pFedSAM: Secure Federated Learning Against Backdoor Attacks via Personalized Sharpness-Aware Minimization

Zhenxiao Zhang[1], Yuanxiong Guo[2], Yanmin Gong[1]
[1]Department of Electrical and Computer Engineering, University of Texas at San Antonio
[2]Department of Information Systems and Cyber Security, University of Texas at San Antonio
Email: {zhenxiao.zhang, yuanxiong.guo, yanmin.gong}@utsa.edu

***Abstract*—Federated learning is a distributed learning paradigm that allows clients to perform collaborative model training without sharing their local data. Despite its benefit, federated learning is vulnerable to backdoor attacks where malicious clients inject backdoors into the global model aggregation process so that the resulting model will misclassify the samples with backdoor triggers while performing normally on the benign samples. Existing defenses against backdoor attacks either are effective only under very specific attack models or severely deteriorate the model performance on benign samples. To address these deficiencies, this paper proposes pFedSAM, a new federated learning method based on partial model personalization and sharpness-aware training. Theoretically, we analyze the convergence properties of pFedSAM for the general non-convex and heterogeneous data setting. Empirically, we conduct extensive experiments on a suite of federated datasets and show the superiority of pFedSAM over state-of-the-art robust baselines in terms of both robustness and accuracy.**

***Index Terms*—federated learning, backdoor attack, personalization, sharpness-aware minimization**

I. Introduction

Federated learning (FL) has emerged as a transformative paradigm in machine learning, enabling collaborative model training among distributed clients while keeping their data locally. Although FL has achieved success in many applications, such as medical image analysis and the Internet of things, FL systems confront significant security threats due to their distributed nature, particularly from backdoor attacks [1]–[4]. Specifically, by stealthily injecting backdoor triggers into the trained model, attackers aim to mislead any input with the backdoor trigger to a target label while ensuring that the backdoored model's performance on benign samples remains unaffected. Such stealthy manipulation makes backdoor attacks one of the most serious threats to the real-world deployment of FL systems.

Existing defenses against backdoor attacks in FL can be roughly divided into two categories [5]: anomaly update detection and robust federated training. The first category consists of anomaly detection approaches that can identify whether the submitted updates are malicious and then remove the malicious ones, such as Krum [6], Trimmed Mean [7], and Bulyan [8]. However, these methods are effective only under very specific attack models (i.e., attack strategies of the adversary and data distribution of the benign clients). The second category comprises robust federated training methods that can directly mitigate backdoor attacks during the training process, such as norm clipping [4] and adding noise [9]. These solutions require modification of the individual weights of benign model updates and therefore result in severe degradation of model performance on benign samples. Moreover, most of the aforementioned works only work in the single-shot attack setting where a small number of malicious clients participate in a few rounds but fail under the stronger continuous attack setting where malicious clients continuously participate in the entire FL training period [10].

A few recent works [11]–[13] have demonstrated that personalized federated learning (pFL) methods that were originally designed to improve accuracy under heterogeneous data distribution could also provide some robustness benefits. Specifically, Li et al. [12] and Lin et al. [13] utilize model personalization to defend against *untargeted* poisoning attacks that aim to corrupt FL models' prediction performance or make FL training diverge, but do not address the more challenging problem of backdoor attacks. Lin et al. [11] further demonstrate that pFL with partial model-sharing can notably enhance robustness against backdoor attacks in comparison to pFL with full model-sharing under the continuous attack setting, but it solely focuses on the black-box setting where malicious adversaries can only manipulate training data and have no control of the training process. Considering the white-box setting where malicious clients can control the local training process, [14] demonstrates that pFL methods with partial model-sharing remain vulnerable to backdoor attacks. Therefore, a straightforward implementation of pFL is susceptible to new attacks tailored for pFL and does not ensure robustness against real-world backdoor attacks.

In this paper, we propose pFedSAM, a novel personalized FL method that can inherently defend against both black-box and white-box state-of-the-art backdoor attacks while maintaining the benign performance of the models. This is achieved by two key modules: partial model personalization and sharpness-aware training. The partial model personalization lets each client own its locally preserved linear classifier to block the propagation of backdoor features from malicious clients to benign clients. The sharpness-aware training generates local flat model updates with better stability and

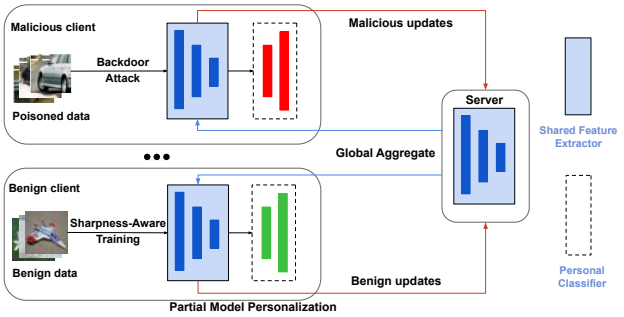

Fig. 1: An overview of pFedSAM under backdoor attacks. Partial model personalization allows each client to retain a personal classifier locally and only share the feature extractor with the server for aggregation. The malicious client performs the backdoor attack and sends the malicious updates, while the benign client engages in sharpness-aware training and sends the benign updates to the server. The server aggregates the shared feature extractor and sends it back to all clients.

perturbation resilience, resulting in a globally flat model that is robust to the injection of backdoor features from malicious clients. The overview of pFedSAM is shown in Fig. 1. We summarize our main contributions as follows.

- We propose pFedSAM, a novel pFL method that offers better robustness against both black-box and white-box backdoor attacks while retaining similar or superior accuracy on the benign model performance relative to other common robust FL methods.
- We provide convergence guarantees for our proposed pFedSAM method under the general nonconvex and non-IID data distribution setting.
- We conduct an extensive evaluation of the proposed method on several FL benchmark datasets by comparing it with state-of-the-art baselines under the stronger and stealthier continuous black-box and white-box backdoor attacks. The empirical results show that the proposed method can largely outperform the baselines in terms of both attack success rate and main task accuracy.

II. Background and Related Works

A. Personalized Federated Learning

We consider a typical FL system involving N clients and one server. Each client $i \in [N]$ holds a local training dataset $\mathcal{D}_i = \{\xi_{i,j}\}_{j=1}^{D_i}$ where $\xi_{i,j}$ is a training example and D_i is the size of the local training dataset. The total number of training examples across N devices is $D = \sum_{i=1}^{N} D_i$. Let $w \in \mathbb{R}^d$ denote the parameters of a model and $f_i(w, \xi_{i,j})$ be the loss of the model on the sample $\xi_{i,j}$. Then the loss function of client i is $F_i(w) = (1/D_i) \sum_{j \in \mathcal{D}_j} f_i(w_i, \xi_{i,j})$. The objective of standard FL is to find model parameters that minimize the weighted average loss over all clients:

$$\min_{w} \sum_{i=1}^{N} \alpha_i F_i(w), \tag{1}$$

where $\alpha_i > 0$ is the weight assigned to client i, and $\sum_{i=1}^{N} \alpha_i = 1$. However, standard FL can be ineffective and undesirable under data heterogeneity [12]. Instead, pFL aims to train local personalized models instead of a single global model across all clients, which is more adaptive to each client's local dataset and has been shown to improve model accuracy under practical non-IID scenarios. Based on the form of model sharing with the server, existing pFL methods can be divided into two categories: full model-sharing and partial model-sharing. 1) *Full model-sharing* [12]: The objective can be summarized as

$$\min_{w_0, \{w_i\}_{i=1}^{N}} \sum_{i=1}^{N} \alpha_i \left(F_i(w_i) + \lambda_i R(w_0, w_i) \right), \tag{2}$$

where w_0 is a reference model shared among all clients, w_i means the local personalized model owned by client i, and λ_i is the weight of the regularization term $R(w_0, w_i)$ for client i. 2) *Partial model-sharing* [15], [16]: The objective can be summarized as

$$\min_{\phi, \{h_i\}_{i=1}^{N}} \sum_{i=1}^{N} \alpha_i F_i(\phi, h_i), \tag{3}$$

where the full model parameters w_i of each client i are divided into two parts: *shared* parameters $\phi \in \mathbb{R}^{d_0}$ and *personal* parameters $h_i \in \mathbb{R}^{d_i}$, i.e. $w_i = (\phi, h_i)$.

As shown in [16], partial model-sharing personalization can obtain most of the benefit of full model-sharing personalization with only a small fraction of personalized parameters. Our work builds on FedRep [15], a pFL algorithm with partial model-sharing that focuses on learning shared representations and personal classifier heads between clients but does not consider robustness. In contrast, our work provides a novel robust FL framework. Moreover, a major different ingredient of our algorithm is the sharpness-aware training for shared representation learning, which finds backdoor-resilient global shared parameters in each FL round.

B. Backdoor Attacks in Federated Learning

In FL backdoor attacks, the adversary controls a group of malicious clients to manipulate their local models, which are then aggregated into the global model and affect its properties. In particular, the adversary wants the backdoored global model to mislead the prediction on inputs with the backdoor trigger to a target label while behaving normally on all benign samples. There are generally two categories for FL backdoor attacks: 1) *black-box setting*, where malicious clients tamper with a fraction of their training data, also known as data poisoning, to inject a backdoor into their local models during the training [3]; and 2) *white-box setting*, where the adversary poisons the training data of the malicious clients and manipulates their training processes by modifying the resulting uploaded models, also known as model poisoning, to maximize attack impact while avoiding being detected. Examples of white-box backdoor attacks include *constrain-and-scale attack* [1], *projected gradient descent attack with model replacement* [2], DBA [3], and BapFL [14].

C. SAM training

SAM [17] and its variant [18] are initially developed to enhance model generalization by simultaneously minimizing the loss and its sharpness. This technique, which seeks flat minima, has recently been extended to provide robustness against backdoor attacks in works [19]. However, they focus on the fine-tuning stage in centralized learning. The aspects of defense during the training phase and in FL remain unexplored. We utilize the SAM training technique as part of the genre of backdoor defense. To the best of our knowledge, this is the first initiative to implement SAM training for backdoor defense during the training phase in an FL setting.

III. pFedSAM: Federated Learning with Personalized Sharpness-Aware Minimization

A. pFedSAM Algorithm

We present the pFedSAM algorithm to solve Problem (3) and describe its detailed procedures in Algorithm 1. The overall training process of pFedSAM consists of updating two parts of a client's model in an alternating manner: local *personal* parameters h_i and global *shared* parameters ϕ.

At the beginning of each t-th round, the server randomly samples a subset of the clients $\mathcal{S}^t$ to join the learning process (line 2) and broadcasts the current global version of the shared parameters ϕ^t to clients in $\mathcal{S}^t$ (line 3). Then, each selected client $i \in \mathcal{S}^t$ performs local training in two stages. First, it fixes the local version of the shared parameters ϕ_i to be the received global one ϕ^t and then performs τ_h iterations of SGD to update the personal parameters h_i (lines 5–9). Second, it fixes the personal parameters h_i to be the newly updated one h_i^{t,τ_h} obtained from the first stage (line 10) and then updates the shared parameters ϕ_i. Here, instead of seeking out shared parameters that simply have low training loss by minimizing $F_i(\phi, h_i)$, we propose to find shared parameters whose entire neighborhoods have uniformly low training loss. This can be formulated as solving the following Sharpness-Aware Minimization (SAM) problem that jointly minimizes the loss function and smooths the loss landscape:

$$\min_{\phi} \max_{\|\epsilon\|_2 \leq \rho} F_i(\phi + \epsilon, h_i^{t+1}), \tag{4}$$

where ρ is a predefined constant controlling the radius of the perturbation. Intuitively, through optimizing the objective (4), the resulting local version of the shared parameters ϕ_i has a smoother local loss landscape and exhibits inherent robustness to perturbations. Then by aggregating all the local models with a smoother local loss landscape at the server, the flatness of the aggregated global model is boosted as well, making it more resilient to the injection of backdoor features from malicious clients.

To solve the min-max problem (4), we adopt the efficient and effective approximation technique proposed in [17].

Algorithm 1: pFedSAM

Input: Initial states $\phi^0, \{h_i^0\}_{i=1}^N$, client sampling ratio r, number of local iterations τ_h, τ_ϕ, number of communication rounds T, learning rates η_h, η_ϕ, and neighborhood size ρ

Output: Personalized models $(\phi^T, h_i^T), \forall i \in [N]$.

1: **for** $t = 0, 1, \ldots, T-1$ **do**
2: Server randomly samples a set of rN clients $\mathcal{S}^t$.
3: Server broadcasts the current global version of the shared parameters ϕ^t to all clients in $\mathcal{S}^t$.
4: **for** each client $i \in \mathcal{S}^t$ in parallel **do**
5: Initialize $h_i^{t,0} = h_i^t$
6: **for** $s = 0, \ldots, \tau_h - 1$ **do**
7: Compute stochastic gradient $\widetilde{\nabla}_h F_i(\phi^t, h_i^{t,s})$
8: $h_i^{t,s+1} = h_i^{t,s} - \eta_h \widetilde{\nabla}_h F_i(\phi^t, h_i^{t,s})$
9: **end for**
10: Update $h_i^{t+1} = h_i^{t,\tau_h}$ and initialize $\phi_i^{t,0} = \phi^t$
11: **for** $s = 0, \ldots, \tau_\phi - 1$ **do**
12: Update shared parameters using SAM according to (6) and (7)
13: **end for**
14: Update $\phi_i^{t+1} = \phi_i^{t,\tau_\phi}$
15: Client sends ϕ_i^{t+1} back to server
16: **end for**
17: **for** each client $i \notin \mathcal{S}^t$ **do**
18: $h_i^{t+1} = h_i^t$
19: **end for**
20: Server updates $\phi^{t+1} = \frac{1}{rN} \sum_{i \in \mathcal{S}^t} \phi_i^{t+1}$
21: **end for**

Specifically, via the use of the first-order Taylor expansion of F_i, the solution of the inner maximization problem is

$$\begin{aligned} \epsilon^*(\phi) &\approx \arg\max_{\|\epsilon\|_2 \leq \rho} \left\{ F_i(\phi, h_i^{t+1}) + \epsilon^T \nabla_\phi F_i(\phi, h_i^{t+1}) \right\} \\ &= \rho \frac{\nabla_\phi F_i(\phi, h_i^{t+1})}{\|\nabla_\phi F_i(\phi, h_i^{t+1})\|_2} \end{aligned} \tag{5}$$

Substituting (5) back into (4) and taking the differentiation w.r.t. ϕ, we can obtain the approximate SAM gradient as $\nabla_\phi F_i(\phi, h_i^{t+1})|_{\phi + \epsilon^*(\phi)}$. Therefore, at the s-th local iteration of round t, SAM first computes partial stochastic gradient $\widetilde{\nabla}_\phi F_i(\phi_i^{t,s}, h_i^{t+1})$ and calculates the perturbation $\epsilon(\phi_i^{t,s})$ as follows:

$$\epsilon(\phi_i^{t,s}) = \rho \frac{\widetilde{\nabla}_\phi F_i(\phi_i^{t,s}, h_i^{t+1})}{\|\widetilde{\nabla}_\phi F_i(\phi_i^{t,s}, h_i^{t+1})\|_2}. \tag{6}$$

Then the perturbation is used to update the shared parameters as follows:

$$\phi_i^{t,s+1} = \phi_i^{t,s} - \eta_\phi \widetilde{\nabla}_\phi F_i(\phi_i^{t,s} + \epsilon(\phi_i^{t,s}), h_i^{t+1}), \tag{7}$$

where η_ϕ is the learning rate. The same procedure repeats for τ_ϕ local iterations (lines 10–13).

After local training, each selected client i only sends the updated local version of the shared parameters ϕ_i^{t+1} to the server, which aggregates them from all selected clients to compute the global version of the shared parameters ϕ^{t+1} for the next round (line 20). The updated personal parameters h_i^{t+1}

are kept locally at the client to serve as the initialization when the client is selected for another round.

B. Convergence Properties of pFedSAM

In this section, we give the convergence results of pFedSAM. To simplify presentation, we denote $H = (h_1, \ldots, h_N) \in \mathbb{R}^{d_1+\cdots+d_N}$. We consider a general setting with $\alpha_i = 1/N$ without loss of generality. Then our objective becomes $\min_{\phi,H} F(\phi, H) = \frac{1}{N}\sum_{i=1}^{N} F_i(\phi, h_i)$. Before stating our theoretical results, we make the following assumptions for the convergence analysis.

Assumption 1 (Smoothness)**.** *For each $i \in [N]$, the function F_i is continuously differentiable. There exist constants L_ϕ, L_h, $L_{\phi h}$, $L_{h\phi}$ such that for each $i \in [N]$: 1) $\nabla_\phi F_i(\phi, h_i)$ is L_ϕ-Lipschitz with respect to ϕ and $L_{\phi h}$-Lipschitz with respect to h_i, and $\nabla_h F_i(\phi, h_i)$ is L_h-Lipschitz with respect to h_i and $L_{h\phi}$-Lipschitz with respect to ϕ. The relative cross-sensitivity of $\nabla_\phi F_i$ with respect to h_i and $\nabla_h F_i$ with respect to ϕ is defined by the following scalar: $\chi := \max\{L_{\phi h}, L_{h\phi}\}/\sqrt{L_\phi L_h}$.*

Assumption 2 (Bounded Variance)**.** *The stochastic gradients in Algorithm 1 are unbiased and have bouned variance. That is, for all ϕ and h_i, $\mathbb{E}[\widetilde{\nabla}_\phi F_i(\phi, h_i)] = \nabla_\phi F_i(\phi, h_i)$, $\mathbb{E}[\widetilde{\nabla}_h F_i(\phi, h_i)] = \nabla_h F_i(\phi, h_i)$. Furthermore, there exist constants σ_ϕ and σ_h such that $\mathbb{E}\|widetilde\nabla_\phi F_i(\phi, h_i) - \nabla_\phi F_i(\phi, h_i)\|^2 \le \sigma_\phi^2$, $\mathbb{E}\|\widetilde{\nabla}_h F_i(\phi, h_i) - \nabla_h F_i(\phi, h_i)\|^2 \le \sigma_h^2$.*

Assumptions 1 and 2 are standard in the analysis of SGD [20]–[24]. Here, we can view $\nabla_\phi F_i(\phi, h_i)$, when i is randomly sampled from $[N]$, as a stochastic partial gradient of $F(\phi, H)$. The following assumption imposes a constant variance bound.

Assumption 3 (Partial Gradient Diversity)**.** *There exist a constant δ such that for all ϕ and H, $\frac{1}{N}\sum_{i=1}^{N} \|\nabla_\phi F_i(\phi, h_i) - \nabla_\phi F_i(\phi, H)\|^2 \le \delta^2$.*

We denote $\Delta F_0 = F(\phi^0, H^0) - F^*$ with F^* being the minimal value of $F(\cdot)$. Further, we use the shorthands $H^t = (h_1^t, \cdots, h_N^t)$, $\Delta_\phi^t = \|\nabla_\phi F(\phi^t, H^t)\|^2$, and $\Delta_h^t = 1/n\sum_{i=1}^{N} \|\nabla_h F(\phi^t, h_i^t)\|^2$.

Next, we propose our main theoretical results of the proposed pFedSAM algorithm in the following theorem.

Theorem 1 (Convergence of Algorithm 1)**.** *Under Assumptions 1-3, if the learning rates satisfy $\eta_\phi = \alpha/(L_\phi \tau_\phi)$ and $\eta_h = \alpha/(L_h \tau_h)$, where α depends on the parameters $L_\phi, L_h, \chi^2, \sigma_\phi^2, \sigma_h^2, r$, and the number of total rounds T, we have*

$$\frac{1}{T}\sum_{t=0}^{T-1}\left(\frac{1}{L_\phi}\mathbb{E}[\Delta_\phi^t] + \frac{r}{L_h}\mathbb{E}[\Delta_h^t]\right) \le \frac{(\Delta F_0 \Omega_1^2)^{1/2}}{\sqrt{T}} \quad (8)$$

$$+\frac{(\Delta F_0^2 \Omega_2^2)^{1/3}}{T^{2/3}} + \mathcal{O}(\tfrac{1}{T}), \quad (9)$$

where the effective variance terms are defined as follows:

$$\Omega_1^2 = \frac{\sigma_h^2}{L_h}(r + \chi^2(1-r)) + \frac{\sigma_\phi^2}{L_\phi} + \frac{\delta_\phi^2}{L_\phi}(1-r),$$

$$\Omega_2^2 = \frac{\chi^2 \sigma_h^2}{L_h} + \frac{\rho^2}{\tau_\phi} + \frac{\sigma_\phi^2 + \delta^2}{L_\phi}.$$

The details of the proof can be found in [25]. The left-hand side of (8) represents the time-averaged value of a weighted combination of $\mathbb{E}[\Delta_\phi^t]$ and $\mathbb{E}[\Delta_h^t]$. The convergence rate, dictating how rapidly this value diminishes to zero, is tied to the effective noise variances Ω_1^2 and Ω_2^2. These variances result from the SAM gradient perturbation parameter ρ^2 and three stochastic variances σ^2, σ_ϕ^2, and σ_h^2.

IV. Experiments

In this section, we empirically evaluate pFedSAM's robustness against the state-of-the-art black-box and white-box backdoor attacks. For black-box, we use the BadNet attack [26], a common centralized training attack. For white-box, we implement the DBA [3] and BapFL [14] attacks. DBA significantly enhances the persistence and stealthiness against FL on diverse data by breaking down the BadNet trigger pattern into distinct local patterns and injecting them in a distributed way. BapFL [14] is the most recent backdoor attack specifically tailored for pFL with partial model-sharing. We compare our proposed pFedSAM method with seven widely used defense strategies in FL: Krum [6], Multi-Krum [6], Adding Noise (AD) [4], Norm Clipping (NC) [2], Ditto [12], FedRep [15], and Simple-Tuning (ST) [11]. Krum and Multi-Krum aim to filter out malicious clients by selecting one or multiple model updates based on similarity for aggregation. NC and AD mitigate backdoor attacks by limiting the norm of model updates or adding Gaussian noise before aggregation. We set the threshold $c \in \{0.5, 1.0\}$ in NC and noise scales $\sigma \in \{10^{-5}, 5 \times 10^{-4}\}$ in AD. Ditto is a full model-sharing pFL method that has been demonstrated to provide robustness benefits. FedRep is a partial model-sharing pFL method, which has been validated in [11] to offer superior robustness against black-box attacks compared to other pFL methods. ST is a newly proposed defense method in [11] that re-initializes and retrains the local linear classifier on a benign local dataset while freezing the remaining parameters of its model.

A. Experimental Settings

We run each experiment 5 times with distinct random seeds and provide the average accuracy in the same last round for fair comparison.

Datasets and Models. Following the prior works in robust FL [2], [9], [14], we use two common datasets: MNIST and CIFAR-10. The heterogeneity of these datasets across clients is controlled by following the Dirichlet distribution [27] with concentration parameter β (default $\beta = 0.5$), where a smaller β indicates greater heterogeneity. For CIFAR-10, we use a CNN with two convolutional layers and three fully connected layers, and for MNIST, an MLP with two hidden layers.

Attack Setup. We perform FL training over 100 and 300 rounds for MNIST and CIFAR-10, respectively, with a default of 100 clients. We consider the stronger and stealthier backdoor attack setting in which malicious clients continuously participate in each round. Following prior studies [3], [11], [14], we randomly sample 10 clients, including 4 malicious ones for DBA and BapFL, or 1 for BadNet; the remaining

TABLE I: Black-box BadNet attack evaluation

Defenses	MNIST		CIFAR-10	
	ACC	ASR	ACC	ASR
FedAvg (no defense)	**96.09**	97.03	70.94	31.88
FedRep	90.44	35.86	72.85	7.15
Ditto	87.66	58.41	72.28	30.61
NC ($c = 0.5$)	95.82	98.62	56.06	19.64
NC ($c = 1.0$)	95.96	98.57	69.54	22.96
AD ($\sigma = 10^{-5}$)	95.26	96.90	70.97	20.59
AD ($\sigma = 5 \times 10^{-4}$)	95.12	96.43	42.15	10.86
Krum	93.58	31.96	62.88	15.10
Multi-Krum	95.86	19.58	69.68	15.55
ST	66.58	33.92	**75.21**	15.57
pFedSAM	91.42	**14.51**	75.06	**5.76**

TABLE II: White-box DBA attack evaluation

Defenses	MNIST		CIFAR-10	
	ACC	ASR	ACC	ASR
FedAvg (no defense)	**94.54**	100.00	66.32	74.82
FedRep	87.47	21.69	67.73	6.19
Ditto	56.57	85.23	60.53	15.01
NC ($c = 0.5$)	92.68	97.25	65.68	37.04
NC ($c = 1.0$)	92.89	99.96	66.33	44.99
AD ($\sigma = 10^{-5}$)	93.97	99.99	57.24	34.91
AD ($\sigma = 5 \times 10^{-4}$)	93.07	99.88	36.88	15.24
Krum	88.62	92.50	62.01	8.33
Multi-Krum	94.01	97.36	**69.68**	20.92
ST	77.95	51.76	67.88	8.77
pFedSAM	89.10	**12.13**	69.61	**5.06**

clients are benign. For pFedSAM, we set 2 local epochs for personal parameters and 2 for shared parameters in each FL round, with $\rho = 0.05$. All methods follow the same local epochs as pFedSAM. Malicious clients poison 20 out of 64 samples per batch for CIFAR-10 and MNIST.

Evaluation Metrics. We use two metrics, attack success rate (ASR) and main task accuracy (ACC), to assess pFedSAM's effectiveness. ASR is the proportion of successfully attacked poisoned samples among all poisoned samples, while ACC measures the model's accuracy on benign samples. An effective backdoor attack should achieve high ASR and ACC, indicating it can manipulate outputs without degrading primary task performance. To ensure unbiased results, ASR is computed only on samples where the true label differs from the target label [3].

B. Experimental Results

BadNet Attack. Table I shows that the BadNet attack reaches over 97% and 31% ASRs on MNIST and CIFAR-10. Partial model-sharing pFL methods like FedRep and pFedSAM effectively lower the ASR while maintaining high ACC in the black-box setting. Fedrep effectively reduces the ASR below 36% on MNIST and 8% on CIFAR-10. Conversely, the full model-sharing pFL method Ditto shows limited robustness, reducing ASR by only 1% on CIFAR-10 and remaining over 58% on MNIST. This matches the prior results observed in [11].

NC with $c = 1.0$ offers minor robustness gains over FedAvg, reducing ASR by only 1% on MNIST. Lowering c to 0.5 can mitigate attacker influence, but significantly drops ACC, especially on CIFAR-10 (below 57%). AD with high noise scale ($\sigma = 5 \times 10^{-4}$) similarly impacts accuracy, bringing ACC below 43%. Krum and Multi-Krum enhance robustness by filtering malicious clients, yet struggle to fully identify attackers, keeping ASR at least 19% on MNIST and 15% on CIFAR-10, with some benign clients inadvertently filtered, impacting ACC. Though NC, AD, Krum, and Multi-Krum show robustness gains, they face significant robustness-accuracy trade-offs. ST achieves ASRs similar to Krum but lacks stable ACC, failing to fully counteract backdoor risks.

Among all methods, pFedSAM delivers the best robustness, achieving the lowest ASR while maintaining high ACC. Specifically, pFedSAM reduces ASR to 5.76% on CIFAR-10 and 14.51% on MNIST, achieving ACC comparable to the highest across methods. Compared to FedRep, pFedSAM improves both ACC and ASR, highlighting the benefits of sharpness-aware training for enhanced robustness.

DBA Attack. Table II highlights pFedSAM's effectiveness against DBA. The ASR is 3%-42% higher than the BadNet attack on FedAvg shows that DBA is a more aggressive attack than BadNet. Partial model-sharing methods like FedRep and pFedSAM exhibit notable robustness, whereas full model-sharing (e.g., Ditto) lacks effectiveness, with Ditto reducing ASR to 15.01% on CIFAR-10 but exceeding 85% on MNIST and having low ACC across both datasets.

NC fails to mitigate DBA effectively on either dataset. Even at a low threshold ($c = 0.5$), ASR remains above 97% on MNIST and 37% on CIFAR-10. AD also faces a robustness-ACC trade-off; although a higher noise level ($\sigma = 5 \times 10^{-4}$) reduces ASR to 15.24% on CIFAR-10, it impacts ACC, dropping it below 37%. Krum and Multi-Krum partially mitigate backdoor effects, especially on CIFAR-10, with Krum reducing ASR to as low as 8.33%, though they are ineffective against DBA on MNIST, which is more challenging. ST shows comparable ASR to Krum on CIFAR-10 but lacks stable ACC on MNIST, failing to fully counter backdoor risks.

We observe that partial model-sharing methods like FedRep and pFedSAM, can largely defend against DBA on both datasets, achieving significantly lower ASR. pFedSAM provides the best robustness, maintaining high ACC and achieving a 1%-2% increase in ACC and a 1%-9% ASR reduction over FedRep, highlighting the advantages of sharpness-aware training in improving robustness against backdoor attacks and enhancing performance on benign samples.

BapFL Attack. As the BapFL attack is custom-designed for partial model-sharing pFL, we evaluate this attack only on two methods for comparison: FedRep and pFedSAM. Moreover, we show the flexibility of pFedSAM in incorporating other defense strategies to achieve even better robustness. The results are shown in Table III. Due to its ASR being 15%-42% higher than DBA attack on FedRep, and DBA being stronger than BadNET, it clearly shows that BapFL is more aggressive than DBA and BadNet under the pFL setting.

From Table III, we can observe that the straightforward implementation of pFL with partial model-sharing, FedRep, remains susceptible to BapFL attack with relatively high

TABLE III: White-box BapFL attack evaluation

Defenses	MNIST		CIFAR-10	
	ACC	ASR	ACC	ASR
FedRep	76.77	37.50	77.50	53.13
pFedSAM	**80.36**	29.17	78.87	40.91
pFedSAM + NC ($c = 0.5$)	78.45	16.67	**81.79**	21.53
pFedSAM + NC ($c = 1.0$)	75.54	18.93	81.70	29.12
pFedSAM + AD ($\sigma = 10^{-5}$)	79.05	19.50	80.91	26.85
pFedSAM + AD ($\sigma = 5 \times 10^{-4}$)	80.17	29.36	33.83	6.70
pFedSAM + Krum	57.15	5.81	76.66	3.55
pFedSAM + Multi-Krum	75.46	**3.66**	80.97	**2.82**
pFedSAM + ST	44.76	7.04	70.38	19.67

ASRs, such as 53% on CIFAR-10 and 37.50%. Compared with FedRep, pFedSAM can enhance robustness and accuracy simultaneously under BapFL due to the use of sharpness-aware training in updating the shared parameters, but it still cannot provide an effective defense by itself. Therefore, we combine pFedSAM with the existing defense strategies (NC, AD, Krum, Multi-Krum, and ST) by applying them to the global aggregation step of the shared parameters in pFedSAM to see the effectiveness. We also observe that pFedSAM in conjunction with AD, NC, or ST does not effectively mitigate the BapFL attack. However, BapFL can be effectively mitigated when integrating the Krum or Multi-Krum into our proposed pFedSAM. Specifically, by integrating pFedSAM with Multi-Krum, it can reduce the ASR to below 4% on both datasets and keep the ACC degradation within 1%.

V. Conclusion

In this paper, we propose pFedSAM, a pFL method resilient to both black-box and white-box backdoor attacks. We also prove its convergence in non-convex, non-IID data settings. Future work will explore other SAM optimizers and broader experiments.

VI. Acknowledgments

The work of Z. Zhang, Y. Guo, and Y. Gong was partially supported by NSF under grants CNS-2047761, CNS-2106761, and CNS-2318683.

References

[1] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *International conference on artificial intelligence and statistics*, vol. 108, 2020, pp. 2938–2948.

[2] H. Wang, K. Sreenivasan, S. Rajput, H. Vishwakarma, S. Agarwal, J.-y. Sohn, K. Lee, and D. Papailiopoulos, "Attack of the tails: Yes, you really can backdoor federated learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 16 070–16 084, 2020.

[3] C. Xie, K. Huang, P.-Y. Chen, and B. Li, "DBA: Distributed backdoor attacks against federated learning," in *International Conference on Learning Representations*, 2020.

[4] Z. Sun, P. Kairouz, A. T. Suresh, and H. B. McMahan, "Can you really backdoor federated learning?" 2019. [Online]. Available: https://arxiv.org/abs/1911.07963

[5] T. D. Nguyen, P. Rieger, R. De Viti, H. Chen, B. B. Brandenburg, H. Yalame, H. Möllering, H. Fereidooni, S. Marchal, M. Miettinen *et al.*, "FLAME: Taming backdoors in federated learning," in *31st USENIX Security Symposium*, 2022, pp. 1415–1432.

[6] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," *Advances in neural information processing systems*, vol. 30, 2017.

[7] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *International Conference on Machine Learning*, vol. 80, 2018, pp. 5650–5659.

[8] E. M. El Mhamdi, R. Guerraoui, and S. Rouault, "The hidden vulnerability of distributed learning in byzantium," in *International Conference on Machine Learning*, vol. 80, 2018, pp. 3521–3530.

[9] C. Xie, M. Chen, P.-Y. Chen, and B. Li, "CRFL: Certifiably robust federated learning against backdoor attacks," in *International Conference on Machine Learning*, 2021, pp. 11 372–11 382.

[10] K. Zhang, G. Tao, Q. Xu, S. Cheng, S. An, Y. Liu, S. Feng, G. Shen, P.-Y. Chen, S. Ma, and X. Zhang, "FLIP: A provable defense framework for backdoor mitigation in federated learning," in *International Conference on Learning Representations*, 2023.

[11] Z. Qin, L. Yao, D. Chen, Y. Li, B. Ding, and M. Cheng, "Revisiting personalized federated learning: Robustness against backdoor attacks," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023, pp. 4743–4755.

[12] T. Li, S. Hu, A. Beirami, and V. Smith, "Ditto: Fair and robust federated learning through personalization," in *International Conference on Machine Learning*, vol. 139, 2021, pp. 6357–6368.

[13] S. Lin, Y. Han, X. Li, and Z. Zhang, "Personalized federated learning towards communication efficiency, robustness and fairness," *Advances in Neural Information Processing Systems*, vol. 35, pp. 30 471–30 485, 2022.

[14] T. Ye, C. Chen, Y. Wang, X. Li, and M. Gao, "BapFL: You can backdoor personalized federated learning," *ACM Transactions on Knowledge Discovery from Data*, vol. 18, no. 7, pp. 1–17, 2024.

[15] L. Collins, H. Hassani, A. Mokhtari, and S. Shakkottai, "Exploiting shared representations for personalized federated learning," in *International conference on machine learning*, vol. 139, 2021, pp. 2089–2099.

[16] K. Pillutla, K. Malik, A.-R. Mohamed, M. Rabbat, M. Sanjabi, and L. Xiao, "Federated learning with partial model personalization," in *International Conference on Machine Learning*, vol. 162, 2022, pp. 17 716–17 758.

[17] P. Foret, A. Kleiner, H. Mobahi, and B. Neyshabur, "Sharpness-aware minimization for efficiently improving generalization," in *International Conference on Learning Representations*, 2021.

[18] J. Kwon, J. Kim, H. Park, and I. K. Choi, "ASAM: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks," in *International Conference on Machine Learning*, vol. 139, 2021, pp. 5905–5914.

[19] M. Zhu, S. Wei, L. Shen, Y. Fan, and B. Wu, "Enhancing fine-tuning based backdoor defense with sharpness-aware minimization," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2023, pp. 4466–4477.

[20] Y. Guo, Y. Sun, R. Hu, and Y. Gong, "Hybrid local SGD for federated learning with heterogeneous communications," in *International Conference on Learning Representations*, 2022.

[21] Z. Zhang, Z. Gao, Y. Guo, and Y. Gong, "Scalable and low-latency federated learning with cooperative mobile edge networking," *IEEE Transactions on Mobile Computing*, vol. 23, no. 1, pp. 1–11, 2022.

[22] Z. Zhang, Y. Guo, Y. Fang, and Y. Gong, "Communication and energy efficient wireless federated learning with intrinsic privacy," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 4, pp. 4035–4047, 2023.

[23] Z. Zhang, Z. Gao, Y. Guoh, and Y. Gong, "Heterogeneity-aware cooperative federated edge learning with adaptive computation and communication compression," *IEEE Transactions on Mobile Computing*, vol. 24, no. 3, pp. 2073–2084, 2025.

[24] R. Hu, Y. Guo, and Y. Gong, "Federated learning with sparsified model perturbation: Improving accuracy under client-level differential privacy," *IEEE Transactions on Mobile Computing*, vol. 23, no. 8, pp. 8242–8255, 2024.

[25] Z. Zhang, Y. Guo, and Y. Gong, "pfedSAM: Secure federated learning against backdoor attacks via personalized sharpness-aware minimization," 2024. [Online]. Available: https://openreview.net/forum?id=cz0kQD95o4

[26] T. Gu, K. Liu, B. Dolan-Gavitt, and S. Garg, "BadNets: Evaluating backdooring attacks on deep neural networks," *IEEE Access*, vol. 7, pp. 47 230–47 244, 2019.

[27] H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," 2019. [Online]. Available: https://arxiv.org/abs/1909.06335