $\theta_{m,t}$. To save computational costs, a client does not necessarily wait for its local convergence. Instead, we recommend that each client use its personal parameter as a surrogate of $\theta_{m,t}$ at each round t, namely to use θ_m^t. In other words, at round t, we approximate σ_m^2 with:

$$\widehat{\sigma_m^2} = \text{empirical variance of } \{\theta_m^1, \ldots, \theta_m^t\}. \tag{13}$$

Likewise, at round t, we estimate σ_0^2 by:

$$\widehat{\sigma_0^2} = \text{empirical variance of } \{\theta_1^t, \ldots, \theta_M^t\}. \tag{14}$$

For multi-dimensional parameters, a counterpart of the derivations in earlier sections will involve matrix multiplications, which does not fit the usual SGD-based learning process. We thus simplify the problem by introducing the following uncertainty measures. For vectors $x_1, \ldots, x_M$, their empirical variance is defined as the trace of $\sum_{m\in[M]}(x_m - \bar{x})(x_m - \bar{x})^{\mathrm{T}}$, which is the sum of entry-wise empirical variances. $\widehat{\sigma_m^2}$ and $\widehat{\sigma_0^2}$ will be defined from such empirical variances, similar to Eqn. (13) and (14). In practice, for large neural network models, we suggest deploying Self-FL after the baseline FL training runs for certain rounds (namely warm-start) so that the fluctuation of local models does not hinder variance approximation.

Combining the above discussion and the generic method in Subsection 3.1, we propose our personalized FL solution in Algorithm 1. We include the implementation details of Self-FL in Section 4 and Appendix. We derived how to learn $\theta^{(\mathrm{G})}$ and $\theta_m^{(\mathrm{FL})}$ for parametric models in Subsection D. The key motivation of Algorithm 1 is that a client often cannot obtain the exact $\theta_m^{(\mathrm{L})}$ (especially in DL), so we interweave local training and personalization through an iterative FL procedure. Note that the computation of $\theta^{(\mathrm{G})}$ and $\theta_m^{(\mathrm{FL})}$ requires uncertainty measurements approximated from communications between the clients and the server.

Algorithm 1 Active Personal FL (Self-FL)

Input: A server and M clients. Communication rounds T, client activity rate C, client m's local data D_m and learning rate η_m.

for each communication round $t = 1, \ldots T$ **do**
- Sample clients: $\mathbb{M}_t \leftarrow \max(\lfloor C \cdot M \rfloor, 1)$
- **for** each client $m \in \mathbb{M}_t$ **in parallel do**
  - Distribute server model θ^{t-1} to client m
  - Estimate $\widehat{\sigma_m^2}$ using Eqn. (13)
  - Compute local step l_m from Eqn. (8) and local initialization θ_m^{INIT} via Eqn. (7)
  - $\theta_m^t \leftarrow LocalTrain(\theta_m^{\mathrm{INIT}}, \eta_m, l_m; D_m)$
- Server estimates $\widehat{\sigma_0^2}$ using Eqn. (14)
- Server performs global aggregation and updates global model θ^t via Eqn. (11)

4 Experimental Studies

4.1 Experimental setup

We use AWS p316 instances for all experiments. To evaluate the performance of Self-FL, we consider synthetic data, images, texts, and audios. The loss in Alg. 1 can be general. For our experiments on MNIST, FEMNIST, CIFAR10, Sent140, and wake-word detection, we used the cross-entropy loss. Our empirical results on DL problems suggest that Self-FL provides promising personalization performance in complex scenarios.

Synthetic Data. We construct a synthetic dataset based on the two-layer Bayesian model discussed in Subsection 2.1 where each data point is a scalar. The FL system has a total of $M = 20$ clients, and the task is to estimate the global and local model parameters. The heterogeneity level of the data can be controlled by specifying the inter-client uncertainty σ_0^2 and the local dataset size N_m for each client m. We construct two synthetic FL datasets, one with high homogeneity and one with high heterogeneity, to investigate the performance of FL algorithms in different scenarios. The empirical results are provided in Appendix Subsection G.1.

MNIST Data. This image dataset has 10 output classes and input dimension of 28×28. We use a multinomial logistic regression model for this task. The FL system has a total of $1,000$ clients. We use the same non-IID MNIST dataset as FedProx (Li, 2020), where each client has samples of two-digit classes, and the local sample size of clients follows a power law (Li et al., 2020b).

FEMNIST Data. The federated extended MNIST (FEMNIST) dataset has 62 output classes. The FL system has $M = 200$ clients. We use the same approach as FedProx (Li et al., 2020b) for heterogeneous data generation where ten lower-case characters ('a'-'j') from EMNIST are selected, and each client holds five classes of them.

CIFAR10 Data. This image dataset has images of dimension $32 \times 32 \times 3$ and 10 output classes. We use the non-i.i.d. data provided by the pFedMe repository Canh T. Dinh (2020). There are a total of 20 clients in the system where each user has images of three classes.

Sent140 Data. This is a text sentiment analysis dataset containing tweets from Sentiment140 Go et al. (2009). It has two output classes and 772 clients. We generate non-i.i.d. data using the same procedure as FedProx Li (2020). Detailed setup and our experimental results on Sent140 are provided in Appendix Subsection G.3.

Private Wake-word Data. We also evaluate Self-FL on a private audio data from Amazon Alexa for the wake-word detection task. This is a common task for smart home devices where the on-device model needs to determine whether the user says a pre-specified keyword to wake up the device. This task has two output classes, namely 'wake-word detected' and 'wake-word absent.' The audio stream from the speaker is represented as a spectrogram and is fed into a convolution neural network (CNN). The CNN is a stack of convolutional and max-pooling layers (11 in total). The number of trainable parameters is around $3 \cdot 10^6$. The heterogeneity between clients comes from the intrinsic property of different device types (e.g., determined by hardware and use scenarios). We use an in-house far-field corpus that contains de-identified far-field data collected from millions of speakers under different conditions with users' content. This dataset contains 39 thousand hours of training data and 14 thousand hours of test data.

To support practical FL systems where only a subset of clients participate in one communication round (also known as client sampling), Self-FL can update the global model θ^t with *smoothing average*. Particularly, given an activity rate (i.e., ratio of clients selected in each round) $C \in (0, 1)$, we first obtain the current estimation $\hat{\theta}^t$ by applying Eqn. (11) to the active clients. Then, the server updates the global model using $\theta^t = (1 - C) \cdot \theta^{t-1} + C \cdot \hat{\theta}^t$. Updating the global model with a smoothing average allows us to integrate historical information and reduce instability, especially for a small activity rate.

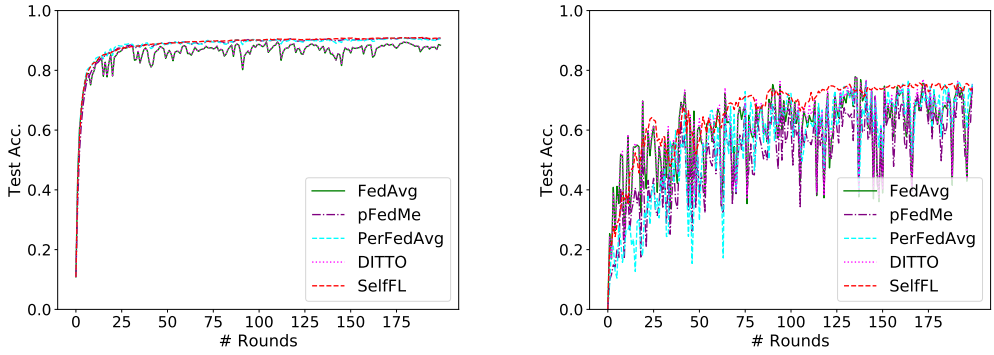
4.2 Effectiveness on the image domain

Results on MNIST and FEMNIST. A description of MNIST and FEMNIST data are detailed in Subsection 4.1. The activity rate is set to $C = 0.1$. For MNIST, we use learning rate $\eta = 0.03$ and batch size 10 as suggested in FedProx (Li, 2020). For FEMNIST, we use $\eta = 0.01$ and batch size 10. The FL training starts from scratch and runs for 200 rounds. For comparison, we implement FedAvg and three other personalized FL techniques, DITTO (Li et al., 2021), pFedMe (Dinh et al., 2020), and PerFedAvg (Fallah et al., 2020b) with the same hyper-parameter configurations.

For real-world data, the theoretical value of the local training steps l_m computed using Eqn. (16) might be too large for the client (due to limitation of client's computation/memory budget). In this experiment, we determine the actual local training steps for Self-FL framework as $\hat{l}_m = \min\{l_m, l_{\text{max}}\}$ where l_{max} is a pre-defined cutoff threshold. We use a fixed local training step of 20 for other FL algorithms and set $l_{\text{max}} = 40$ for Self-FL. Note that σ_m^2 represents the variance of θ_m (i.e., model parameters such as neural coefficients). In our experiments, σ_m^2 is approximated using the empirical variance of θ_m across rounds.

Figures 1a and 1b compare the convergence performance of different FL algorithms on MNIST and FEMNIST, respectively. The horizontal axis denotes the communication round. The vertical axis denotes the weighted test accuracy of individual clients, where the weight is proportional to the sample size. We observe that: (i) Self-FL algorithm yields more *stable convergence* compared with FedAvg and the other three personalized FL methods due to our smoothing average in aggregation; (ii) Self-FL achieves the highest personalized accuracy when the global model converges.

To corroborate that the performance advantage of Self-FL does not come from a larger potential local training steps (we set $l_{max} = 40$ for Self-FL), we perform an additional experiment where we use a

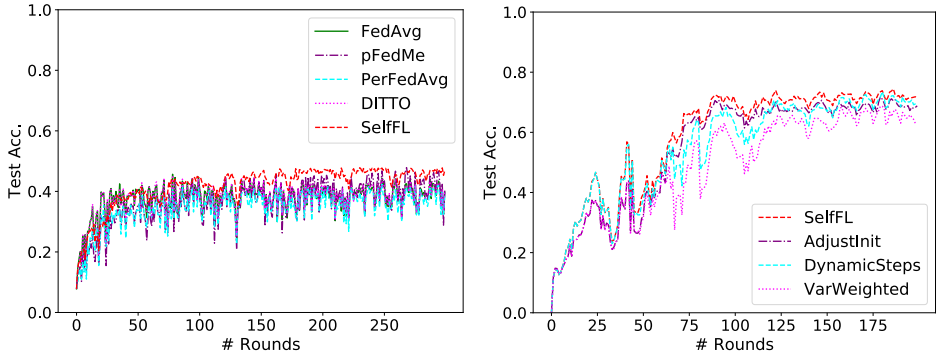


(a) MNIST (b) FEMNIST

Figure 1: Weighted client-level test accuracy in different rounds (with 0.1 activity rate).

local step as $l = 40$ (instead of 20) for DITTO, pFedMe, and PerFedAvg on the FEMNIST dataset. In this scenario, the test accuracy of these three FL algorithms is 71.33%, 68.87%, and 71.08%, respectively. Our proposed method achieves a test accuracy of 76.93%. Furthermore, it is worth noting that other FL methods do not have a principled way to determine the value of l. The adaptive local steps in Self-FL also address this challenge.

Results on CIFAR10. We also perform experiments on the CIFAR-10 dataset (from the pFedMe repository Canh T. Dinh (2020)) with the CNN model from PyTorch example PyTorch (2022). In each round, we randomly select 5 out of 20 clients to participate in FL training. We use a learning rate of 0.01 for all FL algorithms and set the local training step as 40 for the baseline methods. Figure 2a visualizes the test accuracy comparison, showing that Self-FL achieves a higher accuracy.



(a) CIFAR10 (b) FEMNIST

Figure 2: Weighted client-level test accuracy.

Ablation study on Self-FL. Recall that Self-FL consists of three key components: (i) Adjusting local initialization; (ii) Dynamic local training steps; (iii) Variance-weighted global aggregation (Section 3). To investigate the effect of each component, we perform an ablation study where we apply the full deployment of Self-FL and only only component of it. Figure 2b compares the test accuracy of the global model on FEMNIST benchmark in these four settings. We can observe that adjusting the local initialization at each round (Eqn. (10)) gives the highest contribution to Self-FL's performance.

To characterize the distribution of personalized performance across clients, we define two new metrics to evaluate different FL algorithms: (i) weighted test accuracy of clients that have top 10% most samples; (ii) worst 10% clients' average test accuracy. The evaluation results on FEMNIST (where the client sample size ranges from 1 to 206) are shown in Table 1 (standard errors in parentheses). We can observe that Self-FL outperforms other FL algorithms in terms of these additional metrics.

4.3 Effectiveness on the audio domain

In this section, we evaluate the performance of Self-FL on an audio dataset (Subsection 4.1) for wake-word detection. Particularly, we use a CNN that is pre-trained on the training data of different device types (i.e., heterogeneous data) as the initial global model to *warm-start* FL training for all evaluated FL algorithms. The personalization task aims to improve the wake-word detection performance at the

Table 1: Test accuracy comparison on FEMNIST.

FL Alg.	FedAvg	DITTO	pFedMe	PerFedAvg	Self-FL
Top 10% most samples	70.48% (0.07%)	70.58% (0.14%)	72.76% (0.28%)	74.95% (0.56%)	76.24% (0.07%)
Worst 10% user avg acc	0% (0%)	0% (0%)	17.97% (1.23%)	24.51% (1.55%)	37.01% (1.57%)

device type level. In this experiment, we assume there are five clients in the FL system. Each client only has the training data for a specific device without data overlapping. Furthermore, all the clients participate in each communication round. Note that FedAvg (McMahan et al., 2017) and DITTO (Li et al., 2021) typically use the sample size-based weighted average to aggregate models, which do not take data heterogeneity into account. For comparison, we implement FedAvg and DITTO with both equal-weighted and sample size-based weighted model averaging (denoted by the suffix "-e' and '-w', respectively) during aggregation. For meta-learning-based PerFedAvg (Fallah et al., 2020b), we use its first-order approximation and equal-weighted aggregation. We did not report pFedMe on this dataset because we found it did not converge with various hyper-parameters.

Evaluation metric. One can control the trade-off between false accepts and false rejects of wake-word detection by tuning the threshold. Since we use a pre-trained model to warm-start FL, we first evaluate the detection performance of this pre-trained model as the baseline. To compare the performance of different FL algorithms, we use the *relative false accept (FA)* value of the resulting model when the corresponding relative false reject (FR) is close to one as the metric. So a relative FA smaller than one is preferred. Here, the relative FA and FR are computed with respect to the baseline. We detail the results in two scenarios below.

Table 2a summarizes the results of the updated global model. Recall that a smaller relative FA indicates better performance. Each column reports the relative FA for a specific device type. The results show that Self-FL achieves the lowest relative FA. Several updated global models have worse performance than the baseline model, e.g., for FedAvg-w and DITTO-w in Table 2a. This is due to the setting here that local models are initialized from a pre-trained model, and the comparison is relative to that warm-start. Because clients' data are highly heterogeneous, a sample size-based aggregation rule used in FedAvg-w and Ditto-w results in a deteriorated global model compared with the original start. We provide ablation studies in the client sampling setting in the Appendix Subsection G.4, which show Self-FL still performs well.

Table 2: Detection performance (relative FA) of model on a test dataset.

FL methods	Device Types				
	A	B	C	D	E
Self-FL	**0.92**	**0.94**	**0.91**	**0.91**	1.01
FedAvg-w	8.39	4.00	12.80	8.61	10.62
FedAvg-e	0.97	0.96	1.00	0.92	1.00
DITTO-w	8.38	4.00	12.75	8.61	10.23
DITTO-e	0.97	0.95	1.00	0.93	**0.99**
PerFedAvg	1.06	0.98	1.08	0.93	1.01

(a) Global model.

FL methods	Device Type				
	A	B	C	D	E
Self-FL	**0.93**	**0.91**	**0.90**	**0.90**	0.99
FedAvg-e	0.95	0.95	0.93	0.91	0.98
DITTO-e	0.97	0.96	0.93	0.91	0.96
PerFedAvg	1.02	1.11	1.08	1.00	**0.93**

(b) Personalized model.

We further compare the personalization performance of Self-FL with FedAvg-e and two personalized FL algorithms, DITTO-e and PerFedAvg, on each device type's test data and summarize the results in Table 2b. We do not consider FedAvg-w and DITTO-w in this experiment since they result in performance degradation, as seen from Table 2a. One can see that Self-FL outperforms the other methods across all device types, thus demonstrating a better personalization capability.

5 Concluding Remarks

In this paper, we proposed Self-FL to address the challenge of balancing local model regularization and global model aggregation in personalized FL. Its key component is using uncertainty-driven local training steps and aggregation rules instead of conventional local fine-tuning and size-based weights. Overall, Self-FL connects personalized FL with hierarchical modeling and utilizes uncertainty quantification to drive personalization. Extensive empirical studies of Self-FL show its promising performance. We do not envision any negative societal impact of the work.

The **Appendix** contains further experimental studies, remarks, related works, and technical proofs.

References

Jakub Konevcny, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proc. AISTATS*, pages 1273–1282. PMLR, 2017.

Wei Yang Bryan Lim, Nguyen Cong Luong, Dinh Thai Hoang, Yutao Jiao, Ying-Chang Liang, Qiang Yang, Dusit Niyato, and Chunyan Miao. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 2020.

Jie Ding, Eric Tramel, Anit Kumar Sahu, Shuang Wu, Salman Avestimehr, and Tao Zhang. Federated learning challenges and opportunities: An outlook. *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2022.

Amanda Purington, Jessie G Taft, Shruti Sannon, Natalya N Bazarova, and Samuel Hardman Taylor. "alexa is my new bff" social roles, user satisfaction, and personification of the amazon echo. In *Proceedings of the 2017 CHI conference extended abstracts on human factors in computing systems*, pages 2853–2859, 2017.

Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. In *Advances in Neural Information Processing Systems*, pages 4424–4434, 2017.

Yihan Jiang, Jakub Konevcnỳ, Keith Rush, and Sreeram Kannan. Improving federated learning personalization via model agnostic meta learning. *arXiv preprint arXiv:1909.12488*, 2019.

Mikhail Khodak, Maria-Florina F Balcan, and Ameet S Talwalkar. Adaptive gradient-based meta-learning methods. In *Advances in Neural Information Processing Systems*, pages 5917–5928, 2019.

Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning: A meta-learning approach. *arXiv preprint arXiv:2002.07948*, 2020a.

Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning: A meta-learning approach. *arXiv preprint arXiv:2002.07948*, 2020b.

Maruan Al-Shedivat, Liam Li, Eric Xing, and Ameet Talwalkar. On data efficiency of meta-learning. In *International Conference on Artificial Intelligence and Statistics*, pages 1369–1377. PMLR, 2021.

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

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.

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

Ang Li, Jingwei Sun, Binghui Wang, Lin Duan, Sicheng Li, Yiran Chen, and Hai Li. Lotteryfl: Personalized and communication-efficient federated learning with lottery ticket hypothesis on non-iid datasets. *arXiv preprint arXiv:2008.03371*, 2020a.

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.

Othmane Marfoq, Giovanni Neglia, Aurélien Bellet, Laetitia Kameni, and Richard Vidal. Federated multi-task learning under a mixture of distributions. *Advances in Neural Information Processing Systems*, 34, 2021.

Paul Vanhaesebrouck, Aurélien Bellet, and Marc Tommasi. Decentralized collaborative learning of personalized models over networks. In *Artificial Intelligence and Statistics*, pages 509–517. PMLR, 2017.

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.

Yutao Huang, Lingyang Chu, Zirui Zhou, Lanjun Wang, Jiangchuan Liu, Jian Pei, and Yong Zhang. Personalized cross-silo federated learning on non-iid data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7865–7873, 2021.

Canh T Dinh, Nguyen H Tran, and Tuan Dung Nguyen. Personalized federated learning with moreau envelopes. *arXiv preprint arXiv:2006.08848*, 2020.

Aviv Shamsian, Aviv Navon, Ethan Fetaya, and Gal Chechik. Personalized federated learning using hypernetworks. In *International Conference on Machine Learning*, pages 9489–9502. PMLR, 2021.

Michael Zhang, Karan Sapra, Sanja Fidler, Serena Yeung, and Jose M Alvarez. Personalized federated learning with first order model optimization. In *International Conference on Learning Representations*, 2020.

Tian Li. Github repo of paper federated optimization in heterogeneous networks. `https://github.com/litian96/FedProx`, July 2020.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In *Proceedings of Machine Learning and Systems*, volume 2, pages 429–450, 2020b.

Tuan Dung Nguyen Canh T. Dinh, Nguyen H. Tran. Personalized federated learning with moreau envelopes (neurips 2020). `https://github.com/CharlieDinh/pFedMe`, Jan 2020.

Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. *CS224N project report, Stanford*, 1(12):2009, 2009.

PyTorch. Training a classifier. `https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html`, April 2022.

Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konevcnỳ, Stefano Mazzocchi, H Brendan McMahan, et al. Towards federated learning at scale: System design. *arXiv preprint arXiv:1902.01046*, 2019.

Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. Qsgd: Communication-efficient sgd via gradient quantization and encoding. In *Advances in Neural Information Processing Systems*, pages 1709–1720, 2017.

Nikita Ivkin, Daniel Rothchild, Enayat Ullah, Ion Stoica, Raman Arora, et al. Communication-efficient distributed sgd with sketching. In *Advances in Neural Information Processing Systems*, pages 13144–13154, 2019.

Enmao Diao, Jie Ding, and Vahid Tarokh. HeteroFL: Computation and communication efficient federated learning for heterogeneous clients. *International Conference on Learning Representations (ICLR)*, 2020.

Maruan Al-Shedivat, Jennifer Gillenwater, Eric Xing, and Afshin Rostamizadeh. Federated learning via posterior averaging: A new perspective and practical algorithms. *arXiv preprint arXiv:2010.05273*, 2020.

Hong-You Chen and Wei-Lun Chao. Fedbe: Making bayesian model ensemble applicable to federated learning. *arXiv preprint arXiv:2009.01974*, 2020.

Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konevcnỳ, Sanjiv Kumar, and H Brendan McMahan. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.

Mikhail Khodak, Renbo Tu, Tian Li, Liam Li, Maria-Florina F Balcan, Virginia Smith, and Ameet Talwalkar. Federated hyperparameter tuning: Challenges, baselines, and connections to weight-sharing. *Advances in Neural Information Processing Systems*, 34, 2021.

Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020c.

Takayuki Nishio and Ryo Yonetani. Client selection for federated learning with heterogeneous resources in mobile edge. In *ICC 2019-2019 IEEE International Conference on Communications (ICC)*, pages 1–7. IEEE, 2019.

Tian Li, Maziar Sanjabi, Ahmad Beirami, and Virginia Smith. Fair resource allocation in federated learning. *arXiv preprint arXiv:1905.10497*, 2019.

Xun Xian, Xinran Wang, Jie Ding, and Reza Ghanadan. Assisted learning: a framework for multi-organization learning. *Proc. NeurIPS 2020*, 2020.

Enmao Diao, Jie Ding, and Vahid Tarokh. Gradient assisted learning. *arXiv preprint arXiv:2106.01425*, 2021a.

Enmao Diao, Vahid Tarokh, and Jie Ding. Privacy-preserving multi-target multi-domain recommender systems with assisted autoencoders. *arXiv preprint arXiv:2110.13340*, 2021b.

Xinran Wang, Yu Xiang, Jun Gao, and Jie Ding. Information laundering for model privacy. *Proc. ICLR*, 2021.

Aad W Van der Vaart. *Asymptotic statistics*, volume 3. Cambridge university press, 2000.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

Enmao Diao, Jie Ding, and Vahid Tarokh. SemiFL: Communication efficient semi-supervised federated learning with unlabeled clients. *arXiv preprint arXiv:2106.01432*, 2021c.

Jiawei Zhang, Jie Ding, and Yuhong Yang. A binary regression adaptive goodness-of-fit test. *Journal of the American Statistical Association*, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] Our empirical results in Section 4 justify ours claims.
   (b) Did you describe the limitations of your work? [Yes] We discuss our limitations in Appendix Section G.7.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] We do not envision any potential negative societal impacts of this work and mentioned that in Section 5.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] . We state the full assumptions in Section 2 of the main body and Section D in the Appendix.
   (b) Did you include complete proofs of all theoretical results? [Yes] . We provide complete proofs in Section 2, Appendix Section D and Section H.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We have included the codes in the supplementary material.

(b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] . We specify the training details in Section 4.1 and Appendix Section G.

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] . We report the standard error of our experiments in Section 4 and Appendix Section G.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] . We provide the computing resources in Section 4.1.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [Yes] . We use the open-sourced GitHub repository of FedProx, pFedMe, and PerFedAvg in our experiments (Section 4) and we cited the authors.

(b) Did you mention the license of the assets? [Yes] .

(c) Did you include any new assets either in the supplemental material or as a URL? [No]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Appendix

This **Appendix** is structured as follows. We provide a summary of frequently used notations in Section B and additional related works in Section C. We show how Self-FL learns in the setting of a general parametric model in Section D. In Section E, we discuss how the uncertainty quantity in Algorithm 1 can be computed in an online fashion. In Section F, we introduce a variant of Self-FL named "Augmented Self-FL", which serves as a natural alternative to Algorithm 1. Then, in Section G, we conduct ablation studies and provide additional experimental results. Finally, we provide the proof of Proposition 3.1 in Section H.

B Summary of Notations

We summarize the frequently used notations in Table 3.

Table 3: Frequently used notations

Notation	Meaning
M	number of clients
σ_m^2	variance of client m's estimated parameters, or the "intra-client uncertainty" in general
σ_0^2	variance of different clients' underlying parameters, or the "inter-client uncertainty" in general
$\theta_m^{(\mathrm{L})}$, $v_m^{(\mathrm{L})}$	posterior mean and variance of client m's personal parameter
$\theta_m^{(\mathrm{FL})}$, $v_m^{(\mathrm{FL})}$	posterior mean and variance of client m's personal parameter conditional on all the data
$\theta^{(\mathrm{G})}$, $v^{(\mathrm{G})}$	global model's posterior mean and variance
η	learning rate
ℓ	learning steps
t	FL round
θ_m^t	client m's FL-optimal parameter
$\theta_{m,t}$	client m's local optimal parameter at round t
θ^t	global parameter at around t

C Further Remarks on Related Work

Federated learning. A general goal of FL is to train massively distributed models at a large scale (Bonawitz et al., 2019). FedAvg (McMahan et al., 2017) is perhaps the most widely adopted FL baseline, which reduces communication costs by allowing clients to train the local models for multiple iterations. To further reduce communication costs, data compression techniques such as quantization and sketching have been developed for FL (Konevcny et al., 2016; Alistarh et al., 2017; Ivkin et al., 2019). To further reduce computation costs, techniques to train a large model using small-capacity devices such as HeteroFL (Diao et al., 2020) have been developed.

Bayesian Federated Learning. There have been prior work that use the Bayesian perspective in FL. For instance, FedPA Al-Shedivat et al. (2020) formulates FL as a posterior inference problem where the global posterior is obtained by averaging the clients' local posteriors. FedBE Chen and Chao (2020) proposes an aggregation method by interpreting local models as samples from the global model distribution and leveraging Bayesian model ensemble to aggregate the predictions. There are two main differences between Self-FL and the above works. First, both FedPA and FedBE aim to learn a global FL model on the server side instead of personalized models for clients (the focus of our paper). Thus, the developed update rules and use scenarios are entirely different. Second, our approach is based on a novel two-level hierarchical Bayes perspective that leverages the inter-client and intra-client uncertainty to drive FL optimization. FedPA and FedBE use standard (one-level) Bayesian posterior to update the global model.

Adaptive Federated Learning. There has been a direction of research to automate the hyperparameter tuning process in FL. For example, FedOPT Reddi et al. (2020) proposes an FL framework with server and client optimizers to improve FL convergence instead of personalization (our focus). The adaptive server optimization in Reddi et al. (2020) is orthogonal to Self-FL. FedOPT involves tuning hyperparameters for the server's adaptive optimizer (e.g., selecting β_1, β_2, and ϵ for Adam). Also, its

aggregation rule is the same as standard FL and can be possibly replaced by our method for better performance. FedEx Khodak et al. (2021) proposes a hyperparameter tuning method inspired by weight sharing in neural architecture search. Although FedEx can auto-tune local hyperparameters, it may need extra efforts to determine its own hyperparameters, such as η_t, λ_t, and configurations c. In terms of the update rules, FedEx and Self-FL are also very different. In particular, a client in FedEx will sample a configuration c from a hyperparameter distribution D, and the server will perform an exponentiated update for D; Self-FL will estimate empirical variances (online) to simultaneously determine local initialization, training steps, and global model aggregation.

Heterogeneous FL. Earlier studies of FL typically assumed that local models share the same architecture as the global model (Li et al., 2020c) to produce a single global inference model. This assumption limits the global model's complexity for the most data-indigent client. To personalize the computation and communication capabilities of each client, the work of (Diao et al., 2020) proposed a new FL framework named HeteroFL to train heterogeneous local models and still produce a single global inference model. Although this model heterogeneity can be regarded as personalization of client-side resources, it differs significantly from personalized FL in this work, where our goal is to derive client-specific models due to clients' heterogeneous data. To address system heterogeneity, methods based on asynchronous communication and active client sampling have also been developed (Bonawitz et al., 2019; Nishio and Yonetani, 2019). To encourage a fair distribution of accuracy across clients, a method based on client-weighted loss was developed in Li et al. (2019).

Assisted learning. Beyond edge computing, emerging real-world applications concern the collaboration among organizational learners such as research labs, government agencies, or companies. However, to avoid leaking useful and possibly proprietary information, an organization typically enforces stringent security measures, significantly limiting such collaboration. Assisted learning (AL) (Xian et al., 2020; Diao et al., 2021a,b) has been developed as a decentralized framework for organizational learners (with rich data and computation resources) to autonomously assist each other without sharing data, task labels, or models. Consequently, each learner also obtains a "personalized model" to serve its own task. A critical difference between AL and FL is that participating learners in AL do not share task labels or a global model to meet organizational model privacy Wang et al. (2021). Also, since a learner is resource-rich, AL often focuses on reducing the assistance rounds instead of communication costs at each round. As such, the techniques developed under AL are entirely different from those in the personalized FL literature.

D General parametric models

In a more general parametric model setting, the nature of the personalized FL problem remains the same as the simple case in Subsection 2.1. Suppose that there are M clients, and their data are assumed to be generated from:

$$\theta_m \mid \theta_0 \overset{\text{IID}}{\sim} \mathcal{N}(\theta_0, \sigma_0^2),\ w_{m,i} \overset{\text{IID}}{\sim} p_m(\cdot \mid \theta_m), i \in [N_m].$$

For simplicity, we suppose that each observation $w_{m,i}$ is a scalar. The related discussions can be easily extended to multivariate settings. Compared with Eqn. (1), each client may have a different amount of data and follow client-specific distributions (denoted by p_m). Let $z_m \triangleq \theta_m^{(\text{L})}$ denote the minimum of the negative log-likelihood function $L_m(\theta_m) = \sum_{i=1}^{N_m} \text{loss}(w_{m,i}, \theta_m)$. Standard asymptotic statistics for M-estimators (Van der Vaart, 2000, Ch.5) under regularity conditions show that $\sqrt{N_m}(z_m - \theta_m) \to_d \mathcal{N}(0, v_m^2)$ as $N_m \to \infty$, with a constant v_m^2 (the inverse Fisher information). Thus, we approximately have:

$$z_m \mid \theta_m \sim \mathcal{N}(\theta_m, N_m^{-1} v_m^2).$$

In other words, we may treat the statistic $\theta_m^{(L)}$ as the "data" z_m in the two-level Gaussian model in Subsections 2.1. Letting $\sigma_m^2 \triangleq N_m^{-1} v_m^2$ and taking it into the equations in Subsection 2.1, we can approximate posterior quantities accordingly. Also, the objective function $L_m(\theta_m)$ is approximated by its second-order Taylor expansion in the form of Eqn. (4). It is worth mentioning that the asymptotic results for parametric models may not hold for neural networks.

In particular, we let:

$$\theta_1^{(\mathrm{FL})} \triangleq \frac{N_1 v_1^{-2}\theta_1^{(\mathrm{L})} + \sum_{m\neq 1}(\sigma_0^2 + N_m^{-1}v_m^2)^{-1}\theta_m^{(\mathrm{L})}}{N_1 v_1^{-2} + \sum_{m\neq 1}(\sigma_0^2 + N_m^{-1}v_m^2)^{-1}},$$
$$\theta^{(\mathrm{G})} \triangleq \frac{\sum_{m\in[M]}(\sigma_0^2 + N_m^{-1}v_m^2)^{-1}\theta_m^{(\mathrm{L})}}{\sum_{m\in[M]}(\sigma_0^2 + N_m^{-1}v_m^2)^{-1}}.$$

Each client, say client 1, can obtain its personal optimal solution $\theta_1^{(\mathrm{FL})}$ through the following initial value and optimization parameters η, ℓ.

$$\theta_1^{\mathrm{INIT}} = \frac{\sum_{m\neq 1}(\sigma_0^2 + N_m^{-1}v_m^2)^{-1}\theta_m^{(\mathrm{L})}}{\sum_{m\neq 1}(\sigma_0^2 + N_m^{-1}v_m^2)^{-1}}, \tag{15}$$

$$(1 - N_1 v_1^{-2}\eta)^{\ell} = \frac{\sum_{m\neq 1}(\sigma_0^2 + N_m^{-1}v_m^2)^{-1}}{N_1 v_1^{-2} + \sum_{m\neq 1}(\sigma_0^2 + N_m^{-1}v_m^2)^{-1}}. \tag{16}$$

Remark D.1 (**Interpretations of the choice of η and ℓ**). Let us consider the following extreme cases.

• (Large sample size, few clients) Suppose that $N_1 = \cdots = N_M \gg M > 1$, and v_1^2, σ_0^2 are comparable. Then, for client 1, Eqn. (16) becomes:

$$(1 - N_1 v_1^{-2}\eta)^{\ell} \approx \frac{(M-1)\sigma_0^{-2}}{N_1 v_1^{-2} + (M-1)\sigma_0^{-2}} \approx \frac{M}{N_1}\frac{v_1^2}{\sigma_0^2},$$

which implies that client 1 needs to run aggressively with its local data. Also, the smaller $\sigma_1^2 = v_1^2/N_1$ (namely better data quality or larger sample size), the more local training steps is favored.

• (Small sample size, many clients) Suppose that $1 \ll N_1 = \cdots = N_M \ll M$, and $v_1^2 = \cdots = v_M^2$. We have:

$$\begin{aligned}(1 - N_1 v_1^{-2}\eta)^{\ell} &\approx \frac{(M-1)(\sigma_0^2 + N_1^{-1}v_1^2)^{-1}}{N_1 v_1^{-2} + (M-1)(\sigma_0^2 + N_1^{-1}v_1^2)^{-1}} \\ &\approx 1 - \frac{N_1}{M}\frac{\sigma_0^2}{v_1^2},\end{aligned} \tag{17}$$

which implies that client 1 needs to set:

$$N_1 v_1^{-2}\eta\cdot\ell \approx \frac{N_1}{M}\frac{\sigma_0^2}{v_1^2}, \quad \text{or} \quad \eta\cdot\ell \approx \frac{\sigma_0^2}{M}.$$

It is interesting to see that the choice of η, ℓ is independent of the client in this scenario.

• (Homogeneous clients) Suppose that $\sigma_0^2 = 0$, meaning no discrepancy between clients' data distributions. We have

$$(1 - N_1 v_1^{-2}\eta)^{\ell} = \frac{\sum_{m\neq 1} N_m v_m^{-2}}{\sum_{m\in[M]} N_m v_m^{-2}}.$$

If we further assume that $v_1 = \cdots = v_M$, M is large, and $N_1, \ldots, N_M$ are at the same order, client 1 needs to set $N_1 v_1^{-2}\eta\cdot\ell \approx N_1/N$, where $N \triangleq N_1 + \cdots + N_M$. In other words, $\eta\cdot\ell \approx v_1^2/N$, which is the same among all clients.

E Implementation Details of Algorithm 1

In Algorithm 1, we need to evaluate the empirical variance of a set of models. This can be a memory concern for edge devices in practice. We use the following way to perform online calculation, so that the required hardware memory does not grow with the number of rounds or clients.

To calculate the sample variance

$$\widehat{\sigma_t^2} \triangleq \frac{1}{t}\sum_{i\in[t]}(x_i - \bar{x}_t)^2 \tag{18}$$

where $\bar{x}_t \triangleq t^{-1}\sum_{i\in[t]} x_i$, we do not need to store $x_1, \ldots, x_t$. Instead, we only need to store $\bar{x}_t$ at time step t, and calculate the sample variance in the following recursive way.

$$\begin{aligned}
&\text{For } t = 1, 2, \ldots \\
&\bar{x}_t = \frac{t-1}{t}\bar{x}_{t-1} + \frac{1}{t}x_t, \\
&\widehat{\sigma_t^2} = \frac{t-1}{t}\widehat{\sigma_{t-1}^2} + \frac{t-1}{t}(\bar{x}_t - \bar{x}_{t-1}) + \frac{(x_t - \bar{x}_t)^2}{t},
\end{aligned} \tag{19}$$

with $\bar{x}_0 = \widehat{\sigma_0^2} = 0$. It can be verified that the $\widehat{\sigma_t^2}$ in (19) is equivalent to that in (18). The recursive computation is particularly favorable for large neural networks with millions of parameters and small hardware memory.

Remark E.1 (Client sampling). Suppose that in each round, only a subset of clients is activated. The subscript t in the online update of σ_m^2 is client m's the local time counter, namely the total counts that client m is activated in FL communications. This local time counter shall be distinguished from the global counter t that indicates the system communication round. Also, a client's intra-client uncertainty is only online updated in those rounds that it is activated.

F An Alternative Version of Algorithm 1

In this section, we present a slightly more complicated version of Self-FL in Algorithm 2, as an alternative of Algorithm 1. Its pseudocode can be found at the end of the Appendix, and the main changes are highlighted with blue fonts. The motivation goes back to the discussions in Subsection 3.2, where we mentioned the use of a client m's local parameter at each round as a bootstrapped estimate of the underlying $\theta_m^{(\mathrm{L})}$. In Algorithm 2, each client needs to run additional local training until convergence (denoted by $\theta_{m,t}$) in each FL round and communicates two local models with the server.

This alternative variant requires the computation and communication of local-optimal solutions ($\theta_{m,t}$). Thus, the implementation of Self-FL in Algorithm 2 incurs higher computation costs for local clients (due to additional local fine-tuning) and doubles the communication overhead compared with the standard FedAvg (McMahan et al., 2017). We will perform ablation studies in Subsection G.2 to compare these two algorithms.

G Additional Experiments

We perform extended experiments of Self-FL and summarize the results in this section. Particularly, in Subsection G.2, we compare the performance of two implementation variants of Self-FL. In Subsection G.4, we revisit the audio data in a client sampling setting, as a supplement to the full participation setting studied in Subsection 4.3. In Subsections G.5 and G.6, we show the impact of the client activity rate $C \in (0, 1]$ and provide insights into the adaptive local training steps of Self-FL, respectively.

G.1 Convergence study on synthetic data

For FedAvg on the synthetic dataset, we use a gradient descent-based local update rule: $\theta_1^l = \theta_1^{l-1} - \eta(\theta_1^{l-1} - \theta_1^{(L)})/\sigma_1^2$. For DITTO, an additional regularization term is used and the update rule is: $\theta_1^l = \theta_1^{l-1} - \eta(\theta_1^{l-1} - \theta_1^{(L)})/\sigma_1^2 - \lambda(\theta_1^{l-1} - \theta^{(G)})$. A skeptical reader may wonder whether DITTO is equivalent to Self-FL for a particular choice of λ. This is not the case, even in our earlier example of the two-layer Gaussian model. The two methods differ in the aggregation rule and the initialization of each client.

Case 1: Homogeneous setting. We use a small inter-client uncertainty in this setup and assume that clients have similar local sample sizes. There are $M = 20$ clients in the FL system where the local sample size N_m is uniformly sampled from the range $[10, 20]$. We set the parent parameter $\theta_0 = 1.6$. The inter-client and intra-client uncertainty are $\sigma_0^2 = 0.001$, $\sigma_m^2 = 0.1$, respectively. The experiment is repeated 200 times, and we report the mean value as the final metric. The standard error of the experiment is within 0.001. Note that the effective σ_m^2 shall be divided by N_m.

Figure 3 compares the parameter estimation performance of Self-FL algorithm with FedAvg and three existing personalized FL techniques: DITTO (Li et al., 2021), pFedMe (Dinh et al., 2020), and

Algorithm 2 Self-Aware Personal Federated Learning by Introducing Local Minimum $\theta_{m,t}$ (Augmented Self-FL).
The main differences with Algorithm 1 are highlighted with blue fonts.

Input: A server and M clients. System-wide objects, including the initial model parameters $\theta_m^0 = \theta_{m,0} = \theta^0$, initial uncertainty of clients' parameters $\widehat{\sigma_0^2} = 0$, the activity rate C, the number of communication rounds T, server batch size B_s, and client batch size B_m, and the loss function $(z, \theta) \mapsto \text{loss}(z, \theta)$ corresponding to a common model architecture. Each client m's local resources, including N_m data observations, parameter θ_m, number of local training steps ℓ_m, and learning rate η_m.

System executes:

for each communication round $t = 1, \dots T$ **do**
  $\mathbb{M}_t \leftarrow \max(\lfloor C \cdot M \rfloor, 1)$ active clients uniformly sampled without replacement
  for each client $m \in \mathbb{M}_t$ **in parallel do**
    Distribute server model parameters θ^{t-1} to local client m
    $(\theta_m^t, \theta_{m,t}, \widehat{\sigma_m^2}) \leftarrow$ **ClientUpdate**$(z_{m,1}, \dots, z_{m,N_m}, \theta^{t-1}, \widehat{\sigma_0^2})$
  $(\theta^t, \widehat{\sigma_0^2}) \leftarrow$ **ServerUpdate**$(\theta_m^t, \theta_{m,t}, \widehat{\sigma_m^2}, m \in \mathbb{M}_t)$

ClientUpdate $(z_{m,1}, \dots, z_{m,N_m}, \theta^{t-1}, \widehat{\sigma_0^2})$**:**

Calculate the uncertainty of the local optimal solution $\theta_m^{(\text{L})}$ of client m:

$$\widehat{\sigma_m^2} = \text{empirical variance of } \{\theta_{m,1}, \dots, \theta_{m,t-1}\}. \tag{20}$$

Calculate the initial parameter

$$\theta_m^t \triangleq \theta^{t-1} - \frac{(\widehat{\sigma_0^2} + \widehat{\sigma_m^2})^{-1}}{\sum_{k:k\neq m}(\widehat{\sigma_0^2} + \widehat{\sigma_k^2})^{-1}}(\theta_{m,t-1} - \theta^{t-1}), \tag{21}$$

Choose $\ell_m \geq 1$ according to

$$\left(1 - \frac{\eta_m}{B_m \widehat{\sigma_m^2}}\right)^{\ell_m} = \frac{\sum_{k:k\neq m}(\widehat{\sigma_0^2} + \widehat{\sigma_k^2})^{-1}}{1/\widehat{\sigma_m^2} + \sum_{k:k\neq m}(\widehat{\sigma_0^2} + \widehat{\sigma_k^2})^{-1}}$$

for local step s from 1 to ℓ_m **do**
  Sample a batch $I \subset [N_m]$ of size B_m
  Update $\theta_m^t \leftarrow \theta_m^t - \eta_m \nabla_\theta \sum_{i\in I} \text{loss}(w_{m,i}, \theta_m^t)$,
Let $\theta_{m,t} \leftarrow \theta_m^t$.
for local step s from $\ell_m + 1$ to ∞ **do**
  Sample a batch $I_s \subset [N_m]$ of size B_m
  Update $\theta_{m,t} \leftarrow \theta_{m,t} - \eta_m \nabla_\theta \sum_{i\in I_s} \text{loss}(w_{m,i}, \theta_{m,t})$,
  If it converges, then Break.
Store $\theta_{m,t}$ locally
Return $(\theta_m^t, \theta_{m,t}, \widehat{\sigma_m^2})$ and send them to the server

ServerUpdate $(\theta_m^t, \theta_{m,t}, \widehat{\sigma_m^2}, m \in \mathbb{M}_t)$**:**

Calculate the uncertainty of clients' underlying parameters

$$\widehat{\sigma_0^2} = \text{empirical variance of } \{\theta_{m,t}, m \in \mathbb{M}_t\}. \tag{22}$$

Calculate model parameters

$$\theta^t \triangleq \frac{\sum_{m\in\mathbb{M}_t}(\widehat{\sigma_0^2} + \widehat{\sigma_m^2})^{-1}\theta_{m,t}}{\sum_{m\in\mathbb{M}_t}(\widehat{\sigma_0^2} + \widehat{\sigma_m^2})^{-1}}. \tag{23}$$

Return $(\theta^t, \widehat{\sigma_0^2})$

PerFedAvg (Fallah et al., 2020b). The estimation error is measured in ℓ_1 norm. All FL algorithms use the same learning rate 10^{-4} and are trained for 200 rounds. For Self-FL, the local training steps l_m's are computed using Eqn. (16). For other FL methods, the local training steps are set to 50.

We consider the activity rate $C \in \{0.1, 0.2, 0.5, 0.8, 1\}$ and compare different algorithms in each setting. We draw two conclusions from Figure 3. (i) When there are sufficient active clients in each communication round, Self-FL and PerFedAvg achieve the lowest estimation error for both global and local parameters compared with others. (ii) When the number of active clients is small, Self-FL outperforms the other methods.

Case 2: Heterogeneous setting. To generate heterogeneous data across clients, we use a large value for inter-client uncertainty σ_0^2 and assume that clients have different scales of local samples size N_m. We consider $\sigma_0^2 = 1$ and $\sigma_m^2 = 0.1$. The sample size of each client is uniformly drawn from $[10, 200]$ (which is much broader than that in Case 1). Other settings are the same as Case 1.

Recall that if the inter-client uncertainty is larger, the clients' data are more irrelevant, and the FL posterior approaches the local posterior. Then, the client learns on its own. The global model becomes a simple equal-weighted averaging of local models (detailed in Remark 2.1). We show the performance comparison results of different FL algorithms on the heterogeneous dataset in Figure 4. One can see from the figure that Self-FL's estimation of both the global parameter and the local ones are insensitive to the client participation ratio.

G.2 Ablation study of Self-FL's performance with Algorithms 1 and 2

Results on synthetic data. In this experiment, we conduct an ablation study of Self-FL's performance with two different implementation variants outlined in Algorithm 1 (using the early-stop parameter θ_m^t) and Algorithm 2 (using the sufficiently trained local parameter $\theta_{m,t}$). We use the same synthetic datasets and learning parameters as Section 4. We repeat the experiments for 200 times and report the mean value. The standard error is kept within 0.001. The results are visualized in Figure 5. One can see from the comparison that the performance gap between these two variants is negligible on both homogeneous and heterogeneous data.

Results on MNIST and FEMNIST images. Figure 6 shows the evaluation results, where the y-axis denotes the weighted training accuracy of all personalized models. The two curves in Figure 6b are very close and the accuracy difference is small than 1%.

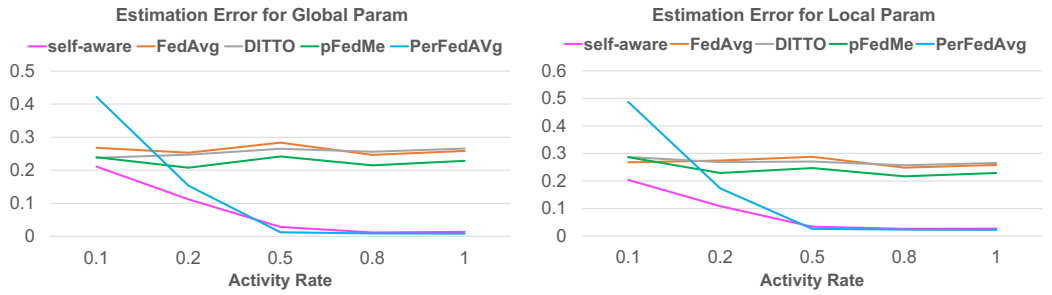
Results on the Amazon Alexa audio dataset. We also perform ablation experiments to assess the performance of two versions of Self-FL on the wake-word audio dataset. The results in Table 4 show that their performance is similar.

Table 4: Detection performance (relative FA) of the *global model* on the test dataset of wake-word audio data. The devices are in the normal state.

FL methods	Device Types				
	A	B	C	D	E
Self-FL (Algorithm 2)	0.93	0.94	0.96	0.89	0.98
Self-FL (Algorithm 1)	0.92	0.94	0.91	0.91	1.01

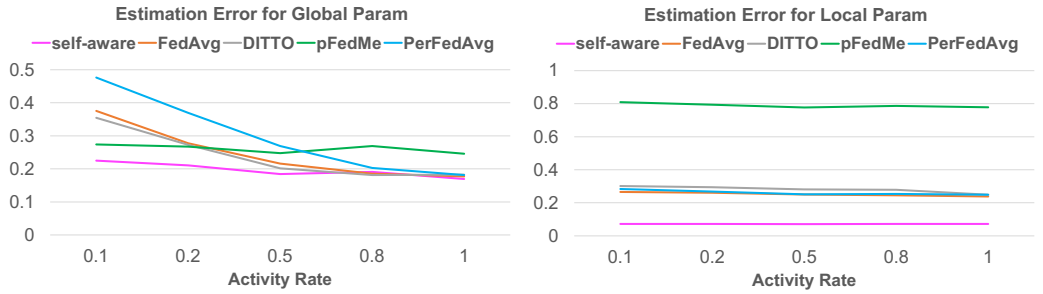
G.3 Experiments on Sent140 Dataset

Sent140 Go et al. (2009) is a text sentiment classification dataset. We generate non-i.i.d. FL training data of Sent140 following the same procedure as FedProx Li (2020). As for the ML model, we



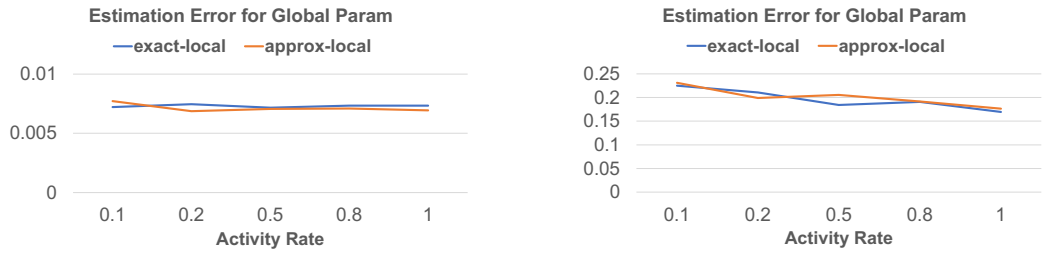
(a) (b)

Figure 3: Performance of different FL algorithms on synthetic data of high homogeneity. Estimations errors of the global and local parameters (averaged over clients) are shown in (a) and (b), respectively.



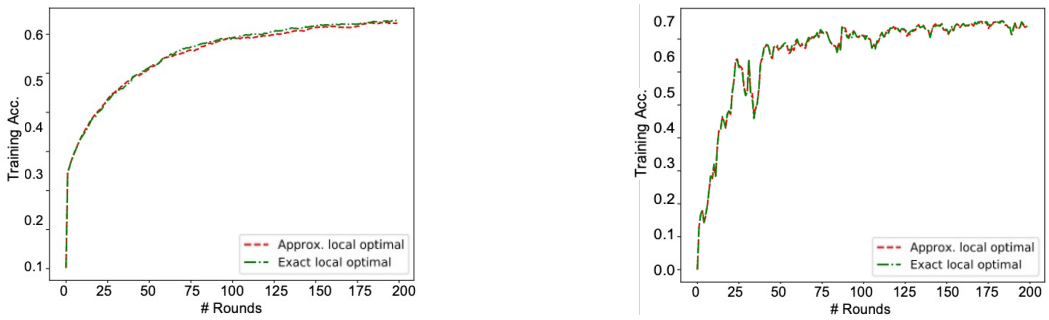
(a) (b)

Figure 4: Performance comparison on synthetic datasets of high heterogeneity. Estimation errors are shown for global parameter (a) and local parameter (b).



(a) Homogeneous data (b) Heterogeneous data

Figure 5: Performance comparison of Algorithms 1 and 2 on synthetic data.

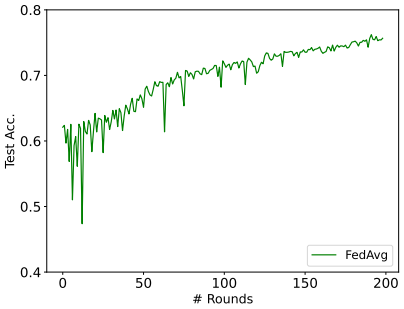


(a) MNIST (b) FEMNIST

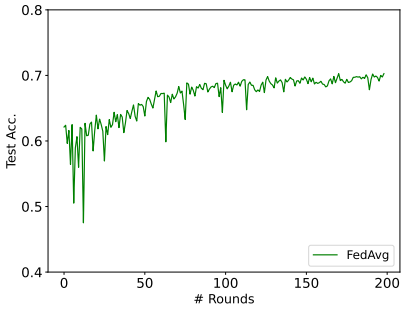
Figure 6: Performance comparison of Algorithms 1 and 2 on the MNIST and FEMNIST image data.

use a two-layer LSTM binary classifier for this experiment. The model has two LSTM layers with 256 hidden units, and the last layer is a fully-connected layer. We use the pre-trained 300D GloVe embedding Pennington et al. (2014) to transform the input sequence of 25 characters into the embedding space. For each FL algorithm, we train the local models with a learning rate of 0.3, a batch size of 10, and randomly sub-sample 77 clients in each communication round (corresponding to a client activity rate $C = 0.1$). Each selected local client trains his model for 2 epochs. This hyper-parameter configuration is suggested in the previous paper Li et al. (2020b). Since the number of local epochs is small, we skip Self-FL's computation of the local training steps l_m and use the same local epochs $l_m = 2$.

We use *warm start* in the Sent140 experiment. Particularly, we train a global model from scratch with FedAvg for 200 rounds and use it as the initialization of the global model for all the other FL algorithms. With this warm start, we continue training the global model with various FL algorithms for another 400 rounds. Figure 7 shows the training and test accuracy of the local model obtained by FedAvg McMahan et al. (2017). We report the weighted accuracy across all users in the FL system where the weight ratio of each user is proportional to his local sample size.



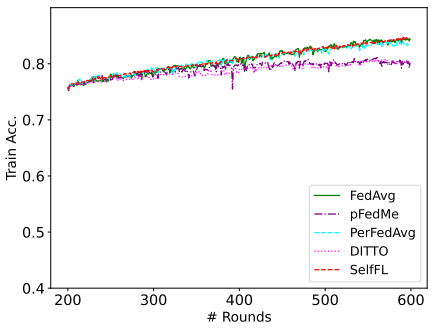
(a) Training accuracy.



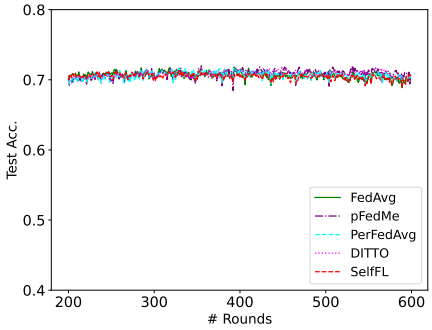
(b) Test accuracy.

Figure 7: Performance of FL training from scratch using FedAvg (*warm-start* stage).

With the pre-trained global model from FedAvg (at round 200) as the 'warm-start' initialization, we continue FL training with different algorithms for another 400 rounds. Figure 8 compares the training and test accuracy of the personalized/local models obtained by different FL algorithms. The accuracy is aggregated across clients where the aggregation weight is proportional to the local sample size. Note that Figure 8 shows the accuracy change from round 200 (where warm start ends) to round 600. We can see from Figure 8a that both Self-FL and FedAvg demonstrates better convergence performance compared to DITTO Li et al. (2021), pFedMe Dinh et al. (2020), and PerFedAvg Fallah et al. (2020b). Recall that the first 200 rounds are trained with FedAvg and continued by other FL algorithms. As the result, the test accuracy in Figure 8b does not change much from round 200 to round 600 since it 'saturates' during the warm-start stage (Figure 7b).



(a) Training accuracy.



(b) Test accuracy.

Figure 8: Performance comparison of different FL algorithms on Sent140 text data. Note that the x-axis starts from round 200 since we use warm-start in this experiment.

G.4 Self-FL's performance on wake-word data in clients selection setting

In this experiment, we evaluate the performance of FL algorithms in a client selection scenario. More specifically, we assume there are 4 clients for each of the five device types (A$\sim$E). Each client has a uniform partition of the training set for a specific device type. The FL system has a total of 20 clients. We set the activity rate to $C = 0.25$.

We consider two variants of Self-FL in the clients selection scenario: (i) Client-level variant. This implementation aggregates local models from the currently active clients in each round. (ii) Cluster-level variant. This method groups clients into clusters and then perform aggregation at the cluster level. In our experiments, we cluster clients using device types. We can also use the intra-client uncertainty (σ_m^2) as the clustering criteria since this information is available for the server (Algorithm 1). With the technique of clients clustering, the server stores the latest model parameter for each cluster as its 'representative'. In each communication round, the server loads the weight for each non-active cluster. The global model is updated by aggregating across all clusters. It is worth noting that storing the latest model for each cluster is much more scalable than saving the model separately for each client. The performance of the above two variants is shown in Table 5. We can see that the cluster-level aggregation obtains more conservative global models, and its performance improvement is more

Table 5: Detection performance (relative FA) of the global model obtained using two variants of Self-FL on test data.

FL methods	State	Device Type				
		A	**B**	**C**	**D**	**E**
Self-FL (client-level)	normal	0.91	0.92	0.97	0.91	0.99
	playback	0.93	0.96	1.80	1.00	1.00
	alarm	NA	0.89	NA	1.04	1.00
Self-FL (cluster-level)	normal	0.96	0.95	1.00	0.91	0.97
	playback	0.86	0.99	1.00	0.97	1.01
	alarm	NA	0.89	NA	1.00	1.00

stable across different device types and operation states compared to the client-level variant. This observation is intuitive, because keeping track of the latest weights for each cluster representative for aggregation can prevent the global model from catastrophic forgetting when only a small subset of clients are active in each round.

G.5 Impact of the activity rate C

We use the client activity rate as the weight for smoothing the average in the above experiments. While this yields a more stable update, it could cause a slow convergence. To investigate the impact of the weights in our smoothing average update for the global model, we set the activity rate to 0.02 and compare the performance of Self-FL with FedAvg and DITTO. Figure 9 shows the empirical results in this sparse client participation case. One can see that Self-FL converges slower than FedAvg and DITTO when we use the activity rate (0.02) as the weight for the current estimation of the global model in smoothing average. This is expected since a small weight for $\hat{\theta}^t$ means that the update of the global model is more conservative.

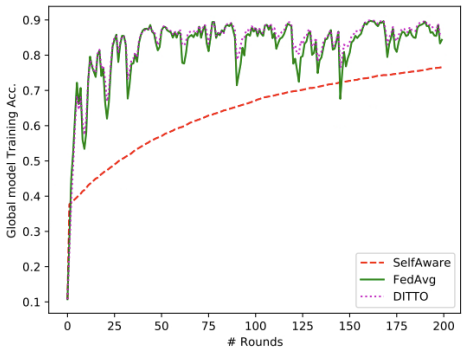


Figure 9: Convergence performance of different FL algorithms on the MNIST dataset when the activity rate is set to 0.02.

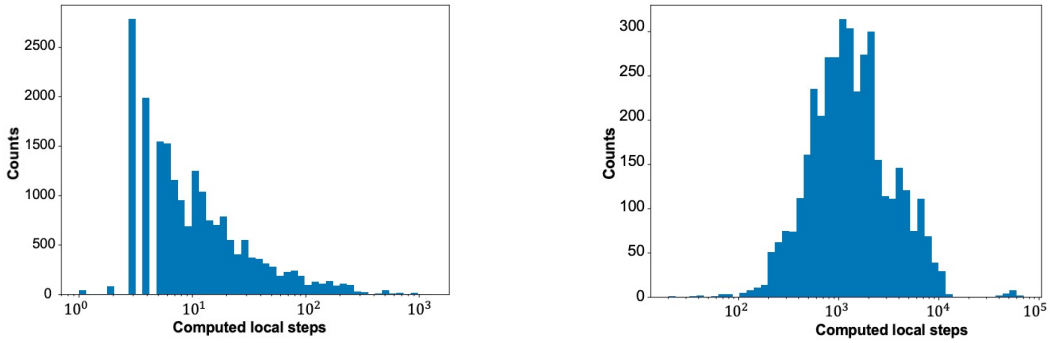
We also perform an ablation study on the wake-word dataset to investigate the impact of the activity rate. The results are summarized in Table 6. We use the client-level implementation of Self-FL in Subsection G.4 in this experiments. one can see that Self-FL framework obtains better wake-word detection performance (i.e., smaller relative FA values) when the activity rate is not too small.

G.6 Study of Self-FL's adaptive local training

A key component of Self-FL is the data-adaptive local training procedure as discussed in Section 3. The local optimization trajectory is designed in such a way that it achieves a good balance between local model improvement and global model update. We study the theoretical values of local training steps l_m computed using Eqn. (8). Figure 10 shows the histogram of l_m of the active clients in round $t = 100$. We also visualize the dynamics of the local training steps for a randomly selected client across FL rounds in Figure 11. The value of l_m exists for a subset of rounds since we use the activity rate $C = 0.1$. We can see that the computed number of steps l_m has an overall decreasing trend after round 150, suggesting a drop of the intra-client uncertainty σ_m^2. In other words, the quality of personalized model θ_m^t improves eventually.

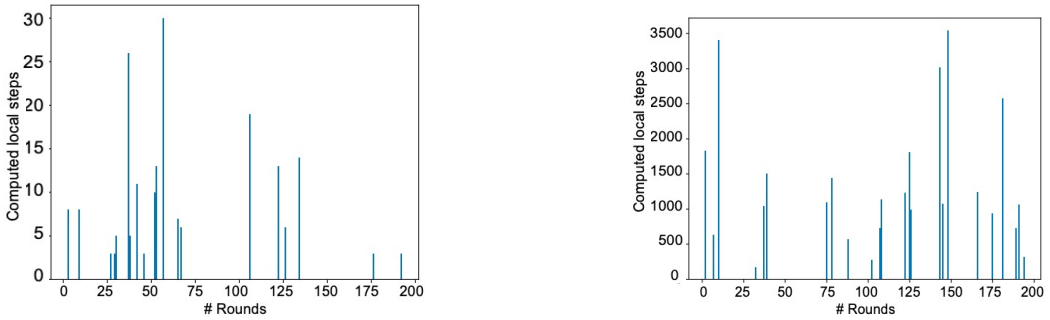
Table 6: Self-FL's performance with two different activity rates on the wake-word dataset. The relative FA values of the updated global models are reported.

Activity Rate	State	Device Types				
		A	B	C	D	E
C = 0.25	normal	**0.91**	0.92	0.97	0.91	1.00
	playback	0.93	0.96	1.80	1.00	1.00
	alarm	NA	0.89	NA	1.04	1.00
C = 0.1	normal	0.95	0.95	1.00	0.91	0.97
	playback	0.86	0.97	1.00	0.96	1.01
	alarm	NA	0.89	NA	1.00	1.00



(a) MNIST (b) FEMNIST

Figure 10: Histogram of the computed number of local training steps for active clients at the round $t = 100$.



(a) MNIST (b) FEMNIST

Figure 11: Trajectory of the computed local training steps for a randomly selected client in different FL rounds.

G.7 Limitations of Self-FL and Future Directions

There are some new problems left from the work that deserve further research. First, in many practical applications, clients only have unlabeled data, and it is not easy to annotate those data. This poses a significant challenge in training personalized models and evaluating them. An important problem is to integrate self-supervised FL techniques Diao et al. (2021c) into personalization to address such a lack-of-label issue. Second, we considered heterogeneity in terms of clients' data distributions. There are other aspects of heterogeneity, e.g., those at the system and model levels. A more integrated view of FL heterogeneity is lacking. Third, a critical problem in personalized FL is to assess whether any particular client model, for a given set of local data, has achieved its theoretical limit from, e.g., a goodness-of-fit perspective Zhang et al. (2021). We refer to Ding et al. (2022) for an outlook on other challenges related to personalized FL.

H Proof of Proposition 3.1

In this section, we provide detailed proof of Proposition 3.1 introduced in the main paper. For brevity, we write $\sum_{k\in[M]}$ and $\sum_{k\in[M]-\{m\}}$ as $\sum_k$ and $\sum_{k:k\neq m}$, respectively. We define $\sum_{i,j:\, i\neq j}$ similarly.

Let $w_m = (\sigma_m^2 + \sigma_0^2)^{-1}$. According to the assumption, let c_v be an upper bound of $\max_{m\in[M]} \sigma_m^2$. Then we have

$$w_{\min} \triangleq (c_v + \sigma_0^2)^{-1} \leq \min_{m\in[M]} w_m \leq \max_{m\in[M]} w_m \leq \sigma_0^{-2} \triangleq w_{\max}. \tag{24}$$

Let $v = \max_{m\in[M]}(w_m / \sum_k w_k)$. It follows from (24) that

$$v = O(M^{-1}) \tag{25}$$

for large M. Let

$$\zeta_m^t \triangleq \hat{\theta}_m^t - \theta_m^{(\text{FL})}, \quad \forall m \in [M], t = 0, 1, \ldots \tag{26}$$

Recall that by the assumption, C is a constant upper bound of $\max_{m\in[M]} |\zeta_m^0|$. Next, we will provide a uniform bound on $|\zeta_m^t|$, by considering every $t \geq 1$.

According to the update rule, as shown in equation (6), we have

$$\theta_m^t = \frac{\sigma_m^{-2}}{\sigma_m^{-2} + \sum_{k:\, k\neq m} w_k} \theta_m^{(\text{L})} + \frac{\sum_{k:\, k\neq m} w_k}{\sigma_m^{-2} + \sum_{k:\, k\neq m} w_k} \theta_m^{t,\text{INIT}}, \tag{27}$$

where $\theta_m^{t,\text{INIT}}$ is the initial parameter of client m at time t, introduced in (10). Recall from (3) that

$$\theta_m^{(\text{FL})} = \frac{\sigma_m^{-2}\theta_m^{(\text{L})} + \sum_{k:\, k\neq m} w_k \theta_k^{(\text{L})}}{\sigma_m^{-2} + \sum_{k:\, k\neq m} w_k}. \tag{28}$$

Combining (27) and (28), and invoking the definition in (12), we have

$$|\zeta_m^t| = |\theta_m^t - \theta_m^{(\text{FL})}| = \left| \frac{\sum_{k:\, k\neq m} w_k}{\sigma_m^{-2} + \sum_{k:\, k\neq m} w_k} \left(\theta_m^{t,\text{INIT}} - \frac{\sum_{k:\, k\neq m} w_k \theta_k^{(\text{L})}}{\sum_{k:\, k\neq m} w_k} \right) \right| \tag{29}$$

$$\leq q \left| \theta_m^{t,\text{INIT}} - \frac{\sum_{k:\, k\neq m} w_k \theta_k^{(\text{L})}}{\sum_{k:\, k\neq m} w_k} \right| \tag{30}$$

By the definition of $\theta_m^{t,\text{INIT}}$, we may rewrite it as

$$\theta_m^{t,\text{INIT}} = \theta^{t-1} - \frac{(\sigma_0^2 + \sigma_m^2)^{-1}}{\sum_{k:\, k\neq m} (\sigma_0^2 + \sigma_k^2)^{-1}} (\theta_m^{t-1} - \theta^{t-1}) = \frac{\sum_{k:\, k\neq m} w_k \theta_m^{t-1}}{\sum_{k:\, k\neq m} w_k}. \tag{31}$$

Therefore, with the definition in (26), we have

$$\theta_m^{t,\text{INIT}} - \frac{\sum_{k:\, k\neq m} w_k \theta_k^{(\text{L})}}{\sum_{k:\, k\neq m} w_k} = \frac{\sum_{k:\, k\neq m} w_k (\theta_k^{t-1} - \theta_k^{(\text{L})})}{\sum_{k:\, k\neq m} w_k} \tag{32}$$

$$= \frac{\sum_{k:\, k\neq m} w_k (\zeta_k^{t-1} + \theta_k^{(\text{FL})} - \theta_k^{(\text{L})})}{\sum_{k:\, k\neq m} w_k} \tag{33}$$

$$= \frac{\sum_{k:\, k\neq m} w_k \zeta_k^{t-1}}{\sum_{k:\, k\neq m} w_k} + \frac{\sum_{k:\, k\neq m} w_k \theta_k^{(\text{FL})}}{\sum_{k:\, k\neq m} w_k} - \frac{\sum_{k:\, k\neq m} w_k \theta_k^{(\text{L})}}{\sum_{k:\, k\neq m} w_k}. \tag{34}$$

Meanwhile, it can be verified that

$$\frac{\sum_{k:\, k\neq m} w_k \theta_k^{(\text{FL})}}{\sum_{k:\, k\neq m} w_k} = \frac{\sum_k w_k \theta_k^{(\text{FL})}}{\sum_k w_k} (1 - v_m)^{-1} - \frac{v_m}{1 - v_m} \theta_m^{(\text{FL})}, \quad \text{where } v_m \triangleq \frac{w_m}{\sum_k w_k} \leq v, \tag{35}$$

$$\frac{\sum_{k:\, k\neq m} w_k \theta_k^{(\text{L})}}{\sum_{k:\, k\neq m} w_k} = \frac{\sum_k w_k \theta_k^{(\text{L})}}{\sum_k w_k} (1 - v_m)^{-1} - \frac{v_m}{1 - v_m} \theta_m^{(\text{L})}, \tag{36}$$

which further implies that

$$\left|\frac{\sum_{k:\,k\neq m} w_k\theta_k^{(\mathrm{FL})}}{\sum_{k:\,k\neq m} w_k} - \frac{\sum_{k:\,k\neq m} w_k\theta_k^{(\mathrm{L})}}{\sum_{k:\,k\neq m} w_k}\right| \leq \frac{1}{1-v}|D| + \frac{2vc_\theta}{1-v}, \tag{37}$$

$$\text{where } D \triangleq \frac{\sum_k w_k(\theta_k^{(\mathrm{FL})} - \theta_k^{(\mathrm{L})})}{\sum_k w_k}, \tag{38}$$

and c_θ is an upper bound of $|\theta_m^{(\mathrm{FL})}|$ and $|\theta_m^{(\mathrm{L})}|$. The definition of $\theta_m^{(\mathrm{FL})}$ implies that $|\theta_k^{(\mathrm{FL})}| \leq \max_{m\in[M]}|\theta_m^{(\mathrm{FL})}|$ for all $k \in [M]$. Since $\theta_m^{(\mathrm{L})}$'s are independent Gaussian with variance bounded by c_v, we have $\max_{m\in[M]}|\theta_m^{(\mathrm{FL})}| = O_p(\sqrt{\log M})$. Thus, we may choose

$$c_\theta = O_p(\sqrt{\log M}). \tag{39}$$

Taking (34) and (37) into (30), we obtain

$$|\zeta_m^t| \leq q \max_{m\in[M]}\left|\frac{\sum_{k:\,k\neq m} w_k\zeta_k^{t-1}}{\sum_{k:\,k\neq m} w_k}\right| + |D| \leq q \max_{m\in[M]}|\zeta_m^{t-1}| + \frac{|D| + 2vc_\theta}{1-v}, \quad \forall m \in [M]. \tag{40}$$

It follows from (40) that

$$\max_{m\in[M]}|\zeta_m^t| \leq q^t \max_{m\in[M]}|\zeta_m^0| + \frac{|D| + 2vc_\theta}{1-v}. \tag{41}$$

Next, we bound $|D|$. Recall that $\mathbb{E}(\theta_i^{(\mathrm{FL})}) = \theta_0$ and $\mathrm{Cov}(\theta_i^{(\mathrm{FL})}, \theta_j^{(\mathrm{FL})}) = 0$ for all $i, j \in [M]$ and $i \neq j$. Since

$$D = \frac{\sum_m w_m(\theta_m^{(\mathrm{FL})} - \theta_m^{(\mathrm{L})})}{\sum_m w_m}, \tag{42}$$

$$\theta_m^{(\mathrm{FL})} - \theta_m^{(\mathrm{L})} = \frac{\sum_{k:\,k\neq m} w_k\theta_k^{(\mathrm{L})}}{\sigma_m^{-2} + \sum_{k:\,k\neq m} w_k} - \frac{\sum_{k:\,k\neq m} w_k}{\sigma_m^{-2} + \sum_{k:\,k\neq m} w_k}\theta_m^{(\mathrm{L})} = \frac{\sum_{k:\,k\neq m} w_k(\theta_k^{(\mathrm{L})} - \theta_m^{(\mathrm{L})})}{\sigma_m^{-2} + \sum_{k:\,k\neq m} w_k} \tag{43}$$

$$\mathbb{V}(\theta_m^{(\mathrm{FL})} - \theta_m^{(\mathrm{L})}) \leq 2\mathbb{V}\left(\frac{\sum_{k:\,k\neq m} w_k\theta_k^{(\mathrm{L})}}{\sigma_m^{-2} + \sum_{k:\,k\neq m} w_k}\right) + 2\mathbb{V}\left(\frac{\sum_{k:\,k\neq m} w_k}{\sigma_m^{-2} + \sum_{k:\,k\neq m} w_k}\theta_m^{(\mathrm{L})}\right) \leq 4\max_k \mathbb{V}(\theta_k^{(\mathrm{L})}) = 4V, \tag{44}$$

we have $\mathbb{E}(D) = 0$, and

$$\mathbb{V}(D) = \frac{\sum_m w_m^2\,\mathbb{V}(\theta_m^{(\mathrm{FL})} - \theta_m^{(\mathrm{L})})}{(\sum_m w_m)^2} + \frac{\sum_{i,j:\,i\neq j} w_iw_j\mathrm{Cov}(\theta_i^{(\mathrm{FL})} - \theta_i^{(\mathrm{L})}, \theta_j^{(\mathrm{FL})} - \theta_j^{(\mathrm{L})})}{(\sum_m w_m)^2} \tag{45}$$

$$\leq \frac{4V\sum_m w_m^2}{(\sum_m w_m)^2} + \sum_{i,j:\,i\neq j}\frac{w_iw_j\mathrm{Cov}(\sum_{k:\,k\neq i} w_k(\theta_k^{(\mathrm{L})} - \theta_i^{(\mathrm{L})}), \sum_{k:\,k\neq j} w_k(\theta_k^{(\mathrm{L})} - \theta_j^{(\mathrm{L})}))}{(\sum_m w_m)^2(\sigma_i^{-2} + \sum_{k:\,k\neq i} w_k)(\sigma_j^{-2} + \sum_{k:\,k\neq j} w_k)} \tag{46}$$

$$= \frac{4V\sum_m w_m^2}{(\sum_m w_m)^2} + \sum_{i,j:\,i\neq j} w_iw_j\frac{\sum_{k:\,k\neq i,j} w_k^2\mathbb{V}(\theta_k^{(\mathrm{L})}) - \left(w_j\mathbb{V}(\theta_j^{(\mathrm{L})})\sum_{k:\,k\neq j} w_k + w_i\mathbb{V}(\theta_i^{(\mathrm{L})})\sum_{k:\,k\neq i} w_k\right)}{(\sum_m w_m)^2(\sigma_i^{-2} + \sum_{k:\,k\neq i} w_k)(\sigma_j^{-2} + \sum_{k:\,k\neq j} w_k)} \tag{47}$$

$$\leq \frac{4V\sum_m w_m^2}{(\sum_m w_m)^2} + \sum_{i,j,k:\,i\neq j,k\neq i,k\neq j}\frac{w_iw_jw_k^2V}{(\sum_m w_m)^2(\sigma_i^{-2} + \sum_{k:\,k\neq i} w_k)(\sigma_j^{-2} + \sum_{k:\,k\neq j} w_k)} \tag{48}$$

$$\leq \frac{4V}{M}\frac{w_{\max}^2}{w_{\min}^2} + \frac{MVw_{\max}^4}{w_{\min}^2(\min_{m\in[M]}\sigma_m^{-2} + (M-1)w_{\min})^2} \tag{49}$$

$$\leq \frac{4V}{M}\frac{w_{\max}^2}{w_{\min}^2} + \frac{MV}{(M-1)^2}\frac{w_{\max}^4}{w_{\min}^4}. \tag{50}$$

Therefore, from (24), we have $\mathbb{V}(D) = O(M^{-1})$ as $M \to \infty$. By the Markov inequality, we further have

$$|D| = O_p(M^{-1/2}). \tag{51}$$

Consequently, taking equations (25), (39), and (51) into (41), we obtain

$$\max_{m \in [M]} |\zeta_m^t| = c_3 \cdot q^t + O_p(M^{-1/2}) + O(M^{-1}) = C \cdot q^t + O_p(M^{-1/2}) \text{ as } M \to \infty, \tag{52}$$

where C is the constant upper bound of $\max_{m \in [M]} |\zeta_m^0|$ (by the assumption). This proves the convergence of each client's personalized model.

For the server model denoted by θ^t, since

$$\theta^t = \frac{\sum_k w_k \theta_k^t}{\sum_k w_k} = \frac{\sum_k w_k (\theta_k^t - \theta_k^{(\mathrm{FL})})}{\sum_k w_k} + \frac{\sum_k w_k (\theta_k^{(\mathrm{FL})} - \theta_k^{(\mathrm{L})})}{\sum_k w_k} + \frac{\sum_k w_k \theta_k^{(\mathrm{L})}}{\sum_k w_k} \tag{53}$$

$$= \frac{\sum_k w_k \zeta_k}{\sum_k w_k} + D + \theta^{(\mathrm{G})}, \tag{54}$$

we have $|\theta^t - \theta^{(\mathrm{G})}| \leq C \cdot q^t + O_p(M^{-1/2})$. This concludes the proof.