Efficient Personalized Federated Learning via Sparse Model-Adaptation

Daoyuan Chen [1] **Liuyi Yao** [1] **Dawei Gao** [1] **Bolin Ding** [1] **Yaliang Li** [1]

Abstract

Federated Learning (FL) aims to train machine learning models for multiple clients without sharing their own private data. Due to the heterogeneity of clients' local data distribution, recent studies explore the personalized FL that learns and deploys distinct local models with the help of auxiliary global models. However, the clients can be heterogeneous in terms of not only local data distribution, but also their computation and communication resources. The capacity and efficiency of personalized models are restricted by the lowest-resource clients, leading to sub-optimal performance and limited practicality of personalized FL. To overcome these challenges, we propose a novel approach named pFedGate for efficient personalized FL by adaptively and efficiently learning sparse local models. With a lightweight trainable gating layer, pFedGate enables clients to reach their full potential in model capacity by generating different sparse models accounting for both the heterogeneous data distributions and resource constraints. Meanwhile, the computation and communication efficiency are both improved thanks to the adaptability between the model sparsity and clients' resources. Further, we theoretically show that the proposed pFedGate has superior complexity with guaranteed convergence and generalization error. Extensive experiments show that pFedGate achieves superior global accuracy, individual accuracy and efficiency simultaneously over state-of-the-art methods. We also demonstrate that pFedGate performs better than competitors in the novel clients participation and partial clients participation scenarios, and can learn meaningful sparse local models adapted to different data distributions.

[1]Alibaba Group. Correspondence to: Yaliang Li <yaliang.li@alibaba-inc.com>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

1. Introduction

Federated Learning (FL) gains increasing popularity in machine learning scenarios where the data are distributed in different places and can not be transmitted due to privacy concerns (Muhammad et al., 2020; Meng et al., 2021; Hong et al., 2021; Wang et al., 2022; Xie et al., 2023). Typical FL trains a unique global model from multiple data owners (clients) by transmitting and aggregating intermediate information with the help of a centralized server (McMahan et al., 2017; Kairouz et al., 2021). Although using a shared global model for all clients shows promising average performance, the inherent statistical heterogeneity among clients challenges the existence and convergence of the global model (Sattler et al., 2020; Li et al., 2020). Recently, there are emerging efforts that introduce personalization into FL by learning and deploying distinct local models (Yang et al., 2019; Karimireddy et al., 2020; Tan et al., 2021). The distinct models are designed particularly to fit the heterogeneous local data distribution via techniques taking care of relationships between the global model and personalized local models, such as multi-task learning (Collins et al., 2021), meta-learning (Dinh et al., 2020a), model mixture (Li et al., 2021c), knowledge distillation (Zhu et al., 2021) and clustering (Ghosh et al., 2020).

However, the heterogeneity among clients exists not only in local data distribution, but also in their computation and communication resources (Chai et al., 2019; 2020; Chen et al., 2023). The lowest-resource clients restrict the capacity and efficiency of the personalized models due to the following reasons: (1) The adopted model architecture of all clients is usually assumed to be the same for aggregation compatibility and (2) The communication bandwidth and participation frequency of clients usually determine how much can they contribute to the model training of other clients and how fast can they agree to meet a converged "central point" w.r.t their local models. This resource heterogeneity is under-explored in most existing personalized FL (PFL) works, and instead, they gain accuracy improvement with a large amount of additional computation or communication costs. Without special design taking the efficiency and resource heterogeneity into account, we can only gain sub-optimal performance and limited practicality of PFL.

To overcome these challenges, in this paper, we propose

a novel method named pFedGate for efficient PFL, which learns to generate personalized sparse models based on adaptive gated weights and different clients' resource constraints. Specifically, we introduce a lightweight trainable gating layer for each client, which predicts sparse, continues and block-wise gated weights and transforms the global model shared across all clients into a personalized sparse one. The gated weights prediction is conditioned on specific samples for better estimation of the heterogeneous data distributions. Thanks to the adaptability between the model sparsity and clients' resources, the personalized models and FL training process gain better computation and communication efficiency. As a result, the model-adaption via sparse gated weights delivers the double benefit of personalization and efficiency: (1) The sparse model-adaption enables each client to reach its full potential in model capacity with no need for compatibility across other low-resource clients, and to deal with a small and focused hypothesis space that is restricted by the personalized sparsity and the local data distribution. (2) Different resource restrictions can be easily imposed on the predicted weights as we consider the block-wise masking under a flexible combinatorial optimization setting. We further provide space-time complexity analysis to show pFedGate' superiority over state-of-the-art (SOTA) methods, and provide theoretical guarantees for pFedGate in terms of its generalization and convergence.

We evaluate the proposed pFedGate on four FL benchmarks compared to several SOTA methods. We show that pFedGate achieves superior global accuracy, individual accuracy and efficiency simultaneously (up to 4.53% average accuracy improvement with 12x smaller sparsity than the compared strongest PFL method). We demonstrate the effectiveness and robustness of pFedGate in the partial clients participation and novel clients participation scenarios. We find that pFedGate can learn meaningful sparse local models adapted to different data distributions, and conduct extensive experiments to study the effect of sparsity and verify the necessity and effectiveness of pFedGate' components.

Our main contributions can be summarized as follows:

- We exploit the potential of co-design of model compression and personalization in FL, and propose a novel efficient PFL approach that learns to generate sparse local models with a fine-grind batch-level adaptation.
- We provide a new formulation for the efficient PFL considering the clients' heterogeneity in both local data distribution and hardware resources, and provide theoretical results about the generalization, convergence and complexity of the proposed method.
- We achieve SOTA results on several FL benchmarks and illustrate the feasibility of gaining better accuracy with improved efficiency for PFL simultaneously. Our codes are at https://github.com/alibaba/FederatedScope/tree/master/benchmark/pFL-Bench.

2. Related Works

Personalized FL. Personalized FL draws increasing attention as it is a natural way to improve FL performance for heterogeneous clients. Many efforts have been devoted via multi-task learning (Smith et al., 2017; Corinzia & Buhmann, 2019; Huang et al., 2021; Marfoq et al., 2021), model mixture (Zhang et al., 2020; Li et al., 2021c), clustering (Briggs et al., 2020; Sattler et al., 2020; Chai et al., 2020), knowledge distillation (Lin et al., 2020; Zhu et al., 2021; Ozkara et al., 2021), meta-learning (Khodak et al., 2019; Jiang et al., 2019; Khodak et al., 2019; Singhal et al., 2021), and transfer learning (Yang et al., 2020; He et al., 2020; Zhang et al., 2021a). Although effective in accuracy improvements, most works pay the cost of additional computation or communication compared to non-personalized methods. For example, Sattler et al. (2020) consider group-wise client relationships, requiring client-wise distance calculation that is computationally intensive in cross-device scenarios. Fallah et al. (2020) leverage model agnostic meta-learning to enable fast local personalized training that requires computationally expensive second-order gradients. Zhang et al. (2021a) learn pair-wise client relationships and need to store and compute similarity matrix with square complexity w.r.t. the number of clients. Marfoq et al. (2021) learn a mixture of multiple global models which multiplies the storing and communicating costs. Our work differs from these works by considering a practical setting that clients are heterogeneous in both the data distribution and hardware resources. Under this setting, we achieve personalization from a novel perspective, personalized sparse model-adaptation, and show the feasibility of gaining better accuracy, computation and communication efficiency at the same time.

Efficient FL. Fruitful FL literature has explored the improvement of communication efficiency such as methods based on gradient compression (Rothchild et al., 2020; Alistarh et al., 2017; Reisizadeh et al., 2020; Haddadpour et al., 2021; Zhang et al., 2021b), model ensemble (Hamer et al., 2020), model sub-parameter sharing (Liang et al., 2020), sub-model selection (Tianchun et al., 2022; Minh & Carl, 2021), and Bayesian neural network (Yurochkin et al., 2019b;a). Few works improve computation and communication efficiency with personalized local models via quantization (Ozkara et al., 2021; Li et al., 2021a; Hong et al., 2022) and model parameter decoupling (Diao et al., 2021; Collins et al., 2021; Huang et al., 2022). We mainly differ from them by adaptively generating the sparse model weights at a fine-grained batch level, which estimates the conditional distribution of heterogeneous local data well and gains high accuracy as shown in our experiments. Besides

the methodology, we present several new theoretical results. Due to the space limit, we give more detailed descriptions and comparisons with some related works in Appendix G.

3. Problem Formulation

The goal of traditional federated learning (FL) is to fit a single global model by collaboratively training models from a set $\mathcal{C}$ of clients without sharing their local data. In this paper, we focus on the personalized federated learning problem in terms of not only client-distinct models, but also client-distinct resource limitations. Specifically, consider each client $i \in \mathcal{C}$ has its own private dataset $\mathcal{S}_i$ that is drawn from a local distribution $\mathcal{D}_i$ over $\mathcal{X} \times \mathcal{Y}$. In general, the local data distributions $\{\mathcal{D}_i\}_{i\in\mathcal{C}}$ are heterogeneous and thus it is promising to learn personalized model $h_{\theta_i} \in \mathcal{H} : \mathcal{X} \mapsto \mathcal{Y}$ parameterized by θ_i for each local distribution $\mathcal{D}_i$, where $\mathcal{H}$ is the set of hypotheses with d dimensions. Besides, clients may have heterogeneous computation and communication resources especially in cross-device FL scenarios. To better account for the data heterogeneity and system heterogeneity, we aim to optimize the following objective:

$$\begin{aligned} &\min_{\{h_{\theta_i}\}_{i\in\mathcal{C}}} \sum_{i\in\mathcal{C}} p_i \cdot \mathbb{E}_{(x,y)\sim\mathcal{D}_i}[f(\theta_i; x, y)], \\ &\quad s.t. \quad size(\theta_i) \le d_i, \quad \forall i \in \mathcal{C}, \end{aligned} \tag{1}$$

where $f(\theta_i; x, y) \triangleq \ell(h_{\theta_i}(x), y)$ and $\ell : \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R}^+$ is the loss function, p_i is non-negative aggregation weight for client i and $\sum_{i\in\mathcal{C}} p_i = 1$. In typical FL setting (McMahan et al., 2017; Kairouz et al., 2021), p_i indicates the participation degree of client i and is set to be proportional to the size of local dataset $|\mathcal{S}_i|$. The $size(\theta_i)$ indicates the number of parameters of θ_i, and d_i indicates the model size limitation for client i. Without special designs to handle the system heterogeneity (here the size restriction), most PFL methods simply adopt another hypothesis space $\tilde{\mathcal{H}}$ with $\min(\{d_i\}_{i\in\mathcal{C}})$ dimensions that are constraints by the lowest-resource clients, and the problem is degraded to the typical one that minimizes the objective without constraints. We consider model size constraint for simplicity and it fairly reflects the computation and communication cost. The formulation can be extended to other configurations such as inference and communication latency in a similar manner.

4. Learning to Generate Sparse Model

In practice, one can solve the problem in Equation (1) with a two-step manner: ① find optimal local models via existing personalized federated learning methods without sparsity constraints, and then ② compress the trained models into required local model size via compression techniques such as pruning, quantization and knowledge distillation (Deng et al., 2020). However, the final local models from step ② usually gain worse performance to models found in step ① (Jiang et al., 2022). Besides, the post-compression process still requires computational and communication costs corresponding to the un-compressed models during the FL process. To alleviate the performance degradation and further improve the efficiency of FL process, we propose to jointly learn and compress the federated models. Specifically, we can directly learn sparse local models whose number of non-zero parameters satisfy the size requirements:

$$\min_{\{h_{\theta'_i}\}_{i\in\mathcal{C}}} \sum_{i\in\mathcal{C}} p_i \cdot \mathbb{E}_{(x,y)\sim\mathcal{D}_i}[\ell(h_{\theta'_i}(x), y)], \tag{2}$$

where $h_{\theta'_i} \in \mathcal{H}_{d_i}$ and $\mathcal{H}_{d_i}$ indicates the subset of $\mathcal{H}$ with hypotheses whose number of non-zero parameters is not larger than d_i. However, the sparse models $h_{\theta'_i}$ can be arbitrarily different across clients due to the various dimension requirements d_i and local data distributions $\mathcal{D}_i$. Directly optimizing $h_{\theta'_i}$ and aggregating them may lead to sub-optimal performance and make the federated learning process unconvergent. As (Marfoq et al., 2021) discussed, clients can not benefit from each other without any additional assumption to the local distributions. Here we consider the case of bounded diversity as following assumption shows:

Assumption 1. *(Bounded Diversity) Let $\mathcal{D}_g$ be a global data distribution that is a mixture of all local data distribution $\mathcal{D}_i$, $i \in \mathcal{C}$. Denote $h_{\theta^*_g}$ and $h_{\theta^*_i}$ as the optimal estimator for the conditional probability of $\mathcal{D}_g$ and $\mathcal{D}_i$ respectively. There exist a $\mathcal{D}_g$ such that the variance between local and global gradients is bounded*

$$||\nabla_\theta f(\theta^*_i) - \nabla_\theta f(\theta^*_g)||^2 \le \sigma_i^2, \quad \forall i \in \mathcal{C}, \tag{3}$$

where $f(\theta)$ is a compact notation as $f(\theta) \triangleq f(\theta; x, y)$ for all data points $(x, y) \in \mathcal{X} \times \mathcal{Y}$.

Proposition 1. *Denote $(\theta'_i)^*$ be the parameter of any sparse optimal estimator for $\mathcal{D}_i, i \in \mathcal{C}$. If f is μ-strongly convex [1], under Assumption 1, there exist continuous weights $M^*_i \in \mathbb{R}^d$ such that $(\theta'_i)^* = \theta^*_g \circ M^*_i$ where $\circ$ indicates Hadamard production over model parameters of θ, and $\{M^*_i\}_{i\in\mathcal{C}}$ can be bounded as*

$$||(M^*_i - 1)^{\circ 4}|| \le \frac{1}{16} \frac{(R^4_{\theta_i} R^4_{\theta_g} r^4_{\theta_i} r^4_{\theta_g})}{(\sigma_i^4 \mu^4 ||(\theta^*_g)^{\circ 4}||)} \quad \forall i \in \mathcal{C}, \tag{4}$$

where $\circ 4$ indicates Hadamard power of 4, r_{θ_i} and r_{θ_g} are the infimum of θ_i and θ_g respectively, R_{θ_i} and R_{θ_g} are the supremum of θ_i and θ_g respectively. [2]

Proposition 1 suggests that we can deal with Problem (2) via indirectly generating personalized sparse models $h_{\theta'_i}$

[1] The strong convexity is only used in the motivated proposition and not involved in the remaining theoretical analysis. We give more discussion on all the Assumptions in Appendix E.

[2] The proofs of all propositions and theorems of our work are in Appendix.

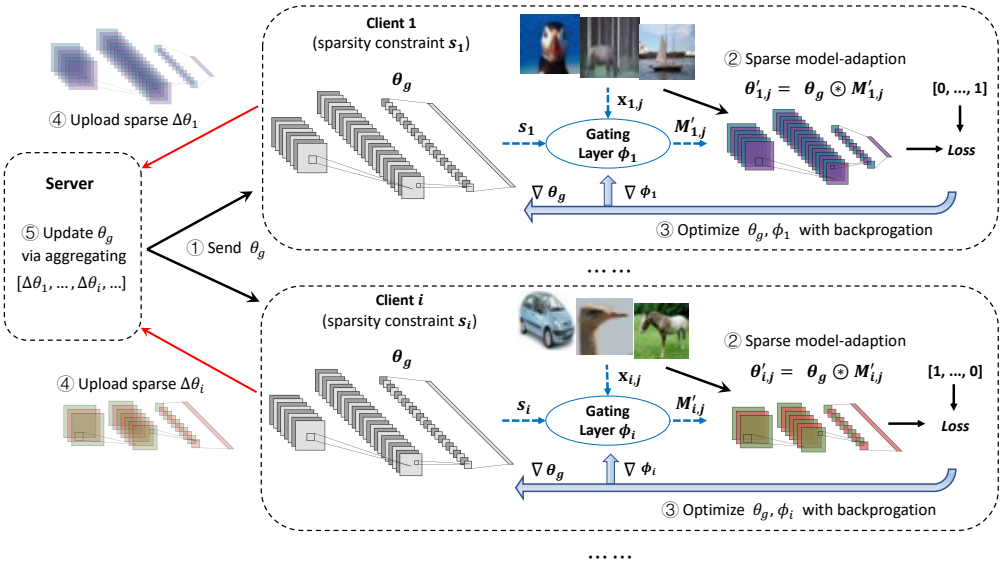

Figure 1: The framework of pFedGate that learns to generate sparse personalized models conditioned on clients' different data samples and resource limitations. The serial numbers indicate the FL processes at each communication round.

parameterized by $\theta'_i = \theta_g \circ M_i$ with personalized weights M_i and a shared global model θ_g, which helps to transfer information across clients and leads to a controllable optimization. Specifically, denote s_i be the model sparsity constraint of client i, Problem (2) can be transformed into following form:

$$\min_{\theta_g, \{M_i\}_{i\in\mathcal{C}},} \sum_{i\in\mathcal{C}} p_i \cdot \mathbb{E}_{(x,y)\sim\mathcal{D}_i}[f(\theta_g \circ M_i; x, y)], \quad s.t. \quad count(M_i \neq 0)/d \leq s_i, \quad \forall i \in \mathcal{C}. \tag{5}$$

With this re-formulation, M_i can be regarded as sparsity-enforced gated weights that absorb the diversity of different local data distribution into the global model, via *scaling* and *blocking* the information flow through the shared model θ_g.

5. PFL with Conditional Gated Weights

5.1. Overall Framework

Note that for discriminative models (*e.g.*, neural networks), the personalized models are estimators for conditional distribution $P_{\mathcal{D}_i}(y|x)$ associated with clients' local data $\mathcal{D}_i$. To achieve efficient PFL and optimize the objective described in Equation (5), instead of learning client-level element-wise weights $M_i \in \mathbb{R}^d$, we propose to learn to predict batch-level block-wise weights $M'_{i,j} = g_{\phi_i}(\mathbf{x}_{i,j})$, where ϕ_i indicates the parameters of a trainable personalized gating layer, and g_{ϕ_i} is the predictive function conditioned on specific data batch $\mathbf{x}_{i,j}$ and sparsity limitation s_i. The gating layer ϕ_i brings up benefits in both efficiency and effectiveness: (1) predicting block-wise weights $M'_{i,j} \in \mathbb{R}^L$ enables us to use a lightweight ϕ_i for $L \in \mathbb{N}^+$ sub-blocks of θ with $L \ll d$, and thus gain much smaller computational costs than the element-wise manner; (2) predicting batch-wise weights can estimate the conditional probability better than the client-level manner since each client may have complex local data distribution mixed by different classes, *e.g.*, using different sparse models to classify dog and bird can achieve better performance than using the same one. We empirically verify the effectiveness of these choices in Section 7.6.

Based on the introduced gating layer, we establish an efficient PFL framework in server-client setting as shown in Figure 1. To enable benefits from each other, all clients share the same global model θ_g that is downloaded from server. Each client $i \in \mathcal{C}$ has a private, lightweight and learnable gating layer ϕ_i, which is trained from local data in a differentiable manner, and learns to achieve fine-grained personalized model-adaptation $\theta'_{i,j} = \theta_g \circledast M'_{i,j} = \theta_g \circledast g_{\phi_i}(\mathbf{x}_{i,j})$ given different data batches, where $\circledast$ indicates block-wise production of θ. To handle diverse resource-limited scenarios, ϕ_i ensures that the personalized adapted model $\theta'_{i,j}$ has required sparsity not larger than s_i. Moreover, the sparsity speeds up the training and inference of local models, as well as the communication via uploading the sparse model updates (more discussion are in Sec.6.3 and Appendix F).

5.2. Adaptive Gating Layer

5.2.1. Layer Architecture

We adopt a simple two-path sandwich architecture for the gating layer as shown in the left part of Figure 2. One path

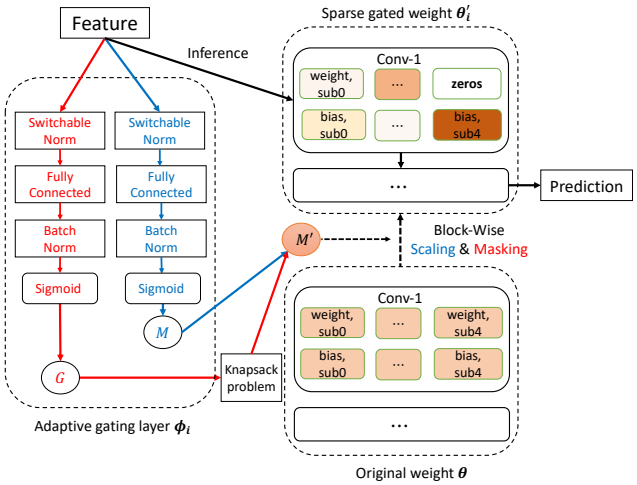

Figure 2: The sparse model-adaptation via gating layer.

predicts the block-wise gated weights M that may violate the hard sparsity constraints, and the other path predicts the block-wise importance scores that are used to adjust M into a sparse one M' such that the final sparse model satisfies the hard limitation (we introduce the adjustment later). Let the number of sub-blocks of θ_g be L and the dimension of flattened input feature be d_X, *i.e.*, $x \in \mathbb{R}^{d_X}$. The middle two fully connected layers have the same size and each of them has $(d_X \cdot L)$ parameters. Here the switchable normalization (Luo et al., 2019) is used to handle batched data samples and different input types via learning mixture of batch norm, instance norm and layer norm with negligible additional number of trainable parameters. Furthermore, the personalized normalization parameters within client-distinct gating layers enable our model-adaptation to absorb different feature distributions (similar empirical evidence is shown by Li et al. (2021d)). The final batch normalization and sigmoid activation are used to stabilize the optimization of θ_g and $\{\phi_i\}$ by bounding M' and thus the scaling degrees on θ_g's parameters (recall that the upper bound in Proposition 1 is dependent on the infimum and supremum of parameters after scaling). In total, the gating layer learns and predicts the sparse gated weights $M' = g_\phi(x) = g(\phi; \mathbf{x})$ given specific data batch $\mathbf{x}$ where $g : \mathbb{R}^{d_X} \mapsto \mathbb{R}^L$, with a lightweight parameter size $d_\phi = 2d_X L$ that is usually much smaller than the size of model θ_g. We show the effectiveness and necessity of these adopted components in Section 7.6.

5.2.2. Operator-type-free Block Splitting

There are many alternatives to split a model into certain sub-blocks, *e.g.*, a convolutional operator can be split channel-wise or filter-wise and an attention operator can be split head-wise. Our gating layer is decoupled with specific block splitting manners and thus can easily support users' various splitting preferences. Besides, we propose a general operator-type-free splitting manner by default to enhance the usability of our approach as illustrated in the right part of Figure 2. Denote $B \in \mathbb{N}^+$ as a given block split factor, and denote d_l as the dimension of a flattened vector that parameterizes the given learnable operator. We group the vector's first $\lfloor d_l \cdot s_{min} \rfloor$ elements as the first sub-block, and equally split the remaining elements into $(B-1)$ sub-blocks (the last sub-block may has fewer size than $\lfloor (d_l \cdot (1 - s_{min})/(B-1) \rfloor$). Here B provides flexible splitting granularity, $s_{min} \in (0, s_i]$ indicates a pre-defined minimal sparsity factor and the predicted gated weights for the first sub-block is enforced to be non-zero to avoid cutting information flow through this operator. The proposed splitting is simple to implement but effective for PFL as shown in our empirical evaluations.

5.2.3. Sparse Gated Weights

To ensure an equal or smaller sparsity of adapted model than the limitation s_i, we propose to leverage the predicted block importance G to adjust the predicted gated weights M into a sparse one M'. Specifically, denote $W \in \mathbb{Z}^L$ be the parameter size look-up table of L sub-blocks, we transform $M' = MI^*$ where $I^* \in \{0,1\}^L$ is binary index and a solution of the following knapsack problem that maximizes total block importance values while satisfying sparsity constraint:

$$\max_I \quad I \cdot G, \qquad s.t. \quad |I \cdot W|/d \leq s_i. \tag{6}$$

In practice, to enable gradient-based optimization of gating layer, we leverage the straight-through trick (Hubara et al., 2016) via differentiable G in backward pass. In Appendix A, we summarize the overall algorithm and present more details about the gradient flows through the gating layer.

6. Algorithm Analysis

6.1. Generalization

Following the parameter-sharing analysis from Baxter (2000) and the generalization analysis from Shamsian et al. (2021), here we give the generalization guarantee of our sparse model. Based on the notations used in previous sections, let $\hat{\mathcal{L}}(\theta_g, \phi_i)$ denote the empirical loss of the sparse model in client i, and $\hat{\mathcal{L}}(\theta_g, \phi_i) \triangleq \frac{1}{|\mathcal{S}_i|} \sum_{(x,y)\in\mathcal{S}_i} f\Big(\big(\theta_g \circledast g_{\phi_i}(x)\big); x, y\Big)$. The expected loss is denoted as $\mathcal{L}(\theta_g, \phi_i) \triangleq \mathbb{E}_{(x,y)\sim\mathcal{D}_i} f\Big(\big(\theta_g \circledast g_{\phi_i}(x)\big); x, y\Big)$.

Assumption 2. *The parameters of global model and the gating layer can be bounded in a ball with a radius of R. The following Lipschitz conditions hold:* $|f(\theta; x, y) - f(\theta'; x, y)| \leq L_f ||\theta_g - \theta'||$, $||h(\theta_i; x) - h(\theta_i'; x)|| \leq L_h ||\theta_i - \theta_i'||$ *and* $||g(\phi_i; x) - g(\phi_i'; x)|| \leq L_g ||\phi_i - \phi_i'||$, *where* $h(\theta_i; \cdot)$ *is the sparse model parameterized by* θ_i *and* $g(\phi_i; \cdot)$ *indicates the gating layer parameterized by* ϕ_i.

Let the parameter spaces of the sparse model and gating layer be of size d and d_ϕ, separately. With the above assumption, we have the following theorem regarding the generalization error bound.

Theorem 1. *Under Assumption 2, with probability at least $1-\delta$, there exist $\tilde{N} = \mathcal{O}(\frac{d}{|\mathcal{C}|\epsilon^2}\log\frac{RL_fL_h(RL_g+s_i)}{\epsilon} + \frac{d_\phi}{\epsilon^2}\log\frac{RL_fL_h(RL_g+s_i)}{\epsilon} - \frac{\log\delta}{|\mathcal{C}|\epsilon^2})$ such that for all θ_g, ϕ_i, $|\mathcal{L}(\theta_g,\phi_i) - \hat{\mathcal{L}}(\theta_g,\phi_i)| \leq \epsilon$ when the number of client i's local data samples is greater than $\tilde{N}$.*

Theorem 1 indicates the generalization depends on the size of the global model (*i.e.*, d) reduced by the number of clients $|\mathcal{C}|$ (the first term), as the global model is shared by all clients. It also depends on the size of the gating layers (the second term), and it is not reduced by $|\mathcal{C}|$ since each client has its own personalized parameters. Besides, the generalization is also affected by the Lipschitz constants of the global model, sparse model, and the gating layer, as they together constrain the parameter space that our method can search. Specifically, as the sparsity s_i decreases, the smaller the $\tilde{N}$, and the better the generalization. This is also to some extent a reflection of Ockham's Razor in our approach. There is similar corroboration in other scenarios, such as the positive effect of the sparsity regularity on generalization ability (Maurer & Pontil, 2012).

6.2. Convergence

Under the following assumptions that are commonly used in convergence analysis of many FL works (Kairouz et al., 2021) (more discussion about the Assumptions are in Appendix E), we can see that the global model updates $\{\Delta\theta^t_{g,i}\}_{i\in\mathcal{C}_s}$ become arbitrarily small, meanwhile the learnable parameters of pFedGate (both θ_g and gating layers $\{\phi_i\}_{i\in\mathcal{C}}$) converge to stationary points with comparable convergence rate to many existing works.

Assumption 3. *(Smoothness) For all clients $i \in \mathcal{C}$, $(x,y) \in \mathcal{D}_i$ and all possible θ_g, the function $f(\theta_g;x,y) \triangleq \ell(h_{\theta_g}(x),y)$ is twice continuously differentiable and L-smooth: $||\nabla f(\theta_g^{'};x,y) - \nabla f(\theta_g;x,y)||^2 \leq L||\theta_g^{'} - \theta_g||$.*

Assumption 4. *(Bounded Output) The function f is bounded below by $f^* \in \mathbb{R}$.*

Assumption 5. *(Bounded variance) For client $i \in \mathcal{C}$ and all possible θ_g, given data sample (x,y) drawn from local data $\mathcal{S}_i$, the local gradient estimator $\nabla_\theta \ell(h_{\theta_g}(x),y)$ is unbiased and has bounded variance σ^2.*

Theorem 2. *Under Assumptions 1–5, if clients use SGD as local optimizer with learning rate η, there exist a large enough number of communication rounds $T^{'}$, such that pFedGate converges with $\eta = \frac{\eta_0}{\sqrt{T^{'}}}$:*

$$\frac{1}{T^{'}}\sum_{t=1}^{T^{'}}\mathbb{E}\left\|\nabla_\theta f\Big(\big(\theta_g^t \circledast g_{\phi_i^t}(x)\big);x,y\Big)\right\|^2 \leq \mathcal{O}\left(\frac{1}{\sqrt{T^{'}}}\right),\ \forall i \in \mathcal{C}, \tag{7}$$

where the expectation is over the random samples and $\mathcal{O}(\cdot)$ hides polylogarithmic factors.

6.3. Space-Time Complexity

We briefly summarize the efficiency of pFedGate in terms of space-time costs. With the structured block sparsity (Sec.5.2.2), the training and inference cost of pFedGate is $\mathcal{O}\big(d(s_i{+}s_{\phi_i})\big)$ for client i at each round, where the model sparsity s_i and relative sparsity of gating layer $s_{\phi_i}{=}count(\phi_i{\neq}0)/d$ are usually small and lead to $(s_i{+}s_{\phi_i}) < 1$. By contrast, the computation cost of FedAvg and several SOTA PFL methods is at least $\mathcal{O}(d)$. The introduced sparsity provides great potential for boosting computation efficiency. For the communication, the upload parameter number of pFedGate is $\mathcal{O}(q_i d)$ that can be smaller than the one of baselines, $\mathcal{O}(d)$. The $q_i{=}count(\Delta\theta_i \neq 0)/d$ indicates the ratio of non-zero model updates, which depends on s_i and local samples trained in the round.

In Appendix F, we provide detailed discussions for these terms and show superiority of pFedGate over several SOTA competitors.

7. Experiments

7.1. Experimental Settings

We adopt four widely used FL datasets in our experiments: EMNIST (Cohen et al., 2017), FEMNIST (Caldas et al., 2018), CIFAR10 and CIFAR100 (Krizhevsky, 2009). They are partitioned into several sub-datasets to simulate the local dataset for a client, and each of them is randomly split into train/val/test datasets with ratio 6:2:2. In the experiments, the data partition follows the heterogeneous settings adopted by Marfoq et al. (2021); Dinh et al. (2020b). For baselines, we choose the classic FedAvg (McMahan et al., 2017) and FedAvg with fine-tuning (*FedAvg-FT*), and as well as several SOTA PFL methods, including *pFedMe* (Dinh et al., 2020b), *LG-FedAvg* (Liang et al., 2020), *Ditto* (Li et al., 2021c) and *FedEM* (Marfoq et al., 2021). In addition, we consider several SOTA PFL methods that also improve efficiency via shared model representations (*FedRep* (Collins et al., 2021)), binary quantization (*FedMask* (Li et al., 2021a)), and sparse sub-models (*HeteroFL* (Diao et al., 2021)).

Due to limited space, please refer to Appendix H for more details about the datasets, models and baselines.

Table 1: Accuracy comparison on widely-adopted FL datasets. $\overline{Acc}$ (%) and $\breve{Acc}$ (%) indicate the average accuracy of all clients and accuracy of the bottom decile clients respectively. $\bar{s}'$ indicates the average of finally achieved sparsity across all clients. Bold and underlined numbers indicate the best and second-best results respectively.

	EMNIST		FEMNIST		CIFAR10		CIFAR100		Average	
	$\overline{Acc}$	$\breve{Acc}$	$\overline{Acc}$	$\breve{Acc}$	$\overline{Acc}$	$\breve{Acc}$	$\overline{Acc}$	$\breve{Acc}$	$\overline{Acc}$	$\breve{Acc}$
Local	71.91	64.28	71.12	57.93	61.91	53.71	26.95	21.13	57.97	49.26
FedAvg	82.54	75.07	76.51	60.82	68.33	60.22	35.21	29.82	65.65	56.48
FedAvg-FT	83.19	76.83	78.43	64.22	69.91	61.55	37.15	30.91	67.17	58.38
pFedMe	83.29	76.48	75.29	57.63	70.29	62.13	36.14	30.13	66.25	56.59
LG-FedAvg	83.25	76.14	75.13	57.62	69.54	61.86	36.72	30.51	66.16	56.53
Ditto	82.75	78.12	79.29	63.24	73.14	62.59	37.32	30.11	68.13	58.52
FedEM	83.41	76.59	80.12	64.81	72.43	62.88	38.28	31.04	68.56	58.83
FedRep	82.42	77.71	79.17	62.89	72.77	63.44	36.68	30.51	67.76	58.64
FedMask, $\bar{s}'$=0.5	81.95	77.26	78.69	62.18	72.43	62.88	36.21	30.13	67.32	58.11
HeteroFL, $\bar{s}'$=0.55	85.64	77.76	77.22	59.13	70.97	63.64	37.01	31.27	67.71	57.95
pFedGate, s=1	<u>87.11</u>	<u>81.43</u>	**87.32**	**77.14**	**75.18**	**66.67**	**42.01**	**35.03**	**72.91**	**65.07**
s=0.5, $\bar{s}'$=0.43	**87.28**	81.15	86.31	75.68	<u>74.07</u>	64.21	<u>40.07</u>	<u>32.38</u>	<u>71.93</u>	63.36
s=0.3, $\bar{s}'$=0.25	87.09	**82.52**	<u>86.75</u>	<u>76.47</u>	73.65	<u>64.39</u>	39.53	31.63	71.76	<u>63.75</u>

7.2. Overall Performance

Clients' Average Performance. We first examine the overall performance by evaluating the accuracy on local test set of each client, and averaging the results of all clients with weights proportional to their local dataset sizes. The detailed overall performance is shown in Table 1, where $\bar{s}'$ indicates the average of finally achieved sparsity across all clients (our block-wise operation may lead to $\bar{s}' \leq s$). We can see that pFedGate achieves better accuracy (averaged 3.1% to 4.53% accuracy improvements) and smaller sparsity even with s=0.3 at the same time. Compared with FedEM, the baseline with best accuracy, pFedGate doesn't have the heavy storage burden, since FedEM needs to store multiple (3) global models in the client. These observations demonstrate the effectiveness and efficiency of the proposed method. Note that compared to the SOTA efficient method HeteroFL, our method achieves better accuracy while with smaller sparsity (0.25 v.s. 0.55). When s=1, we achieve the best performance without sparsity constraints, verifying that the batch-level block-wise adaptation has a strong ability to achieve personalization. When s <1, we still gain comparable performance with s=1, providing evidence that there exist well-performed sparse models and they can be effectively learned by our approach.

Individual Client Performance. We then examine whether pFedGate improved the average performance by sacrificing the performance of some clients. In Table 1, we mark the accuracy of bottom decile local models $\breve{Acc}$ as the $\lfloor |\mathcal{C}|/10 \rfloor$)-th worst accuracy (following Marfoq et al. (2021), we neglect the particularly noisy results from clients with worse accuracy due to their very small local data sizes). We find that pFedGate also gains significant $\breve{Acc}$ improvement (averaged 4.92% to 6.24% than FedEM). We plot the client-wise accuracy difference between our method with $s = 0.5$ and the strongest baseline, FedEM in Figure 3, in which pFedGate significantly improves most clients compared to the strongest baseline. The results in Table 1 and Figure 3 show that pFedGate not only improves the average accuracy and efficiency, but also fairly improves the individual client performance by learning to generate personalized models with adaptive gating layers.

7.3. Personalization Study

We propose to learn personalized gating layers $[\phi_i]$ that estimate the heterogeneous data distribution with sparse model-adaption. To investigate the level of personalization achieved by pFedGate, we conduct experiments on another CIFAR10 non-i.i.d. partition with 10 clients and pathological splitting (McMahan et al., 2017) that sorts the data by labels and assigns to clients with equal-sized shards.

We do hierarchical clustering for the parameters of learned gating layers $[\phi_i]$ with Ward's method (Ward Jr, 1963) and euclidean distance. We illustrate the results in Figure 4 when the models of clients converge with s=0.3 and achieve good performance ($\overline{Acc}$=89.21%), where we mark the local label categories of clients and common label categories after merging in red numbers. Interestingly, we can see that clients learn gating layers with smaller parameter distances along with they share more similarities in label categories. This indicates that pFedGate not only can achieve

Figure 3: The client-wise accuracy difference between pFedGate and FedEM on CIFAR10 dataset.

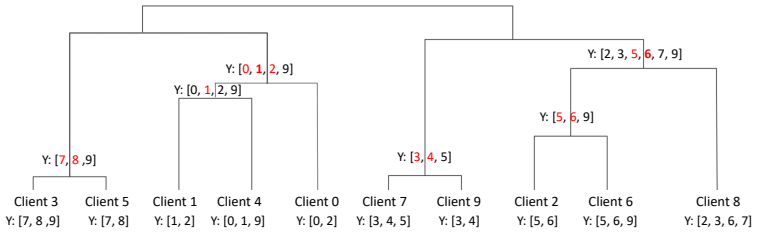

Figure 4: The hierarchical clustering result for parameters of clients' gating layers on CIFAR-10 with 10-clients partition.

Table 2: Training time t (ms) and peak memory Mem (Mb) for one FL round. bz indicates the batch size.

	bz = 32		bz = 128		bz = 512	
	Mem	t	Mem	t	Mem	t
FedAvg	26.0	3.07	27.1	3.89	31.8	4.51
FedEM	29.4	9.02	31.2	11.38	36.3	13.11
ours, s=1	26.8	3.65	27.9	4.09	32.4	4.66
s=0.5	21.0	2.44	22.2	3.21	26.7	3.58
s=0.3	16.9	2.32	18.0	2.81	22.9	3.33

good client-distinct personalization, but also implicitly learn group-wise personalization with a strong sparsity constraint, which could be beneficial to applications where there are inherent partitions among clients.

7.4. Efficiency Comparison

We have seen that pFedGate can achieve good accuracy with sparse personalization. To examine its efficiency as we analyzed in Appendix F, we track training time in ms and peak memory in Mb for one client during one FL round. The results on the CIFAR-10 dataset with different batch sizes are listed in Table 2. We can see that pFedGate can effectively improve the training efficiency and memory cost, where the results are averaged by the number of trained data batches. For example, when s=0.3 and bz=128, pFedGate uses 66% peak memory and 72% training time compared with FedAvg, meanwhile having 5.32% $\overline{Acc}$ and 4.17% $\overset{\smile}{Acc}$ improvements (Table 1). When compared to FedEM that leverages 3 global models, the efficiency is more significantly improved (66% Mem and 28% t) with superior accuracy (1.22% $\overline{Acc}$ and 1.51% $\overset{\smile}{Acc}$ improvements).

It is worth noting that in addition to the model parameters, the training process (including the backward propagation phase) also requires storing the intermediate tensors, whose size depends on the model architecture, the shape of the inputs, and the specific implementation. Thus once the model-adaption is completed, the intermediate tensor involved in the forward and backward processes through the sparse model is correspondingly smaller based on block sparsity, allowing us to get a lower training memory peak and less training time than FedAvg and FedEM. In addition, we empirically see that pFedGate has a comparable convergence speed to the baselines (Appx I.2), with the results of the single-FL-round and single-client efficiency given above, we can see that pFedGate is able to improve accuracy while at the same time improving system efficiency.

7.5. Partial Clients Participation

Table 3: Averaged accuracy when randomly sampling 20% clients at each round. The last two columns indicate the average drop compared to non-sampled case.

	FEMNIST (↑)		CIFAR10 (↑)		Drop (↓)	
	$\overline{Acc}$	$\overset{\smile}{Acc}$	$\overline{Acc}$	$\overset{\smile}{Acc}$	$\overline{Acc}$	$\overset{\smile}{Acc}$
FedAvg	77.45	57.98	66.31	57.96	3.56	2.55
FedEM	78.54	63.37	71.12	60.76	1.45	1.78
ours, s=1	87.86	77.43	74.29	64.11	0.82	1.13
s=0.5	86.72	75.12	74.25	64.36	0.79	0.21
s=0.3	85.78	74.89	73.91	64.38	0.36	0.80

We have seen our method can consistently and simultaneously improve the efficiency and model accuracy as Table 1 shows. One can arise such a natural question: when combined with other efficiency improvement techniques such as client sampling, is pFedGate still effective? Here we run experiments by uniformly sampling 20% clients without replacement at each round, and summarize the results in Table 3. It is observed that compared with baselines having about 1.45% $\sim$ 3.56% performance drop, our method still achieves strong and comparable performance to the non-sampled case, verifying the effectiveness and robustness of our proposed method again.

7.6. Ablation Study

Choices of Gating Layer. To gain further insight into our method, we ablate the gating layer w.r.t. its switchable norm layer (*pre-norm*) and batch norm layer (*post-norm*) within the sandwich structure and the *sigmoid* activation

Table 4: Average accuracy and round to achieve $0.8 \cdot \overline{Acc}$ accuracy ($T_{0.8}$) normalized by standard pFedGate.

	EMNIST		CIFAR100	
	$\overline{Acc}$	$T_{0.8}$	$\overline{Acc}$	$T_{0.8}$
pFedGate, s=0.5	87.28	1	39.72	1
w/o pre-norm	86.25	1.13	38.32	1.24
w/o post-norm	87.14	1.10	39.67	1.43
w/o sigmoid	83.55	1.85	31.29	1.93
client-wise adaptation	84.83	0.74	33.52	0.86

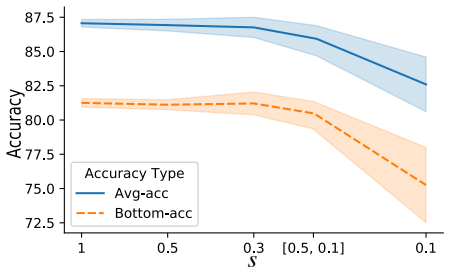

Figure 5: The accuracy on EMNIST when varying s.

that bounds the gated weights. We also change the batch-wise adaption into *client-wise* manner by feeding the input feature mean-pooled over all local training data samples of each client. We show the results in Table 4 with s=0.5.

We find that the two normalization layers improve the convergence speed. Besides, *pre-norm* has a larger impact on performance than *post-norm* since it helps to handle the heterogeneous feature distribution. The *sigmoid* significantly impacts both the convergence and accuracy, showing the effectiveness of gated weights bounding operation, which restricts the dissimilarity level between the personalized model and global model, and in turn helps to learn a global model suitable for different clients. Compared to batch-level manner, although *client-wise adaption* achieves a faster training, it pays cost of significant accuracy drop due to the coarser-grained modeling of conditional distribution of local data and smaller parameter space could be explored under sparsity constraints. In a nutshell, the ablation results verify effectiveness and necessity of our choices in gating layer.

Effect of s. We further study the effect of the sparsity factor s by varying s from 1 to 0.1 on EMNIST dataset and illustrate the results in Figure 5, in which "$[0.5, 0.1]$" indicates that we randomly divide all clients into two equal-sized parts and set s to be 0.5 and 0.1 for these two parts respectively. We can see that our method is robust to keep a good performance when adopting not very small sparsity degrees. Another observation is that the performance variance increases as s decreases. Besides, our method still performs well in a sparsity-mixed setting, verifying the robustness of our method again.

7.7. Overview of More Experiments in Appendix

Due to the space limitation, we provide further experiments and analysis in Appendix in terms of

- Generalization: In Appendix I.1, we present additional results of pFedGate for the un-seen clients that haven't participated in the FL training stage (Marfoq et al., 2021; Yuan et al., 2022; Chen et al., 2022), in which case pFedGate achieves better performance than baselines since it only needs to train the personalized lightweight gating layers with a small number of parameters.
- Convergence: In Appendix I.2, we show that pFedGate can achieve effective optimization, which supports the Theorem 2 and further verifies the efficiency of pFedGate.
- Sparse factor study: In Appendix I.3, we further study the effect of s w.r.t. global-local model gap, FL communication frequency and data heterogeneity degree.
- Sparse model study: In Appendix I.4, we vary the block splitting factor B of pFedGate and make more detailed comparison to FedMask.

8. Conclusion and Future Work

Existing personalized Federated Learning (PFL) methods usually focus on the heterogeneity of local data distribution only and pay additional computation and communication costs. In this paper, we explore the organic combination of model compression and PFL by adaptively generating different sparse local models at a fine-grained batch level, which gains double benefit of personalization performance and efficiency. Theoretical analyses are provided w.r.t. generalization, convergence and space-time complexity. Extensive experiments are conducted to verify the effectiveness, efficiency and robustness of the proposed method in various benchmarks and scenarios. We demonstrate the feasibility of obtaining superior accuracy and efficiency simultaneously for PFL, and hope this work can facilitate more studies on efficient and practical PFL.

Our work also raises some interesting questions such as do the sparse models help the protection against privacy attacks in FL (Zhu & Han, 2019; Qin et al., 2023), since attackers may face more challenging parameter space coupled with unknown personalized gating layers. Besides, the local model update of each client can be dense especially when the number of local training batches is large, as the final mask is the union over triggered sparse masks from different batches. Restricting the triggered masks via intersection operation without compromising the convergence is also worth exploring, which can further provide a worst-case guarantee with user-preferred sparsity values.

References

Ainsworth, S. K., Hayase, J., and Srinivasa, S. Git re-basin: Merging models modulo permutation symmetries. *arXiv preprint arXiv:2209.04836*, 2022.

Alistarh, D., Grubic, D., Li, J., Tomioka, R., and Vojnovic, M. Qsgd: Communication-efficient sgd via gradient quantization and encoding. *NeurIPS*, 30:1709–1720, 2017.

Baxter, J. A model of inductive bias learning. *JAIR*, 12: 149–198, 2000.

Bibikar, S., Vikalo, H., Wang, Z., and Chen, X. Federated dynamic sparse training: Computing less, communicating less, yet learning better. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 6080–6088, 2022.

Briggs, C., Fan, Z., and Andras, P. Federated learning with hierarchical clustering of local updates to improve training on non-iid data. In *IJCNN*, pp. 1–9. IEEE, 2020.

Caldas, S., Duddu, S. M. K., Wu, P., Li, T., Konečný, J., McMahan, H. B., Smith, V., and Talwalkar, A. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

Chai, Z., Fayyaz, H., Fayyaz, Z., Anwar, A., Zhou, Y., Baracaldo, N., Ludwig, H., and Cheng, Y. Towards taming the resource and data heterogeneity in federated learning. In *2019 USENIX Conference on Operational Machine Learning (OpML 19)*, pp. 19–21, 2019.

Chai, Z., Ali, A., Zawad, S., Truex, S., Anwar, A., Baracaldo, N., Zhou, Y., Ludwig, H., Yan, F., and Cheng, Y. Tifl: A tier-based federated learning system. In *Proceedings of the 29th International Symposium on High-Performance Parallel and Distributed Computing*, pp. 125–136, 2020.

Chen, D., Gao, D., Kuang, W., Li, Y., and Ding, B. pFL-bench: A comprehensive benchmark for personalized federated learning. In *NeurIPS*, 2022.

Chen, D., Gao, D., Xie, Y., Pan, X., Li, Z., Li, Y., Ding, B., and Zhou, J. FS-REAL: Towards real-world cross-device federated learning. In *KDD*, 2023.

Cohen, G., Afshar, S., Tapson, J., and Van Schaik, A. Emnist: Extending mnist to handwritten letters. In *IJCNN*, pp. 2921–2926. IEEE, 2017.

Collins, L., Hassani, H., Mokhtari, A., and Shakkottai, S. Exploiting shared representations for personalized federated learning. In *ICML*, volume 139, pp. 2089–2099, 18–24 Jul 2021.

Corinzia, L. and Buhmann, J. M. Variational federated multi-task learning, 2019.

Deng, L., Li, G., Han, S., Shi, L., and Xie, Y. Model compression and hardware acceleration for neural networks: A comprehensive survey. *Proceedings of the IEEE*, 108 (4):485–532, 2020.

Diao, E., Ding, J., and Tarokh, V. Heterofl: Computation and communication efficient federated learning for heterogeneous clients. In *ICLR*, 2021.

Dinh, C. T., Tran, N. H., and Nguyen, T. D. Personalized federated learning with moreau envelopes. *ArXiv*, abs/2006.08848, 2020a.

Dinh, C. T., Tran, N. H., and Nguyen, T. D. Personalized federated learning with moreau envelopes. In *NeurIPS*, 2020b.

Fallah, A., Mokhtari, A., and Ozdaglar, A. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. In *NeurIPS*, volume 33, pp. 3557–3568, 2020.

Ghosh, A., Chung, J., Yin, D., and Ramchandran, K. An efficient framework for clustered federated learning. In *NeurIPS*, volume 33, 2020.

Haddadpour, F., Kamani, M. M., Mokhtari, A., and Mahdavi, M. Federated learning with compression: Unified analysis and sharp guarantees. In *AISTATS*, pp. 2350–2358. PMLR, 2021.

Hamer, J., Mohri, M., and Suresh, A. T. Fedboost: A communication-efficient algorithm for federated learning. In *ICML*, pp. 3973–3983. PMLR, 2020.

Hardy, G. H., Littlewood, J. E., Pólya, G., Pólya, G., et al. *Inequalities*. Cambridge university press, 1952.

He, C., Annavaram, M., and Avestimehr, S. Group knowledge transfer: Federated learning of large cnns at the edge. *NeurIPS*, 33, 2020.

Hong, J., Zhu, Z., Yu, S., Wang, Z., Dodge, H., and Zhou, J. Federated adversarial debiasing for fair and transferable representations. In *KDD*, 2021.

Hong, J., Wang, H., Wang, Z., and Zhou, J. Efficient split-mix federated learning for on-demand and in-situ customization. In *ICLR*, 2022.

Huang, T., Liu, S., Li, S., He, F., Lin, W., and Tao, D. Achieving personalized federated learning with sparse local models. *ArXiv*, abs/2201.11380, 2022.

Huang, Y., Chu, L., Zhou, Z., Wang, L., Liu, J., Pei, J., and Zhang, Y. Personalized cross-silo federated learning on non-iid data. In *AAAI*, volume 35, pp. 7865–7873, 2021.

Hubara, I., Courbariaux, M., Soudry, D., El-Yaniv, R., and Bengio, Y. Binarized neural networks. *NeurIPS*, 29, 2016.

Jiang, Y., Konečný, J., Rush, K., and Kannan, S. Improving Federated Learning Personalization via Model Agnostic Meta Learning. *arXiv:1909.12488*, 2019.

Jiang, Y., Wang, S., Valls, V., Ko, B. J., Lee, W.-H., Leung, K. K., and Tassiulas, L. Model pruning enables efficient federated learning on edge devices. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., D'Oliveira, R. G. L., Eichner, H., Rouayheb, S. E., Evans, D., Gardner, J., Garrett, Z., Gascón, A., Ghazi, B., Gibbons, P. B., Gruteser, M., Harchaoui, Z., He, C., He, L., Huo, Z., Hutchinson, B., Hsu, J., Jaggi, M., Javidi, T., Joshi, G., Khodak, M., Konecný, J., Korolova, A., Koushanfar, F., Koyejo, S., Lepoint, T., Liu, Y., Mittal, P., Mohri, M., Nock, R., Özgür, A., Pagh, R., Qi, H., Ramage, D., Raskar, R., Raykova, M., Song, D., Song, W., Stich, S. U., Sun, Z., Suresh, A. T., Tramèr, F., Vepakomma, P., Wang, J., Xiong, L., Xu, Z., Yang, Q., Yu, F. X., Yu, H., and Zhao, S. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021. ISSN 1935-8237.

Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S., Stich, S., and Suresh, A. T. Scaffold: Stochastic controlled averaging for federated learning. In *ICML*, pp. 5132–5143. PMLR, 2020.

Khodak, M., Balcan, M.-F. F., and Talwalkar, A. S. Adaptive gradient-based meta-learning methods. In *NeurIPS*, pp. 5917–5928, 2019.

Krizhevsky, A. Learning multiple layers of features from tiny images. Msc thesis, 2009.

Li, A., Sun, J., Zeng, X., Zhang, M., Li, H., and Chen, Y. Fedmask: Joint computation and communication-efficient personalized federated learning via heterogeneous masking. In *SenSys*, pp. 42–55, 2021a.

Li, D. and Wang, J. Fedmd: Heterogenous federated learning via model distillation. 2019.

Li, T., Sahu, A. K., Talwalkar, A., and Smith, V. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

Li, T., Hu, S., Beirami, A., and Smith, V. Ditto: Fair and robust federated learning through personalization. In *ICML*, pp. 6357–6368. PMLR, 2021b.

Li, T., Hu, S., Beirami, A., and Smith, V. Ditto: Fair and robust federated learning through personalization. In *ICML*, pp. 6357–6368, 2021c.

Li, X., Huang, K., Yang, W., Wang, S., and Zhang, Z. On the convergence of fedavg on non-iid data. In *ICLR*, 2019.

Li, X., Jiang, M., Zhang, X., Kamp, M., and Dou, Q. Fed{bn}: Federated learning on non-{iid} features via local batch normalization. In *ICLR*, 2021d.

Liang, P. P., Liu, T., Ziyin, L., Salakhutdinov, R., and Morency, L.-P. Think locally, act globally: Federated learning with local and global representations. *arXiv preprint arXiv:2001.01523*, 2020.

Lin, T., Kong, L., Stich, S. U., and Jaggi, M. Ensemble distillation for robust model fusion in federated learning. *NeurIPS*, 33, 2020.

Luo, P., Ren, J., Peng, Z., Zhang, R., and Li, J. Differentiable learning-to-normalize via switchable normalization. *ICLR*, 2019.

Ma, J., Long, G., Zhou, T., Jiang, J., and Zhang, C. On the convergence of clustered federated learning. *arXiv preprint arXiv:2202.06187*, 2022.

Marfoq, O., Neglia, G., Bellet, A., Kameni, L., and Vidal, R. Federated multi-task learning under a mixture of distributions. *NeurIPS*, 34, 2021.

Maurer, A. and Pontil, M. Structured sparsity and generalization. *The Journal of Machine Learning Research*, 13 (1):671–690, mar 2012. ISSN 1532-4435.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, pp. 1273–1282. PMLR, 2017.

Meng, C., Rambhatla, S., and Liu, Y. Cross-node federated graph neural network for spatio-temporal data modeling. In *KDD*, pp. 1202–1211, 2021.

Minh, H. and Carl, K. Neurips 2021, workshop on new frontiers in federated learning. In *Personalized neural architecture search for federated learning*, 2021.

Muhammad, K., Wang, Q., O'Reilly-Morgan, D., Tragos, E., Smyth, B., Hurley, N., Geraci, J., and Lawlor, A. Fedfast: Going beyond average for faster training of federated recommender systems. In *KDD*, pp. 1234–1242, 2020.

Nguyen, J., Malik, K., Zhan, H., Yousefpour, A., Rabbat, M., Malek, M., and Huba, D. Federated learning with buffered asynchronous aggregation. In *International Conference on Artificial Intelligence and Statistics*, pp. 3581–3607. PMLR, 2022.

Ozkara, K., Singh, N., Data, D., and Diggavi, S. Quped: Quantized personalization via distillation with applications to federated learning. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, 2021.

Qin, Z., Yao, L., Chen, D., Li, Y., Ding, B., and Cheng, M. Revisiting personalized federated learning: Robustness against backdoor attacks. In *KDD*, 2023.

Qiu, X., Fernandez-Marques, J., Gusmao, P. P., Gao, Y., Parcollet, T., and Lane, N. D. ZeroFL: Efficient on-device training for federated learning with local sparsity. In *International Conference on Learning Representations*, 2022.

Reddi, S. J., Charles, Z., Zaheer, M., Garrett, Z., Rush, K., Konečný, J., Kumar, S., and McMahan, H. B. Adaptive federated optimization. In *ICLR*, 2021.

Reisizadeh, A., Mokhtari, A., Hassani, H., Jadbabaie, A., and Pedarsani, R. Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. In *AISTATS*, pp. 2021–2031. PMLR, 2020.

Rothchild, D., Panda, A., Ullah, E., Ivkin, N., Stoica, I., Braverman, V., Gonzalez, J., and Arora, R. Fetchsgd: Communication-efficient federated learning with sketching. In *ICML*, pp. 8253–8265. PMLR, 2020.

Sattler, F., Müller, K.-R., and Samek, W. Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *TNNLS*, 2020.

Shamsian, A., Navon, A., Fetaya, E., and Chechik, G. Personalized federated learning using hypernetworks. In Meila, M. and Zhang, T. (eds.), *ICML*, volume 139, pp. 9489–9502, 7 2021.

Singhal, K., Sidahmed, H., Garrett, Z., Wu, S., Rush, J., and Prakash, S. Federated reconstruction: Partially local federated learning. *Advances in Neural Information Processing Systems*, 34, 2021.

Smith, V., Chiang, C.-K., Sanjabi, M., and Talwalkar, A. Federated Multi-Task Learning. In *NeurIPS*, volume 30, pp. 4427–4437, 2017.

Tan, A. Z., Yu, H., Cui, L., and Yang, Q. Towards personalized federated learning. *arXiv preprint arXiv:2103.00710*, 2021.

Tianchun, W., Wei, C., Dongsheng, L., Wenchao, Y., Jingchao, N., Liang, T., Haifeng, C., and Zhang, X. Icdm. In *Personalized Federated Learning via Heterogeneous Modular Networks*, 2022.

Wang, Z., Kuang, W., Xie, Y., Yao, L., Li, Y., Ding, B., and Zhou, J. Federatedscope-gnn: Towards a unified, comprehensive and efficient package for federated graph learning. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 4110–4120, 2022.

Ward Jr, J. H. Hierarchical grouping to optimize an objective function. *JASA*, 58(301):236–244, 1963.

Xie, Y., Wang, Z., Gao, D., Chen, D., Yao, L., Kuang, W., Li, Y., Ding, B., and Zhou, J. Federatedscope: A flexible federated learning platform for heterogeneity. *Proceedings of the VLDB Endowment*, 16(5):1059–1072, 2023.

Yang, H., He, H., Zhang, W., and Cao, X. FedSteg: A Federated Transfer Learning Framework for Secure Image Steganalysis. *IEEE TNSE*, 2020.

Yang, Q., Liu, Y., Chen, T., and Tong, Y. Federated machine learning: Concept and applications. *arXiv: Artificial Intelligence*, 2019.

Yuan, H., Morningstar, W. R., Ning, L., and Singhal, K. What do we mean by generalization in federated learning? In *International Conference on Learning Representations*, 2022.

Yurochkin, M., Agarwal, M., Ghosh, S., Greenewald, K., and Hoang, N. Statistical model aggregation via parameter matching. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32, 2019a.

Yurochkin, M., Agarwal, M., Ghosh, S., Greenewald, K., Hoang, N., and Khazaeni, Y. Bayesian nonparametric federated learning of neural networks. In *International Conference on Machine Learning*, pp. 7252–7261. PMLR, 2019b.

Zhang, J., Guo, S., Ma, X., Wang, H., Xu, W., and Wu, F. Parameterized knowledge transfer for personalized federated learning. In *NeurIPS*, volume 34, 2021a.

Zhang, M., Sapra, K., Fidler, S., Yeung, S., and Alvarez, J. M. Personalized federated learning with first order model optimization. In *ICLR*, 2020.

Zhang, Q., Gu, B., Deng, C., Gu, S., Bo, L., Pei, J., and Huang, H. Asysqn: Faster vertical federated learning algorithms with better computation resource utilization. In *KDD*, pp. 3917–3927, 2021b.

Zhu, L. and Han, S. Deep leakage from gradients. In *NeurIPS*, volume 32. Springer, 2019.

Zhu, Z., Hong, J., and Zhou, J. Data-free knowledge distillation for heterogeneous federated learning. In *ICML*, 2021.

Appendix

We provide more details and further experiments about our work in the appendices:

- Sec.A: the summarized **algorithm** of the proposed method.
- Proofs: the **full proof** of Proposition 1 (Sec.B), Theorem 1 (Sec.C), and Theorem 2 (Sec.D). The discussions on the adopted **assumptions** are also presented in Sec.E.
- Sec.F: the detailed discussion about the **space-time complexity** comparing with other FL methods.
- Sec.G: the detailed comparison and discussion about more **related works**.
- Sec.H: the **implementation details** of experiments such as datasets, baselines, models, hyper-parameters and sparse model aggregation procedure of pFedGate.
- Sec.I: further **experimental results** including Sec.I.1, the performance in novel client participation case; Sec.I.2, the convergence study; Sec.I.3, the effect of s w.r.t. global-local model gap, FL communication frequency and data heterogeneity degree; and Sec.I.4, the model compression manner study, in which we compare pFedGate to the SOTA binary quantization method FedMask (Li et al., 2021a).

A. The Algorithm of the proposed pFedGate method

Algorithm 1 Efficient Personalized FL with pFedGate

Input: T, η_g, η, θ_g^0, $\{\phi_i^0\}_{i\in\mathcal{C}}$, s_{min}, $\{s_i\}_{i\in\mathcal{C}}$

1 **for** $t = 1, \cdots, T$ **do**

2 Server sends θ_g^{t-1} to clients $\mathcal{C}_s$ sampled from $\mathcal{C}$

for *client* $i \in \mathcal{C}_s$ *in parallel* **do**

3 **for** *data batch* $(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{S}_i$ **do**

4 Get sparse gated weight: $M_i' = g(\phi_i^{t-1}; \mathbf{x})$

Do personalized model-adaption:

$\theta_i^{'} = \theta_g^{t-1} \circledast M_i'$

Update global model and local gating layer:

$\theta_g^t = \theta_g^{t-1} - \eta_g(\nabla_\theta f(\theta_i^{'}; \mathbf{x}, \mathbf{y}))$

$\phi_i^t = \phi_i^{t-1} - \eta(\nabla_\phi \theta_i^{'} \nabla_\theta f(\theta_i^{'}; \mathbf{x}, \mathbf{y}))$

5 Upload sparse $\Delta\theta_{g,i}^t := \theta_g^t - \theta_g^{t-1}$

6 Server updates global model:

$\theta_g^t = \text{AGGREGATE}\left(\theta_g^{t-1}, \{\Delta\theta_{g,i}^t\}_{i\in\mathcal{C}_s}\right)$

7 **return** θ_g, $\{\phi_i\}_{i\in\mathcal{C}}$

We summarize the overall algorithm in Algorithm 1. Besides, we present more details about (1) the gradients flow via the gating layer, which contains a knapsack solver; and (2) the global model aggregation.

Differentiability. We leverage the straight-through trick (Hubara et al., 2016) to make the combination of the two flows predicted by the gating layer (red G and blue M in Figure 2) still differentiable. Recall that the combination is $M' = M \cdot I^*$ where I^* is a binary index and a solution to the knapsack problem (Section 5.2). The "straight-through" (ST) here means that we build such a computation graph, in which we only use binary variable I^* in the forward pass stage, while *replace I^* into the differentiable variable G in backward gradient propagation*. Specifically, this can be implemented by the following demonstrative PyTorch code: I^*_{ST} = I^* - G.detach() + G. In this way, the tensor I^*_{ST} and M are both differentiable and can be used to pass the gradient flow via their combination tensor $M' = I^*_{ST} * M$ and the following adapted tensor of the sparse model $\theta_i^{'} = \theta_g^{t-1} \circledast M_i'$. And thus we can train the personalized model and the gating layer as Algorithm 1, lines 6 and 7 shown.

Global Model Aggregation. We present more details about the global model aggregation of pFedGate. After local training, the client only uploads the updates of the parameters in the global model that are selected to form the client's local model. In other words, the client uploads the *non-zero* parameters updates with their corresponding *index* in the global model. The server then aggregates the received model updates according to their index.

Considering the following toy example: there are four clients, and the global model has a total of three parameter blocks. Each client uploads a dictionary whose key denotes the index and the value denotes the corresponding parameter updates.

- Client 1: 0: δ_0, 2: δ_2;
- Client 2: 1: δ_0, 2: δ_2;
- Client 3: 0: δ_0, 1: δ_2;
- Client 4: 0: δ_0, 1: δ_0, 2: δ_2.

After receiving the above updates, the server aggregates the parameter updates with index 0 by weighted averaging the δ_0s from clients 1, 3, and 4. The other parameters are aggregated similarly.

B. Proof of Proposition 1

Since f is μ-strongly convex, by definition we have

$$f(\theta') \geq f(\theta) + \nabla_\theta f(\theta)^T(\theta' - \theta) + \frac{\mu}{2}\|\theta' - \theta\|^2, \ \forall \theta, \theta'. \quad (8)$$

We have $f'(\theta) = f(\theta) - \frac{\mu}{2}||\theta||^2$ is also convex due to the first order condition of f with Equation (8). Considering

the monotone gradient condition for convexity of f', we get

$$(\nabla_\theta f(\theta) - \nabla_\theta f(\theta'))^T(\theta - \theta') \geq \mu\|\theta - \theta'\|^2, \ \forall \theta, \theta'. \quad (9)$$

Applying Cauchy-Schwartz inequality on Equation (9), we get

$$\begin{aligned} &\|\nabla_\theta f(\theta) - \nabla_\theta f(\theta')\|\|\theta - \theta'\| \\ &\geq (\nabla_\theta f(\theta) - \nabla_\theta f(\theta')^T(\theta - \theta') \\ &\geq \mu\|\theta - \theta'\|^2. \end{aligned} \quad (10)$$

Dividing $\|\theta - \theta'\|$ on both sides of Equation (10), we get

$$\|\theta - \theta'\| \leq \|\nabla_\theta f(\theta) - \nabla_\theta f(\theta')\|/\mu, \ \forall \theta, \theta'. \quad (11)$$

Under Assumption 1, for all (θ_i^*, θ_g^*) pairs with Equation (11), we have

$$\|(M_i^* - 1)\theta_g^*\|^2 \leq \cdot \frac{\sigma_i^2}{\mu^2}, \quad \forall i \in \mathcal{C}. \quad (12)$$

Using the reversed Cauchy-Schwarz inequality (Hardy et al., 1952), we have

$$A_i\|(M_i^* - 1) \circ \theta_g^*\| \geq \sqrt{\|(M_i^* - 1)^{\circ 4}\|\|(\theta_g^*)^{\circ 4}\|}, \quad (13)$$

where $\circ$ indicates element-wise multiplication, and $\circ 4$ indicates Hadamard power of 4, $A_i = \frac{1}{4}\frac{(R_{\theta_i}^2 R_{\theta_g}^2 + r_{\theta_i}^2 r_{\theta_g}^2)^2}{R_{\theta_i}^2 R_{\theta_g}^2 r_{\theta_i}^2 r_{\theta_g}^2}$ is a positive constant and $\leq \frac{1}{4}\frac{R_{\theta_i}^2 R_{\theta_g}^2}{r_{\theta_i}^2 r_{\theta_g}^2}$, r_{θ_i} and r_{θ_g} are the infimum of θ_i and θ_g respectively, R_{θ_i} and R_{θ_g} are the supremum of θ_i and θ_g respectively.

Combining Equation (12) and Equation (13), we get the upper bound in Propostion 1 and finish the proof.

C. Proof of Theorem 1

Let $\mathbb{H}^n$ denote the function space with its elements parametrized by θ_g, ϕ_1, $\cdots$, ϕ_n and the distance metric is defined as:

$$\begin{aligned} &d((\theta_g, \phi_1, \cdots, \phi_n) - (\theta_g^{'}, \phi_1^{'}, \cdots, \phi_n^{'})) \\ =&\frac{1}{n}\mathbb{E}_{x,y\sim\mathcal{D}_i}\left[\left|\sum(f(\theta_g, \phi_i; x, y) - \sum f(\theta_g^{'}, \phi_i^{'}; x, y)\right|\right], \end{aligned} \quad (14)$$

With the Lipshitz conditions in Assumption 2, we have:

$$\begin{aligned} &d((\theta_g, \phi_1, \cdots, \phi_n) - (\theta_g^{'}, \phi_1^{'}, \cdots, \phi_n^{'})) \\ \leq&\sum_i \frac{1}{n}\mathbb{E}_{x,y\sim\mathcal{D}_i}\left[\ell(h_{\theta_g,\phi_i}(x), y) - \ell(h_{\theta_g^{'},\phi_i^{'}}(x), y)\right] \\ \leq&L_f||h_{\theta_g,\phi_i} - h_{\theta_g^{'},\phi_i^{'}}|| \\ =&L_f||h_s(\theta_i) - h_s(\theta_i^{'})|| \\ \leq&L_f L_h||\theta_g M_i - \theta_g^{'} M_i^{'}|| \qquad \text{(Lipshitz condition)} \\ \leq&L_f L_h||\theta_g M_i - \theta_g M_i^{'} + \theta_g M_i^{'} - \theta_g^{'} M_i^{'}|| \\ \leq&L_f L_h\left[||\theta_g|| \cdot ||M_i - M_i^{'}|| + ||M_i^{'}|| \cdot ||\theta_g - \theta_g^{'}||\right] \\ &\qquad \text{(Assumption 2)} \\ \leq&L_f L_h\left[R||M_i - M_i^{'}|| + s_i||\theta_g - \theta_g^{'}||\right] \\ \leq&L_f L_h\left[R||g(\phi_i) - g(\phi_i^{'})|| + s_i||\theta_g - \theta_g^{'}||\right] \\ &\qquad \text{(Lipshitz condition)} \\ \leq&L_f L_h\left[RL_g||\phi_i - \phi_i^{'}|| + s_i||\theta_g - \theta_g^{'}||\right], \end{aligned} \quad (15)$$

where R is radius of the ball that can bound the parameters of global model and the gating layer (mentioned in Assumption 2). We can get an ϵ-covering in metric $d((\theta_g, \phi_1, \cdots, \phi_n) - (\theta_g^{'}, \phi_1^{'}, \cdots, \phi_n^{'}))$ if we select a covering of in the parameter space with both $||\phi_i - \phi_i^{'}||$ and $||\theta_g - \theta_g^{'}||$ equal to $\frac{\epsilon}{L_f L_g(RL_g + s_i)}$. Therefore, the covering number of $\mathbb{H}^{|\mathcal{C}|}$, denoted as $\mathcal{B}(\epsilon, \mathbb{H}^{|\mathcal{C}|})$ is: $\log(\mathcal{B}(\epsilon, \mathbb{H}^{|\mathcal{C}|}) = \mathcal{O}\left((|\mathcal{C}|d_\phi + d)\log\frac{RL_f L_h(RL_g + s_i)}{\epsilon}\right)$.

According to (Baxter, 2000; Shamsian et al., 2021), there exit $\tilde{N}$ and $\tilde{N} = \mathcal{O}\left(\frac{1}{n\epsilon^2}\log\frac{\mathcal{B}(\epsilon, \mathbb{H}^{|\mathcal{C}|})}{\delta}\right) = \mathcal{O}(\frac{d}{|\mathcal{C}|\epsilon^2}\log\frac{RL_f L_h(RL_g + s_i)}{\epsilon} + \frac{d_\phi}{\epsilon^2}\log\frac{RL_f L_h(RL_g + s_i)}{\epsilon} - \frac{\log\delta}{|\mathcal{C}|\epsilon^2})$.

D. Proof of Theorem 2

We note that (Marfoq et al., 2021) extends the surrogate optimization into the FL setting and provides convergence results for the algorithms which approximate the target objective functions using first-order surrogate or partial first-order surrogate functions. For simplicity, we use two more compact notations for all the learnable parameters introduced in our formulation as $\vec{v}_{M'} \triangleq \{M_i\}_{i\in\mathcal{C}}$ and $\vec{v}_{M',\phi} \triangleq \{M_i^{'} = g(\phi_i; x)\}_{i\in\mathcal{C}, x\in X_i}$. Here we prove that our method described in Algorithm 1 can be regarded as to optimize the objective function $F(\theta_g, \vec{v}_{M'})$ in Equation (5) with *another partial first-order surrogate function*, and the convergence results from (Marfoq et al., 2021) can be applied into our method. We first recap the formal definition of partial first-order surrogate:

Definition 1 (Partial first-order surrogate (Marfoq et al., 2021)). *A function $F'(\vec{u}, \vec{v}) : \mathbb{R}^{d_u} \times \mathcal{V} \to \mathbb{R}$ is a partial-first-order surrogate of $F(\vec{u}, \vec{v})$ wrt $\vec{u}$ near $(\vec{u}_0, \vec{v}_0) \in R^{d_u} \times \mathcal{V}$ when the following conditions are satisfied:*

1. *$F'(\vec{u}, \vec{v}) \geq F(\vec{u}, \vec{v})$ for all $\vec{u} \in \mathbb{R}^{d_u}$ and $\vec{v} \in \mathcal{V}$;*
2. *$e(\vec{u}, \vec{v}) \triangleq F'(\vec{u}, \vec{v}) - F(\vec{u}, \vec{v})$ is differentiable and L-smooth with respect to $\vec{u}$. Moreover, we have $e(\vec{u}_0, \vec{v}_0) = 0$ and $\nabla_{\vec{u}} e(\vec{u}_0, \vec{v}_0) = 0$.*
3. *$F'(\vec{u}, \vec{v}_0) - F'(\vec{u}, \vec{v}) = d_{\mathcal{V}}(\vec{v}_0, \vec{v})$ for all $\vec{u} \in \mathbb{R}^{d_u}$ and $\vec{v} \in \arg\min_{\vec{v}' \in \mathcal{V}} F'(\vec{u}, \vec{v}')$, where $d_{\mathcal{V}}$ is non-negative and $d_{\mathcal{V}}(\vec{v}, \vec{v}') = 0 \iff \vec{v} = \vec{v}'$.*

Definition 1 indicates that in the partial parameter set $\mathcal{V}$, F' is *majorant* to F and the *smoothness* is satisfied in the neighborhood of a given point.

Denote $F_i'(\theta_g, \vec{v}_{M',\phi})$ as the transformed local objective function depicted by the gating layer ϕ_i, *i.e.*, the sparse gated weight $M_i^{'}$ is generated by the process described in Section 5.2 and intermediate objective in Equation (6). We then restrict our attention from the function $F(\theta_g, \vec{v}_{M'})$ over all clients into $|\mathcal{C}|$ local functions $F_i(\theta_g, \vec{v}_{M',\phi})$ for each client i, since the weighted sum operation is convex and holds the partial majorization and smoothness properties. Consider the iterative version of F_i and F_i' corresponding to the produce in Algorithm 1. We have the partial set of gated weights $\mathcal{V}_\phi = \{\{M_i^{'} = g(\phi_i^t; x)\}_{x \in X_i}\}_{\phi_i^t = \arg\min_\phi (F_i'(\theta_g^{t-1}, \vec{v}_\phi^{t-1}))}$ for all $t \in [T]$. For $\vec{v}_{M',\phi_i^t} \in \mathcal{V}_\phi$, due to the optimality of ϕ_i^t and equality between $\vec{v}_{M'}$ and $\vec{v}_{M',\phi_i^t}$ using block-to-element scatter operation on the gated weights, we have $F_i'(\theta_g, \vec{v}_{M'}) = F_i(\theta_g, \vec{v}_{M',\phi_i^t})$ and Condition 1 satisfied.

Note that the difference between $F_i'(\theta_g, \vec{v}_{M'})$ and $F_i(\theta_g, \vec{v}_{M',\phi})$ is differentiable w.r.t. θ_g since both of them is differentiable by adopting either element-wise or block-wise production on θ_g and the sparse gated weights. With the smoothness Assumption 3, we have Condition 2 satisfied. For Condition 3, it is clear that if the right statement holds, *i.e.*, $\vec{v}_{M',\phi} = \vec{v}'_{M',\phi}$, then $d_{\mathcal{V}}(\vec{v}_{M',\phi}, \vec{v}'_{M',\phi}) = F_i'(\theta_g, \vec{v}_{M',\phi}) - F_i'(\theta_g, \vec{v}'_{M',\phi}) = 0$ and we have the sufficiency between the two statements of Condition 3 satisfied. For the necessity between the two statements of Condition 3, we can consider such a set $\mathcal{V}_\phi \setminus (\tilde{\mathcal{V}}_\phi \cup \overline{\mathcal{V}}_\phi)$, where $\tilde{\mathcal{V}}_\phi$ indicates the subset of $\mathcal{V}_\phi$ in which $t \in [\tilde{T}, T]$ if F_i' converges at step $\tilde{T} \leq T$, and $\overline{\mathcal{V}}_\phi$ indicates the subset of $\mathcal{V}_\phi$ in which $t \in \{t' | F_i'(\theta_g^{t-1}, \vec{v}_\phi^{t-1}) = F_i'(\theta_g^{t'-1}, \vec{v}_\phi^{t'-1}), \forall t, t' \in [T], t' \neq t\}$. The new partial subset ensures the optimal $\vec{v}_{M',\phi}$ is unique by discarding the redundant $\vec{v}_{M',\phi}$ that have the same $F_i'(\theta_g, \vec{v}_{M',\phi})$. For such a new partial subset $\mathcal{V}_\phi \setminus (\tilde{\mathcal{V}}_\phi \cup \overline{\mathcal{V}}_\phi)$, we have the necessity between the two statements of Condition 3 satisfied due to the uniqueness of the solutions of $\arg\min_{\vec{v}'_{M',\phi} \in \mathcal{V}_\phi \setminus (\tilde{\mathcal{V}}_\phi \cup \overline{\mathcal{V}}_\phi)} F'(\vec{\theta_g}, \vec{v}'_{M',\phi})$, and the Condition 1 and Condition 2 still hold since $\mathcal{V}_\phi \setminus (\tilde{\mathcal{V}}_\phi \cup \overline{\mathcal{V}}_\phi)$ is a subset of $\mathcal{V}_\phi$.

We finally show that the assumptions required by the convergence proof in (Marfoq et al., 2021) hold when using our approach.

Proposition 2. *Under the assumptions 1–5, when taking the parameter from θ_g to $\theta_{\phi_i} = \theta_g \circledast \phi_i(x)$ over all $i \in \mathcal{C}, x \in X_i$, the gradient variance of f and diversity across clients is also bounded, and the smoothness of f holds.*

Proof. Note that using the chain rule, the gradients on ϕ_i can be written as $\nabla_\phi(f(\theta_{\phi_i}; x, y)) = \nabla_\phi \theta_{\phi_i} \nabla_\theta f(\theta_{\phi_i}; x, y)$ for $(x, y) \in \mathcal{D}_i$. For simplicity, we denote $\nabla_\phi(f(\theta_{\phi_i}; x, y)) \triangleq f(\theta_{\phi_i})$. Since we generate sparse model as $\theta_{\phi_i} = \theta_g \circledast M_i^{'}$, we have $\nabla_\phi \theta_{\phi_i} = M_i^{'} \nabla_\phi M_i^{'}$. The block-wise production operation $\circledast$ is differentiable and f over θ_g is differentiable according to Assumption 3, thus f over ϕ_i is also differentiable. Then recall that the optimal gated weights are bounded as shown in Proposition 1. Further, we explicitly bound the sparse gated weight during FL training process in $[0, 1]^L$ using the sigmoid function as introduced in Section 5.2. We thus can assume that

$$||\nabla_\phi \theta_{\phi_i'} - \nabla_\phi \theta_{\phi_i}|| \leq L_\phi ||\theta_{\phi_i'} - \theta_{\phi_i}||^2, \tag{16}$$

where L_ϕ is a Lipshitz constant corresponding to our bounded gated weights. We get

$$\begin{aligned}
&||\nabla_\phi f(\theta_{\phi_i'}) - \nabla_\phi f(\theta_{\phi_i})||^2 \\
=&||\nabla_\phi \theta_{\phi_i'} \nabla_\theta f(\theta_{\phi_i'}) - \nabla_\phi \theta_{\phi_i} \nabla_\theta f(\theta_{\phi_i})||^2 \quad \text{(Chain Rule)} \\
=&||\nabla_\phi \theta_{\phi_i'} \nabla_\theta f(\theta_{\phi_i'}) - \nabla_\phi \theta_{\phi_i'} \nabla_\theta f(\theta_{\phi_i}) + \\
&\nabla_\phi \theta_{\phi_i'} \nabla_\theta f(\theta_{\phi_i}) - \nabla_\phi \theta_{\phi_i} \nabla_\theta f(\theta_{\phi_i})||^2 \\
\leq&||\nabla_\phi \theta_{\phi_i'} \nabla_\theta f(\theta_{\phi_i'}) - \nabla_\phi \theta_{\phi_i'} \nabla_\theta f(\theta_{\phi_i})||^2 + \\
&||\nabla_\phi \theta_{\phi_i'} \nabla_\theta f(\theta_{\phi_i}) - \nabla_\phi \theta_{\phi_i} \nabla_\theta f(\theta_{\phi_i})||^2 \\
\leq&L||\nabla_\phi \theta_{\phi_i'}||^2 ||\theta_{\phi_i'} - \theta_{\phi_i}||^2 + \\
&L_\phi ||\nabla_\theta f(\theta_{\phi_i})||^2 ||\theta_{\phi_i'} - \theta_{\phi_i}||^2 \\
\leq&2LL_\phi ||\theta_{\phi_i'} - \theta_{\phi_i}||^2, \quad \text{(Assumption 3 \& Eqn. (16))}
\end{aligned} \tag{17}$$

where the last two lines are because of that Lipschitz implies bounded gradients. Thus we get that f is $(2LL_\phi)$-smooth using our method. Similar to the proof of smoothness that leverages the chain rule and bounded M_i, we can generalize Assumption 1 and Assumption 5 for our method such that the gradients dissimilarity is bounded and f has bounded variance characterized by another constant factor. □

Now we see that each client can optimize a partial first-order surrogate function and the necessary assumptions satisfied with Proposition 2. Applying the Theorem $3.2^{'}$ in (Marfoq et al., 2021), we conclude the proof of our Theorem 2.

E. Discussion on the Assumptions

We note that the adopted theoretical assumptions are reasonable, fairly mild in FL scenarios and widely used by numerous related FL works. To recap, there are five Assumptions including assumption 1 (bounded data diversity), assumption 2 (bounded parameters of the global model and the gating layer, 3 (smoothness of gradients), 4 (bounded output of f) and 5 (bounded gradients variance).

And there are three theoretical results: the motivated Proposition 1 is based on Assumption 1; the Theorem 1 (generation) is based on Assumption 2, the Theorem 2 (convergence) is based on Assumption 1 $\sim$ 5.

- About the Assumption 1 (used in Proposition 1 and Theorem 2), it is necessary to a feasible and effective FL process, in which there is knowledge that can be shared and mutually beneficial between the FL participants. We would like to clarify that we adopt a weaker form than the similar assumption that is also widely used in related literature such as (Ma et al., 2022; Dinh et al., 2020b; Marfoq et al., 2021; Liang et al., 2020). For example, let's consider assumption 3 in (Dinh et al., 2020b): "(Bounded diversity). The variance of local gradients to global gradient is bounded as $\frac{1}{N}\sum_{i=1}^{N}||\nabla f_i(w)-\nabla_\theta f(w)||^2 \leq \sigma_f^2$.". By contrast, our assumption is "There exists a such that the variance between local and global gradients is bounded as $||\nabla_\theta f(\theta_i^*) - \nabla_\theta f(\theta_g^*)||^2 \leq \sigma_i^2, \quad \forall i \in \mathcal{C}$". Note that our assumption is actually **a special case of their form** on sparse model, with our $f(\theta_i^*)$ term corresponding to their $f_i(w)$ term, and our $f_i(w)$ term corresponding to their $f(w)$ term.

- As for the assumption 2 (used in Theorem 1), it is dependent on the hypothesis space of the adopted global model and gating layer, which are controllable by users. For example, we can hard clip their model parameters, and introduce bounded activation functions such as sigmoid into the gating layer structure as the proposed method does. A more specific example of this is our implementation, where we use zero initialization, kaiming-uniform initialization and (0, 1)-uniform initialization for the parameters of the adopted global model and gating layer, which leads to all these init parameters are bounded by one of 0, 1, or constants calculated by the bounded *fan_in* (usually determined by the data shape). Then during the FL courses, we clip the gradients at each backpropagation step (via *torch.nn.utils.clip_grad_norm_()*), resulting in the bounded parameters and thus the assumption holds.

  Besides, assumptions similar to our assumption 2 can be found in the related PFL works, such as pFedHN (Shamsian et al., 2021), in which the authors assume that their introduced weights of the hyper-network and the embeddings are bounded and follow several Lipschitz conditions with constants different from ours.

- The assumptions 3 and 4 are easily satisfied for most discriminative models such as neural networks (Tan et al., 2021; Kairouz et al., 2021; Li et al., 2019). And the assumption 5 is necessary to a feasible and effective FL course, in which there is knowledge that can be shared and mutually beneficial between the FL participants (Ma et al., 2022; Dinh et al., 2020b; Liang et al., 2020; Marfoq et al., 2021).

- As for the strong convexity, it is only used only in Proposition 1, which shows the motivation of the proposed method, and the detailed upper bound of M_i^* is not involved in our remaining theoretical analysis (Theorem 2 and Theorem 1). Although many SOTA FL and PFL works and PFL works with provable analysis are also based on the strong convexity of loss function, we leave it as future work to relax the convexity assumption.

F. Space-Time Complexity

We now examine the efficiency of pFedGate described in Algorithm 1 in terms of computation, communication, and storage costs. Table 5 summarizes the costs for pFedGate and competitors including FedAvg and two SOTA personalized FL methods, where the training and inference costs on clients are considered for each training batch (or each sample when the batch size is 1), and the communication cost is considered for each FL round. We also mark some specific example values when taking the default hyper-parameters of different methods in Table 5. For comparison simplicity, here we assume that for the computation costs, the inference and back-propagation times are proportional to the model size. We hide both constants and polylogarithmic factors dependent on the specific model architectures in $\mathcal{O}(\cdot)$, since we compare these methods with the same model architectures and the same trained data samples. As for communication and storage, $\mathcal{O}(\cdot)$ hides constants and lower-order factors dependent on specific implementations such as platforms, storage, and communication protocols.

Computation. Recall that our method is based on structured block sparsity (Sec.5.2.2), i.e., pFedGate will remove some blocks and generate a sparse sub-network. It actually reduces the computation cost in both inference and training even without sparse matrix computing libraries. The training and inference cost of pFedGate is $\mathcal{O}\big(d(s_i{+}s_{\phi_i})\big)$ for client i at each round.

Communication. In the communication stage, we do not need to transmit the zero sub-blocks (a detailed example is in Sec.A) with very light additional meta-data for indexing. As a result, the pFedGate uploads parameters in $\mathcal{O}(q_i d)$, where $q_i = count(\Delta\theta_i \neq 0)/d$ indicates the ratio of non-zero model updates. Specifically, q_i can be regarded as the proba-

Table 5: The proposed pFedGate achieves better computation, communication and storage complexity over state-of-the-art personalized FL methods. The underlined results indicate the examples when taking default hyper-parameters: the embedding and hyper-network size $d_v = 0.25|\mathcal{C}|, d_h = 100$ for pFedHN (Shamsian et al., 2021), the component model number $k = 3$ for FedEM (Marfoq et al., 2021), the client sparsity $s_i = 0.5$, and relative sparsity of gating layer $s_{\phi_i} = d/2d_X L = 0.05$ for pFedGate. Besides, for pFedGate, we report the average non-zero parameter ratio $q_i = 0.67$, which is dependent on datasets and local training steps. Here we report the average value of q_i when we run FL experiments 3 times on FEMNIST and CIFAR10 with $s_i = 0.5$, local update step as 1 epoch, and batch size as 128.

		FedAvg	pFedHN	FedEM	pFedGate
Computation	Train (Client)	$\mathcal{O}(2d)$	$\mathcal{O}(2d)$	$\mathcal{O}(2kd)$ $\underline{\mathcal{O}(6d)}$	$\mathcal{O}\big(2d(s_i + s_{\phi_i})\big)$ $\underline{\mathcal{O}(1.1d)}$
	Infer (Client)	$\mathcal{O}(d)$	$\mathcal{O}(d)$	$\mathcal{O}(d)$	$\mathcal{O}\big(d(s_i + s_{\phi_i})\big)$ $\underline{\mathcal{O}(0.55d)}$
	Train (Server)	$\mathcal{O}(1)$ $\underline{\mathcal{O}(6d)}$	$\mathcal{O}\big((d_v + d_h)\lvert\mathcal{C}_s\rvert\big)$ $\underline{\mathcal{O}\big((0.25\lvert\mathcal{C}\rvert+100)\lvert\mathcal{C}_s\rvert\big)}$	$\mathcal{O}(1)$	$\mathcal{O}(1)$
Communication	Client	$\mathcal{O}(d)$	$\mathcal{O}(d)$	$\mathcal{O}(kd)$ $\underline{\mathcal{O}(3d)}$	$\mathcal{O}(q_i d)$ $\underline{\mathcal{O}(0.67d)}$
	Server	$\mathcal{O}(d\lvert\mathcal{C}_s\rvert)$	$\mathcal{O}(d\lvert\mathcal{C}_s\rvert)$	$\mathcal{O}(kd\lvert\mathcal{C}_s\rvert)$ $\underline{\mathcal{O}(3d\lvert\mathcal{C}_s\rvert)}$	$\mathcal{O}(d\lvert\mathcal{C}_s\rvert)$
Storage	Client (Peak Memory)	$\mathcal{O}(rd)$	$\mathcal{O}(r'd)$	$\mathcal{O}(krd)$	$\mathcal{O}(d + r'' s_{\phi_i} d)$
	Client (Disk)	$\mathcal{O}(d)$	$\mathcal{O}(d)$	$\mathcal{O}(kd)$ $\underline{\mathcal{O}(3d)}$	$\mathcal{O}\big((1 + s_{\phi_i})d\big)$ $\underline{\mathcal{O}(1.05d)}$
	Server	$\mathcal{O}(d)$	$\mathcal{O}(d_v d_h + d_v\lvert\mathcal{C}\rvert)$ $\underline{\mathcal{O}(25\lvert\mathcal{C}\rvert+0.25\lvert\mathcal{C}\rvert^2)}$	$\mathcal{O}(kd)$ $\underline{\mathcal{O}(3d)}$	$\mathcal{O}(d)$

bility that a parameter is finally not masked in the local training process. We can consider $q_i = \Big(1 - \prod_{j=1}^{|\mathcal{S}|}(1 - p_j)\Big)$, where p_j indicates the probability that a parameter is not within the sub-blocks the gating layer predicts to mask when taking the j-th data batch as input, and $|\mathcal{S}|$ indicates the number of data batches trained in each local FL round. Note that p_1 is equal to the sparsity factor s_i at client i, and p_j becomes smaller as j increases, due to the fact that the local data batches usually share some similarities, and the newly predicted sub-blocks are more likely to be the same as the sub-blocks have been selected.

Storage. (1) For the **peak memory** cost on clients, note that FL requires training on client-side, in addition to the model parameters, the memory cost in FL training process also contains the intermediate tensors in both the forward and the backward propagation phases, whose size depends on the model architecture, the shape of the inputs, and the specific implementation (we abstract this factor as r, r', r'' for FedAvg, pFedHN and pFedGate respectively in Table 5). Thus once the model-adaption of pFedGate is completed, the intermediate tensor involved in the forward and backward processes through the sparse model is correspondingly smaller based on block sparsity, allowing us to get a lower peak training memory than compared methods (empirical results are in Section 7.4). (2) For the **disk** storage, compared FedAvg, the introduced additional storage overhead of pFedGate is very small as the gating layer structure has a simple and small structure, whose shape is about the multiplication of the dimension of the input feature and the number of the total sub-blocks. For example, the size of gating layer d_{ϕ_i} is only 6.3% of the original model for the CNN model used on CIFAR-10. We would like to point out that pFedGate actually has *great potential for further storage savings for multi-task applications*. Considering that if we use one dense original model but learn E distinct gating layers for different E downstream tasks on a client, the storage cost will be $d + T * d_{\phi_i}$, which has a much smaller model storage cost than saving multiple distinct local models. It is promising to extend the current batch-level adaptation to task-level adaptation to further reduce the storage cost, and we leave it as future work.

Cross-methods Comparison. From Table 5 we can see that pFedGate achieves better efficiency than FedAvg in training, inference, and uploading with negligible additional storage cost on clients (as marked in red). By contrast, to achieve good personalization, FedEM introduces more than one global model and pays larger costs in training, communication, and storage (as marked in blue). And pFedHN learns client-wise embedding in the server, which may lead to a single-point bottleneck for cross-device settings and potentially disclosing personal information. To further verify the analysis, we present empirical supports in Section 7.4.

G. More Related Work Comparison

Here we present more details about the works related to our method. QuPeD (Ozkara et al., 2021) combines quantization and knowledge distillation for personalized federated learning. FedMask (Li et al., 2021a) freezes the local models during the whole FL process, learns distinct binary masks for the last several layers of the models, and only transmits and aggregates the learned masks. Different from the quantization-based methods, we adopt continuous gated weights on all sub-blocks of the local model and transmit the sparse model updates, enabling flexible model size reduction and high capability to capture the client relationships via the shared global model.

FedRep (Collins et al., 2021) proposes to upload partial sub-parameters as shared representation and leave the others locally trained. HereroFL (Diao et al., 2021) selects different subsets of global model parameters for clients based on their computational capabilities. Different from these model-decoupling-based works that use the same parameter subset for those clients having the same computation capability, our method generates different sub-models from the whole global model with a larger learnable parameter space to handle heterogeneous data distribution. More importantly, our method differs from these works by adaptively generating the sparse model weights at a fine-grained batch level, which achieves a good estimation for the conditional probability of heterogeneous local data and high accuracy as shown in our experiments.

FedPNAS (Minh & Carl, 2021) leverages neural architecture search (NAS) to search suitable sub-networks for clients, and FedMN (Tianchun et al., 2022) proposes to use routing-hypernetwork to select personalized sub-blocks. For the FedPNAS, we differ from it in the target and studied problem. FedPNAS mainly focuses on improving the personalization performance with sufficient hardware resources. For example, they adopt 5 clients in their experiments that usually correspond to the cross-silo case. While pFedGate focuses on the cross-device setting, where the clients' resources are usually very limited. Compared with FedMN, the performance of pFedGate is theoretically guaranteed for both generalization and convergence. While FedMN doesn't provide any theoretical analyses for generalization or convergence. Besides, the proposed pFedGate method adopts a much larger hypotheses space than FedMN. In FedMN, each modular is either connected or dropped. While pFedGate doesn't only decide the connections of the network (connected or dropped), but also scales the weights of the remained blocks simultaneously to improve the personalized performance.

There are FL works considered sparsity, FedDST (Bibikar et al., 2022) and ZeroFL (Qiu et al., 2022). However, we are different from them in several aspects: First, from the view of the studied problem, both of these two works focus on non-personalized FL that finds a single global model to perform well on all clients, while we consider the personalized case to learn client-wise models. Second, both of them don't provide any theoretical analyses for generalization or convergence, while pFedGate performs well with theoretical guarantees in terms of both generalization and convergence. Third, from the view of methodology, they are based on element-wise unstructured sparsity while ours is based on block-wise structured sparsity, which is easier to gain efficiency benefits in extended implementations. Finally, from the experimental results, we can see that FedDST works in sparsity with 0.5 or 0.8 (Figure 3 in their paper) while ours works in even s=0.1. Another method ZeroFL reported the results of sparsity 0.1 on CIFAR10 and gained 0.42% accuracy improvement compared to vanilla FedAvg (their Table 1, NIID column), while we gained 3.02% improvement over FedAvg with sparsity 0.1 (our Table 1).

We also compared meta-learning-based PFL methods including Per-FedAvg (Fallah et al., 2020) and pFedMe (Dinh et al., 2020b). Per-FedAvg leverages model agnostic meta-learning to enable fast local personalized training. However, it requires computationally expensive second-order gradients, which challenges real-world applications, especially with limited system resources. pFedMe improves Per-FedAvg with faster convergence and smaller computation complexity. We empirically show that our methods can achieve better performance than theirs in Table 6.

Besides, knowledge distillation-based methods can also deal with hardware heterogeneity by allowing different local models to use different architectures. However, many existing KD-based FL/PFL works rely on some common reference object shared among clients, to enable knowledge alignment between the teacher and student models. For example, works (Li & Wang, 2019; Zhu et al., 2021) introduce additional public data for all clients and perform FL with the help of the model logits on the public data. It's worth noticing that such additional common references may not always be available in some scenarios. Further, we conduct empirical comparisons to a known KD-based FL method, FedMD (Li & Wang, 2019). We list the results on FEMNIST and CIFAR-10 in Table 6, where "FedMD, Far-Trans" indicates a far-transfer case by adopting the CIFAR-10 as the public dataset for FEMNIST experiment, and adopting the FEMNIST as the public dataset for CIFAR-10 experiment. The "FedMD, Near-Trans" indicates a near-transfer case by adopting the EMNIST as the public dataset for FEMNIST experiment, and adopting the CIFAR-100 as the public dataset for CIFAR-10 experiment. We find that pFedGate still shows superiority in terms of the average/bottom accuracy, and robustness when we vary the sparsity for pFedGate and vary the public dataset for FedMD.

Table 6: Performance comparison to some baselines in FEMNIST and CIFAR-10 datasets.

	FEMNIST		CIFAR-10	
	$\overline{Acc}$	$\widetilde{Acc}$	$\overline{Acc}$	$\widetilde{Acc}$
FedAvg	76.51	60.82	68.33	60.22
Per-FedAvg	77.18	61.42	70.35	60.96
pFedMe	75.29	57.63	70.29	62.13
FedMD, Far-Trans	61.25	51.37	55.28	52.04
FedMD, Near-Trans	75.87	60.29	67.52	59.43
FedEM	80.12	64.81	72.43	62.88
pFedGate, s=1	87.32	77.14	75.18	66.67
pFedGate, s=0.5	86.31	75.68	74.07	64.21
pFedGate, s=0.3	86.76	76.47	73.65	64.39

H. Implementation Details

Datasets. We conduct experiments on several widely used FL datasets including EMNIST (Cohen et al., 2017) and FEMNIST (Caldas et al., 2018) for 62-class handwritten character recognition, and CIFAR10/CIFAR100 (Krizhevsky, 2009) for 10-class/100-class image classification. We follow the heterogeneous partition manners used in (Marfoq et al., 2021; Caldas et al., 2018; Diao et al., 2021; Fallah et al., 2020): FEMNIST was partitioned by writers and we generate the other three FL datasets using Dirichlet allocation of parameter α=0.4. We adopt 10% sub-samples of EMNIST (81,425 samples) and 15% sub-samples of FEMNIST (98,671 samples), and allocate the sub-datasets to 100 and 539 clients for EMNIST and FEMNIST respectively. For CIFAR10/CIFAR100 (60,000 samples), we allocate them to 100/50 clients. All datasets are randomly split into train/valid/test sets with a ratio 6:2:2.

Baselines and models. We consider the following competitive baselines including SOTA personalized and efficient FL methods in this work:

- *FedAvg* (McMahan et al., 2017): the classical FL method that simply aggregates model updates in a weighted averaging manner;
- *FedAvg-FT*: a basic personalized baseline that fine-tunes the global model with local data before model local evaluation;
- *Local*: a naive personalized method that trains models on local datasets without FL communication;
- *pFedMe* (Dinh et al., 2020b) decouples the personalized model and global model with Moreau envelops based regularization;
- *LG-FedAvg* (Liang et al., 2020) achieves personalization with improved communication efficiency;
- *Ditto* (Li et al., 2021b) is a SOTA PFL method that introduces a local personalized model for each client, which is trained with model parameter regularization according to the global model;
- *FedEM* (Marfoq et al., 2021) deals with data heterogeneity via the mixture of multiple global models. We use 3 models according to the authors' default choice;
- *FedRep* (Marfoq et al., 2021) proposes to improve the FL performance and communication efficiency via sharing partial model parameters among FL participants. Only the low layers of models for feature extraction are uploaded to the server and aggregated, and the remaining head parameters are locally trained for personalization. Following the authors' sharing manner, we adopt the last classification layers as local personalized model parameters;
- *FedMask* (Li et al., 2021a) learns distinct binary masks for the last several layers of the local models, and aggregates the masks with an intersection operation. Following the mask layer choice manner similar to the authors, in our experiments, the one-shot mask is applied in the last layer for the adopted 2-layer CNN and the last 2 layers for the adopted 3-layer CNN respectively;
- *HeteroFL* (Diao et al., 2021) uses heterogeneous model architectures for different clients to achieve personalization and improvements in both computational and communication efficiency. Recall that we can vary each client's sparse ratio s to reflect the clients' resource heterogeneity, since the affordable sparse ratio s can fairly reflect the client's computation and communication capabilities. Clients with very limited computation and communication resources have to set s close to 0; On the contrary, clients with sufficient resources can set s close to 1. Here we use the full model parameters for 10% clients and 50% parameters for the other 90% clients following the authors' computation complexity setting, which leads to an average sparsity $\bar{s}' = 0.55$.

To align with previous works, we use a 2-layer CNN (Reddi et al., 2021; Marfoq et al., 2021) for EMNIST/FEMNIST, and two LeNet-based CNNs (Dinh et al., 2020b; Liang et al., 2020; Shamsian et al., 2021) with different capabilities for CIFAR10 and CIFAR100. Specifically, the model used for EMNIST/FEMNIST two convolutional layers with 5 × 5 kernels, max pooling, and two dense layers with a total of 2,171,786 parameters. The models used for CIFAR10 and CIFAR100 have a LeNet-based structure with an additional linear classification layer, and a total of 256,830 and 3,537,444 parameters respectively.

Table 7: The adopted learning rates for pFedGate in all datasets.

	EMNIST		FEMNIST		CIFAR10		CIFAR100	
	η_g	η	η_g	η	η_g	η	η_g	η
s=1	0.1	0.1	0.1	0.1	0.03	0.05	0.1	1.5
s=0.5	0.1	0.1	0.1	0.05	0.03	1.5	0.03	1.5
s=0.3	0.03	0.5	0.3	0.05	0.03	1.5	0.05	0.5

Platform and Hyper-parameters. We implement all models with PyTorch, and run experiments on Tesla V100 and NVIDIA GeForce GTX 1080 Ti GPUs. For fair comparisons, we adopt the same communication rounds for all methods and search for the optimal configuration of hyper-parameters using the validation sets. We run each experiment 3 times with the optimal configurations and different random seeds, and report the average results. For each method on each dataset, we use the SGD optimizer and grid search the learning rate η_g from $[0.005, 0.01, 0.03, 0.05, 0.1, 0.3, 0.5]$, set the communication round $T = 400$, the batch size as 128 and the local update step as 1 epoch. For pFedGate, the learning rate of gating layer η is searched from $[0.01, 0.05, 0.1, 0.3, 0.5, 1, 1.5]$, and we set the block size splitting factor $B = 5$ for all evaluated models. For FedAvg-FT, we use the local fine-tuning step as 1 epoch. For pFedMe, we search its penalization parameter μ from $[0.0001, 0.001, 0.01, 0.1, 1, 10]$. For Ditto, we search its regularization factor from $[0.05, 0.1, 0.5, 0.8]$. For FedRep, we search its learning rate for the personalized model from $[0.05, 0.005, 0.5, 0.01, 0.1]$. For FedEM, we set its number of global models as 3. We summarize the adopted learning rates in Table 7.

I. Additional Experiments

I.1. Novel Clients Generalization

Table 8: Averaged accuracy when testing on novel clients that have not participated in the training stage of FL. The $\widetilde{Acc}$ indicates the results of novel clients, and Δ indicates the generalization gap $\overline{Acc} - \widetilde{Acc}$.

	FEMNIST			CIFAR10		
	$\overline{Acc}$	$\widetilde{Acc}$	Δ	$\overline{Acc}$	$\widetilde{Acc}$	Δ
FedAvg	76.39	74.93	1.46	67.78	66.15	1.63
FedEM	79.21	78.26	0.95	71.92	70.75	1.17
ours, s=1	87.79	87.18	0.61	73.97	73.18	0.79
s=0.5	86.64	86.19	0.45	74.12	73.19	0.93
s=0.3	85.69	85.12	0.57	73.64	72.78	0.86

To evaluate the generalization of pFedGate for the novel clients that haven't participated in the previous FL training stage, we conduct experiments with similar simulation settings to Yuan et al. (2022); Chen et al. (2022), where the novel clients only fine-tune the global model on their local dataset. Specifically, we randomly select 20% clients as novel clients, and evaluate the selected and remaining clients separately. The results are summarized in Table 8, where $\overline{Acc}$ and $\widetilde{Acc}$ indicate the results of FL-participated clients and novel clients respectively, and $\Delta = \overline{Acc} - \widetilde{Acc}$ indicates the accuracy generalization gap. The results show that the proposed method pFedGate can generalize to novel clients well and gain smaller accuracy gaps than compared methods. For the novel clients, instead of re-training the over-parameterized global model on (possibly small amounts of) local data, pFedGate only needs to train the personalized lightweight gating layers with a small number of parameters, and thus achieves an efficient and effective local adaptation.

Table 9: Convergence FL round to achieve 90% highest accuracy.

	EMNIST	FEMNIST	CIFAR10	CIFAR100
FedAvg	72.3	195.3	253.1	226.9
FedEM	77.4	212.8	265.7	245.1
ours, s=1	54.6	152.9	190.9	183.8
s=0.5	57.9	174.1	219.3	196.5
s=0.3	75.2	214.4	271.8	260.7

I.2. Convergence Study

We demonstrate the convergence curves of FedAvg, HeteroFL with average sparsity rate as 0.55, the strongest baseline, FedEM, and the proposed pFedGate with different sparsity rates on EMNIST dataset in Figure 6. We can see that the proposed method achieves comparable convergence speeds with the baselines, which provides empirical evidence for the Theorem 2. In addition, to gain further insights into the proposed method, we plot the histogram of averaged gating weights of all clients on the CIFAR10 dataset in Figure 7. We can see that the sparse pattern becomes stable as we train the gating layer with more local datasets, showing that the proposed method can achieve effective optimization and converge to certain global points under distinct personalized local models.

Besides, we summarize the convergence FL round that each method achieves the 90% of its highest average accuracy in Table 9, where the values are averaged over three experiments with different random seeds. We observe that pFedGate gains comparable convergence speed with FedAvg and FedEM, providing empirical evidence for Theorem 2. Besides, with the results and discussion of the single-FL-round and single-client efficiency given in 7.4, we can see that pFedGate is indeed able to achieve significant accuracy improvements while at the same time improving system efficiency.

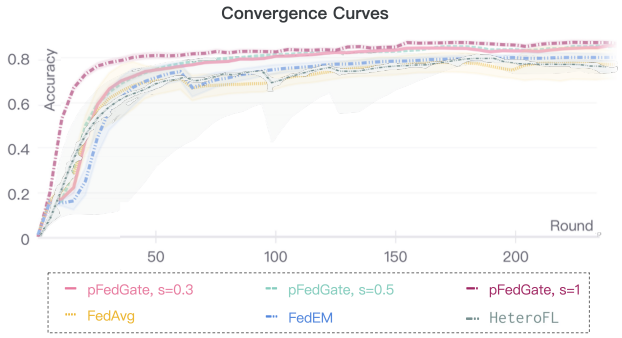

Figure 6: The convergence curves of FedAvg, HereroFL, pFedGate and the strongest baseline, FedEM on EMNIST. The pFedGate achieves comparable or even better (s=1) convergence speeds than baselines.

Table 10: Effect of s when comparing the performance of global and local (personalized) models on EMNIST dataset.

	Local Model		Global Model		Local-Global Gap	
	$\overline{Acc}$	$\widetilde{Acc}$	$\overline{Acc}$	$\widetilde{Acc}$	$\Delta\overline{Acc}$	$\Delta\widetilde{Acc}$
s=1	87.11	81.43	79.47	71.28	7.64	10.15
s=0.5	87.28	81.15	76.46	66.98	10.82	14.17
s=0.3	87.09	82.52	77.14	66.29	9.95	16.23
s=0.1	83.58	75.14	76.45	66.98	7.13	8.16

I.3. More Studies for the Effect of s

Global v.s. Local Models. In Section 7.6, we vary the sparsity factor s from 1 to 0.1 to examine the accuracy of pFedGate. Here we further compare the accuracy of both the local (personalized) models and the global (shared) model. We firstly directly examined the accuracy of the global model for the same number of FL rounds when the local model achieved the best performance, and find that the accuracy of the global model can be very bad as in the proposed pFedGate method, the global model does not receive a direct correction signal for the ground truth (but rather via a sparse adaptation). We thus train the global model with one more local epoch, then aggregate the trained one and evaluate it. The results on EMNIST dataset are listed in Table 10.

Interestingly, we found that pFedGate's local model consistently achieved better accuracy than the global model, the accuracy advantage is more significant on bottom accuracy than on average accuracy. Furthermore, as the sparsity decreases, the advantage increases (from s =1 to 0.3) and then decreases (s =0.3 to 0.1). This suggests that introducing moderate sparsity helps generalization, but at very high sparsity our method is at risk of degradation and the degradation outweighs the benefits of increasing sparsity, requiring further enhancement with small. A discussion of the relevant theoretical intuition is given in Theorem 1.

Communication Frequency and Data Distribution. In addition, to investigate the effect of s w.r.t. FL characteristics, we (1) change the local update steps (the smaller the number of steps, the more frequently the Federation updates) and (2) change the Dirichlet factor to simulate different Non-IID degrees (the smaller the α, the higher the heterogeneity of the data). The results on EMNIST with different local update steps and results on CIFAR-10 with different αs are listed in Table 11 and Table 12 respectively.

Generally speaking, we can find that pFedGate gains comparable performance when the local update step increases from 1 to 4, but a significant drop when the step becomes 8. Besides, as the Non-IID degree increases, the accuracy of pFedGate mostly decreases especially when α=0.01. These observations suggest that there is still room to improve the robustness of pFedGate, particularly in terms of how to effectively aggregate the local models for gaining high accuracy when the local and global models can easily differ greatly. For example, in the future, we could consider combining asynchronous federal learning (Nguyen et al., 2022) and more advanced aggregation mechanisms in the model parameter space (Ainsworth et al., 2022) to allow for larger variations of the participants' models.

I.4. Model Compression Manner Study

We gain sparse local models in the proposed framework via learning personalized gating layers and predicting sparse gated weights for model sub-blocks. As introduced in Section 5.2, we propose a simple and general split manner that chunks the model parameters into B equal-sized sub-blocks, where the split factor B provides flexible splitting granularity. Here we study the sparsification technique proposed in our method by varying the split factor B and comparing it with a SOTA PFL method FedMask (Li et al., 2021a), that adopts binary masks to compress the parameters of models' last layers. The results are shown in Table 13, where we examine the FL accuracy by setting $B = [2, 5, 10]$ and $s = [0.3, 0.5]$.

From Table 13 we have the following observations: Our method achieves better performance than FedMask in various sparsity settings. Besides, when increasing the sparsity ($s = 0.5$ to $s = 0.3$), our method with $B = 5$ still achieves good or even better performance (the accuracy change is +0.248 on average) while FedMask gains large performance drops (the accuracy change is -1.543 on average). These results verify the effectiveness of the proposed adaptive sparse weights prediction again. Note that FedMask freezes the local models during the whole FL process and only learns the binary masks for the last several layers of the models. Instead, we train both the local models and the personalized gating layers, which generate continuous gated weights to mask some sub-blocks and scale the weights of other

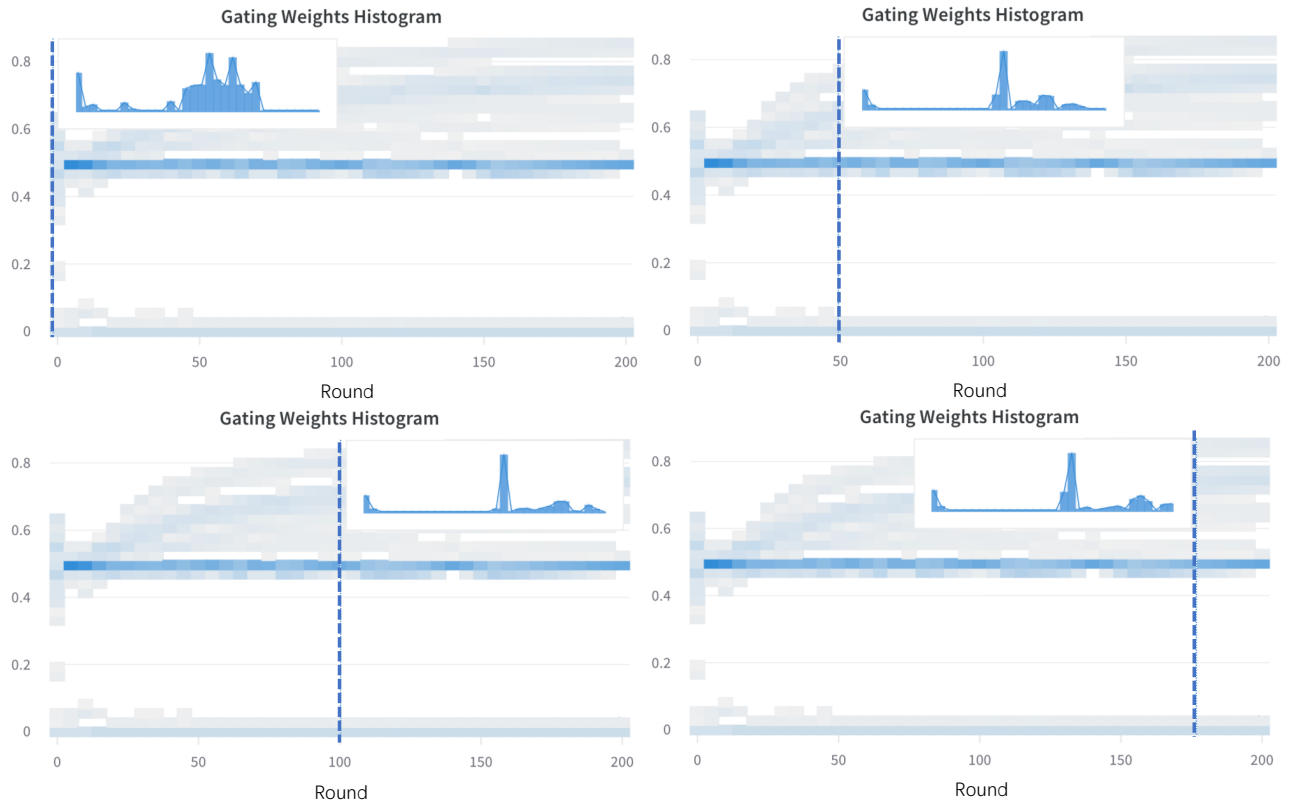

Figure 7: The convergence curves of pFedGate in terms of the histogram of averaged gating weights over all clients on the CIFAR10 dataset.

sub-blocks, leading to high capacities for the learnable personalized models. With a smaller sparsity hyper-parameter value, the size of the model's hypothesis space searched by FedMask (depicted by only the masks for the last several layers) is much smaller than the one of pFedGate (depicted by both the whole model and the gating weights for all the sub-blocks).

As for the impact of B, on the one hand, we can see that too small $B = 2$ gains relatively worse results than $B = 5$ as there may be insufficient sub-blocks to model the clients' diversity. On the other hand, the larger $B = 10$ achieves similar results to $B = 5$ while larger performance drop when increasing sparsity. We empirically found that $B = 5$ has good robustness and adopt it as the default value in our experiments.

Table 11: Effect of s when changing the FL communication frequency on FEMNIST dataset. The smaller the number of steps, the more frequently the Federation updates.

	Step=1		Step=2		Step=4		Step=8	
	$\overline{Acc}$	$\breve{Acc}$	$\overline{Acc}$	$\breve{Acc}$	$\overline{Acc}$	$\breve{Acc}$	$\overline{Acc}$	$\breve{Acc}$
pFedGate, s=1	87.11	81.43	86.87	81.21	87.01	81.07	84.28	79.53
s=0.5	87.28	81.15	87.25	80.99	87.09	80.65	84.62	78.45
s=0.3	87.09	82.52	87.48	82.63	86.39	82.08	85.31	80.86
s=0.1	83.58	75.14	83.27	74.81	82.99	74.58	80.75	73.05

Table 12: Effect of s when changing the Dirichelet factor α of Non-IID CIFAR-10 dataset. The smaller the α is, the higher the heterogeneity of the data.

	α=0.4		α=0.2		α=0.1		α=0.01	
	$\overline{Acc}$	$\breve{Acc}$	$\overline{Acc}$	$\breve{Acc}$	$\overline{Acc}$	$\breve{Acc}$	$\overline{Acc}$	$\breve{Acc}$
pFedGate, s=1	75.18	66.67	74.29	65.71	73.55	65.23	72.63	64.64
s=0.5	74.07	64.21	73.05	63.44	72.55	63.03	71.76	62.02
s=0.3	73.65	64.39	72.58	64.05	71.95	62.83	71.58	62.68
s=0.1	70.26	60.45	69.74	59.56	68.96	59.38	67.45	58.04

Table 13: Averaged accuracy comparison for PFL methods with and without model compression. The numbers in the parentheses indicate the accuracy difference between sparsity $s = 0.5$ and $s = 0.3$ for FedMask and pFedGate.

			FEMNIST		CIFAR10	
			$\overline{Acc}$	$\breve{Acc}$	$\overline{Acc}$	$\breve{Acc}$
FedAvg			76.51	60.82	68.33	60.22
FedEM			80.12	64.81	72.43	62.88
FedMask	s=0.5		78.69	62.18	70.54	61.37
	s=0.3		77.41 (-1.28)	60.69 (-1.49)	68.46 (-2.08)	59.05 (-1.32)
pFedGate	B=5	s=0.5	86.31	75.68	74.07	64.21
		s=0.3	86.75 ($+0.44$)	76.47 ($+0.79$)	73.65 (-0.42)	64.39 ($+0.18$)
	B=2	s=0.5	81.10	71.28	70.81	61.03
		s=0.3	80.03 (-1.07)	70.72 (-0.56)	70.65 (-0.16)	60.66 (-0.37)
	B=10	s=0.5	85.38	74.91	72.94	63.59
		s=0.3	84.67 (-0.71)	75.10 ($+0.19$)	72.48 (-0.46)	63.58 (-0.11)