Enhancing Federated Learning Robustness using Data-Agnostic Model Pruning

Mark Huasong Meng[1,4], Sin G. Teo[1], Guangdong Bai[2(✉)], Kailong Wang[3,4], and Jin Song Dong[4]

[1] Institute for Infocomm Research (I2R), A*STAR, Singapore
[2] University of Queensland, QLD, Australia
[3] Huazhong University of Science and Technology, China
[4] National University of Singapore, Singapore
{menghs,teo_sin_gee}@i2r.a-star.edu.sg, g.bai@uq.edu.au, wangkl@hust.edu.cn, dcsdjs@nus.edu.sg

Abstract. Federated learning enables multiple data owners with a common objective to participate in a machine learning task without sharing their raw data. At each round, clients train local models with their own data and then upload the model parameters to update the global model. This multi-agent form of machine learning has been shown prone to adversarial manipulation by recent studies. Byzantine attackers impersonated as benign clients can stealthily interrupt or destroy the learning process. In this paper, we propose FLAP, a post-aggregation model pruning technique to enhance the Byzantine robustness of federated learning by effectively disabling the malicious and dormant components in the learned neural network models. Our technique is data-agnostic, without requiring clients to submit their dataset or training output, well aligned with the data locality of federated learning. FLAP is performed by the server right after the aggregation, which renders it compatible with an arbitrary aggregation algorithm and existing defensive techniques. Our empirical study demonstrates the effectiveness of FLAP under various settings. It reduces the error rate by up to 10.2% against the state-of-the-art adversarial models. Moreover, FLAP also manages to increase the average accuracy by up to 22.1% against different adversarial settings, mitigating the adversarial impacts while preserving learning fidelity.

1 Introduction

Federated learning (FL) is a machine learning (ML) technique that collaboratively trains a model from decentralized datasets [14]. Unlike the traditional ML that trains a model using a centralized dataset, FL adopts a distributed paradigm where multiple clients contribute to training a model from their local data. Due to the heterogeneity of the data owned by different parties, FL exhibits the great capacity of mitigating the fairness issue from data bias. On the other hand, FL reverses the stereotype that ML can only be carried out in a computationally intensive setting. Through the cluster effect [6], FL enables mobile and edge devices to participate in solving complex real-world problems, such

as financial services [13], cybersecurity [25], healthcare [28, 19], and knowledge discovery [21].

FL is designed to preserve participants' privacy and locality of their data [10, 14, 29]. This process, however, is prone to be manipulated by malicious clients since their data and training processes are not transparent to the server and other participants. In addition, the *Byzantine failure* is a major threat to FL due to its distributed nature. There is no guarantee that every client has faithfully uploaded the trained model to the server. Many attacks exploiting these issues have been discovered by a recent study [5]. For example, *poisoning* is one of the most studied attack methods [1]. Malicious clients can collude with each other and commit a Byzantine attack by intentionally training on adversarial data [4] or directly uploading erroneous model parameters to the server [2, 27]. Unfortunately, such distributed poisoning attack is hard to be detected. Notably, an existing study [2] shows that a poisoning attack can be achieved by merely one malicious client launching a one-shot attack in FL.

To tackle the potential attacks, the research community has proposed multiple defense techniques. Most endeavors pursue Byzantine-robust FL through *Byzantine-resilient aggregations*. It is usually achieved by diverse aggregation algorithms, such as multi-Krum [2] and trimmed mean [27], to detect and eliminate the malicious impact on the global model. Unfortunately, as shown by a recent study [5], these Byzantine-resilient aggregations are only effective when the attacker has no extra knowledge about the FL than benign clients. The FL can still be compromised in case the attacker knows information such as the defensive aggregation technique adopted by the server. As a result, some *auxiliary defense* approaches that cooperate with aggregation algorithms are proposed to address their fragility. They are essentially not a part of the conventional FL process but demonstrate promising efforts towards Byzantine robustness.

Recent advances in auxiliary defenses tend to enhance the robustness of the global model prior to the aggregation. Representative studies include taking advantage of a dedicated dataset to exclude certain clients' updates incurring abnormal test accuracy and/or loss values [3, 5] and performing a supervised model pruning based on clients' voting [26]. They either require a dataset from a similar distribution of clients' training data, demand the population of attackers among participating clients, or assume participating clients are honest all the time. However, these prerequisites may not be realistic, especially in an adversarial environment. An effective defense technique is needed to strengthen the Byzantine robustness of FL.

In this paper, we propose a post-aggregation defense technique for FL by data-agnostic model pruning named FLAP (FL by data-Agnostic Pruning)[5]. FLAP can be performed by the server independently with no reliance on training data and extra contributions from clients. It is not limited by the estimated population of malicious clients. More importantly, FLAP is designed with a generic FL framework that is compatible with diverse aggregation settings. It can

[5] Our source code is hosted at https://github.com/mark-h-meng/flap.

be deployed either alone or together with existing Byzantine-robust techniques to boost their defenses.

Our design is motivated by an insight that model pruning could disable the insignificant and dormant parameters, which are often introduced by poisoning attacks. FLAP aims to enhance the robustness of the global model in an adversarial environment. Meanwhile, it should preserve learning fidelity to the maximum extent. Therefore, we adopt a conservative pruning strategy rather than the cut-and-re-train approach that has been conventionally applied by centralized learning paradigms. More specifically, our pruning aims to remove the parameters with the most negligible impact on the model's output. To this end, FLAP dynamically measures the effect of deleting a unit to identify pruning candidates. It adopts a scale-based sampling strategy for convolutional (Conv) layers and a cross-layer saliency-based sampling strategy for fully-connected (FC) layers. Our evaluation shows that FLAP is effective in preserving robustness and fidelity against diverse adversarial settings. Meanwhile, it is capable of boosting the state-of-the-art (SOTA) defenses towards a higher degree of Byzantine robustness. Our key contributions are summarized below.

- We propose FLAP, a novel FL pruning technique that does not rely on an estimation of malicious clients' population and makes no request for the cooperation of participating clients. It is implemented as an auxiliary defense for generic FL that can be added to any form of existing FL applications.
- We conduct an empirical study to explore the effectiveness of FLAP in an adversarial environment. We find FLAP can enhance the robustness of FL with different aggregation algorithms while preserving the model fidelity.
- We test FLAP against different advanced adversarial models and compare it with the SOTA defenses. Our empirical study shows that FLAP outperforms the existing defense techniques in all adversarial models and boosts the existing defenses for a higher degree of Byzantine robustness.

2 Related Work

Attacking FL. Poisoning attacks for ML can be categorized into *untargeted attacks* [5] and *targeted attacks* [4, 16]. The former is performed to reduce the overall learning accuracy on arbitrary inputs, and the latter aims to precisely misclassify a limited set of classes. An untargeted attack is shown not practical to the extent of FL, as it can be defended at a low cost [20]. Targeted attacks can be further classified into *label-flipping attacks* [9] and *backdoor attacks* [4]. In this paper, we select the targeted label-flipping attacks as the default approach.
Byzantine-robust FL. Defending FL systems is a widely-studied topic. The dominant baseline defenses include *median* [27], *trimmed mean* [27], *multi-Krum* [2], and *Bulyan* [8]. Among them, median and trimmed mean are *statistical-based aggregations* that assess each client's update independently. multi-Krum and Bulyan are representative *distance-based aggregations*. Those baseline algorithms are later found to be fragile to fine-crafted adversarial models [5]. Recent

improvements that address the flaws of baseline defenses mainly focus on proactively detecting malicious clients prior to the aggregation [3, 26] and strengthening the existing FL framework to minimize the impact of malicious upload [11, 17]. Besides that, there are also auxiliary defenses that co-exist with these baselines and address their shortcomings. The SOTA defenses include the *Error Rate based Rejection* (ERR), the *Loss Function based Rejection* (LFR) and the *Union Rejection* (ERR+LFR) [5]. They are shown effective against various adversarial models but demand a proper estimation of malicious clients' population. Cao et al. [3] proposed another Byzantine-robust FL framework by trust bootstrapping. However, it may require the cooperation of the clients because the server needs to collect a small clean dataset from their training data.

Neural Network Pruning in FL. Pruning is a commonly applied model optimization technique [7]. It is considered useful in FL based on an insight that the defense against targeted attack can be achieved by removing not only the poisoned data but also the activation of adversarial inputs in the model [12, 24]. Wu et al. [26] proposed a post-training FL defense by pruning and fine-tuning the global model. However, it relies on a voting process among participating clients, which entails sharing their local models' activation results. That may not be practical because participants may be reluctant to share any knowledge about the models' output based on their own training data, given that disclosing them is prone to a membership attack [18]. In this paper, we study the adoption of pruning that does not rely on the training data, i.e., data-agnostic pruning [15, 22], and therefore can be solely performed by the server without explicitly asking for clients' cooperation.

3 Problem Statement

3.1 Federated Learning

We assume a standard context of FL, in which data is not identically and independently distributed across multiple clients. A client i can only access his/her own data D_i, $i = 1, 2, ..., n$. The server does not have access to clients' data. The learning process is performed in multiple rounds in a synchronous manner.

During an arbitrary round t, clients receive a global model w^{t-1}_{Global} from the server and perform continuous learning with his/her own data D_i, followed by sending the update of the local model's parameters, i.e., $g^t_i = w^t_i - w^{t-1}_{\text{Global}}$, to the server. All the clients' updates would then undergo an aggregation procedure by the server prior to moving to the next round. Let α be the learning rate, the global model can be defined as follows:

$$w^t_{\text{Global}} = w^{t-1}_{\text{Global}} + \alpha \cdot g^t_{\text{Global}} \tag{1}$$

The ultimate goal of FL is to find an aggregated update g^t_{Global} from the clients at a certain round t to result in a minimal loss function $\mathcal{L}(D, w^t_{\text{Global}})$ on the joint training dataset $D = \cup^n_{i=1} D_i$.

We assume a standard aggregation named *FedAvg* [14] as the default option to compute the global update unless otherwise specified. FedAvg calculates the average of the clients' updates as the global update, which can be formally defined as $g^t_{\text{Global}} = \sum_{i=1}^{n} \frac{|D_i|}{|D|} g^t_i$, where $|D|$ and $|D_i|$ represent the size of the joint training dataset and the size of the client i's local training set. In addition to FedAvg, we also consider the server to take some defensive measures to thrive for a resilient and safe global model, which we will detail in Section 5.

3.2 Threat Model

This paper assumes that the attack of FL adopts a targeted label-flipping attack by model poisoning, which is to maximize the possibility of misclassification of the targeted data samples. The malicious clients are granted no extra privileges than the benign ones, such that they can only access their own training sets and model parameters. However, they can still be manipulated by a single attacker to deploy a Byzantine attack. In other words, malicious clients will collude with each other toward the same attack goal. During an attack round, the malicious clients intentionally learn certain (victim) data $\langle x^v_i, y^v_i \rangle \in D_i$ with a wrong (target) label y^τ_i, e.g., learning all digit 1 as digit 7, and learn the remaining data correctly.

Apart from the conventional attack, we also consider that malicious clients can take advantage of the strengthened adversarial models to craft local models [5]. Thus, we anticipate the malicious clients to apply two additional strengthened adversarial models. We brief them below and later assess them in Section 5.

Partial knowledge attack. A malicious client can access all other colluding malicious clients' local models and has knowledge about the aggregation rule on the server side. Through analyzing the parameters of other malicious clients, it can craft the local update to influence the direction of the global model update, which subsequently undermines the robustness of the global model.

Full knowledge attack. On the basis of a partial knowledge attack, a malicious client has full access to the training sets and local models of all participating clients, i.e., the entire FL is completely transparent to the attacker. This scenario is helpful for us to assess *the upper bound* of an attacker's capability and estimate the impact of adversarial manipulation.

4 Proposed FLAP

4.1 Approach Overview

Fig. 1 shows how FLAP is used in FL. Overall, it is a typical FL process except for the addition of the post-aggregation model pruning at the server side (Phase 4). FL with FLAP executes an iterative process that begins with broadcasting the *global model* (Phase 1). For each client, it performs training over the received global model with its own data set (Phase 2). This training process is supposed to be private so that other clients and the server have no access to it. Upon the completion of training, the client sends the learned *local model* parameters'

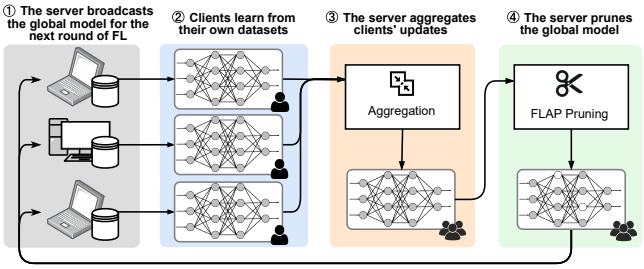

Fig. 1. The workflow of federated learning with FLAP

updates to the server. Once the server has received all participating clients' updates, it aggregates the parameters and produces the global model (Phase 3). Next, FLAP performs model pruning over the newly generated global model (Phase 4) and marks the end of the current round of learning. If the learning is not concluded, the server will broadcast the newly pruned global model to all participating clients and start the next round of learning.

4.2 Model Pruning

Our data-agnostic pruning supports two types of hidden layers of neural network models: the FC layers and the Conv layers. For each time the pruning is launched, FLAP samples the units on the supported hidden layer and nominates a fixed proportion (e.g., 1%) of units to cut. Unlike the conventional data-hungry pruning that adopts an aggressive cut-and-re-train strategy, we design our pruning as a conservative approach that always prefers to remove the units that incur the most insignificant impact to the model output, based on an insight that many attack models stealthily plant their adversarial patterns in those units [12, 24]. Although we are given a chance of continuing training in FL, applying a proper conservative pruning technique is helpful in preserving model fidelity. To this end, we propose different sampling strategies for the two layer types to nominate pruning candidates.

Conv layers sampling. When handing a Conv layer, FLAP calculates the l_1-norm of all the filters, sorts them, and nominates a few filters with the least norm values as pruning candidates. The norm function is selected as the pruning criteria based on the assumption that the malicious clients tend to exploit the neural network through minor and imperceptible filters, and then manipulate their activation to misclassify. As a result, FLAP removes those filters that have been trained with the least significant values.

FC layers sampling. Finding pruning candidates at FC layers is comparably more complicated because the parameters within a FC layer tend to be very similar to each other if we do not take the *cross-layer computation* into account. The least value-based pruning without local retraining usually causes severe performance degradation to the model. FLAP takes advantage of the data-agnostic cross-layer saliency-based pruning that has been studied in [15], where the pruning is conducted in a pair-wise manner. Given a candidate pair of hidden

units, we remove one of its units and double the weight of the other unit, which is expected to supersede the role of the pruned unit.

To find the proper units to be pruned, we first iterate all units in a FC layer and form pairs for them. We then assess the *propagated impact* of pruning a pair of hidden units in the middle layer, which is calculated as a range/interval depending on the legitimate value ranges of the model input. The potential *impact of pruning* is jointly measured by the l_1-norm and *entropy* of the propagated impact. When FLAP prunes a FC layer, it sorts all unit pairs by their impact values and removes the pairs with the lowest impact values.

Global model pruning. Considering the pruning process is proposed to reinforce the reliability of the FL model, FLAP does not alter the model structure during the pruning. Instead, it zeros out the parameter values of all the pruned units. The pruned model becomes the global model of the next round of FL and will be broadcast to all clients for their local training. Overall, we let p denote the pruning process, and thus, the global model can be defined by a modified form of Equation 1 as follows:

$$w^t_{\text{Global}} = p\left(w^{t-1}_{\text{Global}} + \alpha \cdot g^t_{\text{Global}}\right) \tag{2}$$

5 Evaluation

5.1 Experiment Setup

We implement FL based on a public repository[6]. The FLAP and the tested adversarial models are implemented based on TensorFlow. All the presented results are the median value observed from at least five repeated executions.

Federated learning. The simulated FL is composed of 80 participating clients. The distributed training is uniformly configured for each client. Each round of the client's local training consists of two epochs, with the learning rate set to 0.001 and the Adam optimizer applied. FedAvg is selected as the default aggregation algorithm. Our experiments begin with 20 rounds of benign training, which can help the global model maintain stable prediction accuracy. We consider malicious clients to start attacking from the 21st round. By default, 16 out of 80 participating clients (20%) are malicious that collude with each other in conducting model poisoning attacks. We uniformly set the attack goal as misleading the global model to predict the first class as the seventh class.

Pruning. We note that the pruning may not be necessarily carried out for every round of FL. Instead, we stipulate that the server performs pruning from the first round and repeats every five rounds. For each pruning operation, the server zeros out 1% of filters from every Conv layer and 1% of hidden units from every FC layer, by zeroing out their corresponding parameters[7].

Models and datasets. We select FEMNIST and use three different model architectures in our evaluation, including a five-layer MLP, a LeNet-5 ConvNet, and a ResNet-18 model.

[6] https://github.com/pps-lab/fl-analysis

[7] One unit will be pruned if the layer has less than 100 units.

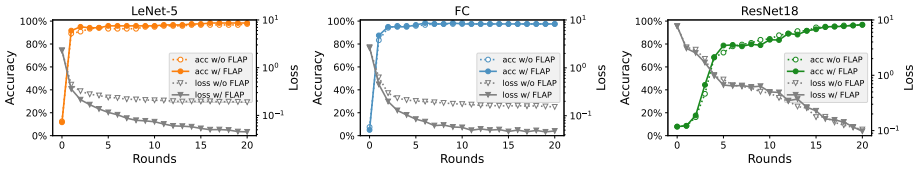

Fig. 2. Test accuracy and loss of FL up to round 20, with and without FLAP

5.2 Benign FL with FLAP

Our first set of experiments aims to investigate if FLAP suits the FL as a post-aggregation optimization. More specifically, we wish to figure out the question "*how does* FLAP *preserve the fidelity of FL in a non-adversarial circumstance?*". To this end, we apply FLAP on all three models under the benign settings and compare the learning process with the FL without it. We evaluate the global model after each round's aggregation and record the test accuracy and loss.

Fig. 2 presents the test accuracy and loss value of the first 20 rounds, with and without the equipment of FLAP. We find that the growth of test accuracy of models with FLAP is almost identical with the models without it. We also observe that the adoption of FLAP accelerates the loss descent on LeNet-5 and MLP models. In summary, FLAP shows promising fidelity preservation on all three models. The adoption of FLAP does not impair the learning process.

5.3 FLAP in Adversarial Settings

Next, we explore *whether* FLAP *can boost existing defensive techniques towards Byzantine-robust FL*. We focus on using the ResNet-18 model to compare FLAP with the existing techniques against various adversarial models.

Starting from round 21, we deploy a model poisoning attack for ten rounds to the default setting of FL. We test our approach with two representative Byzantine-resilient aggregation algorithms. We also learn that the malicious clients in FL may strengthen their attack capacity by gaining extra knowledge from the server and benign clients, and therefore we assess our approach with two advanced adversarial models proposed by Fang et al. [5], namely partial knowledge attack and full knowledge attack.

In our experiments, we evaluate the robustness of the global model by calculating the average error rate of consecutive ten rounds of adversarial learning. We also record the average test accuracy of the global model to reflect the overall learning process. The experimental results can be found in Table 1.

Byzantine-resilient Aggregations. We replace FedAvg with Byzantine resilient aggregations, namely trimmed mean and multi-Krum, and repeat our previous experiments with the default adversarial settings. Both algorithms are configurable with a parameter, which defines the estimated *upper bound* of Byzantine attackers among all participating clients [2]. For each algorithm, we define three modes, named *conservative* (C mode), *perfect* (P mode) and *radical* (R mode). The C mode simulates the server underestimating the existence of malicious clients, the R mode stipulates that the server overestimates the population

Table 1. Average error rates and test accuracy of FL (ResNet-18) in various adversarial settings, with (shown in **bold** text) and without FLAP (shown in plain text).

Aggregation Rules	Auxiliary Defense	Adversarial Modes		
		Targeted Label Flipping	Partial Knowledge	Full Knowledge
Error Rate (Lower is Better)*				
FedAvg	**Nil**	30.8%, **20.0%** (▼)	62.8%, **20.0%** (▼)	40.0%, **20.0%** (▼)
Trimmed Mean†	**Nil**	(C) 87.0%, **74.7%** (▼)	(C) 72.5%, **62.9%** (▼)	(C) 67.2%, **35.4%** (▼)
		(P) 30.0%, **17.5%** (▼)	(P) 22.9%, **17.5%** (▼)	(P) 38.1%, **33.0%** (▼)
		(R) 11.4%, **9.8%** (▼)	(R) 11.5%, **9.8%** (▼)	(R) 12.7%, **10.6%** (▼)
	ERR+ LFR	(C) 59.2%, **20.0%** (▼)	(C) 80.0%, **71.3%** (▼)	(C) 84.6%, **68.3%** (▼)
		(P) 16.8%, **15.6%** (▼)	(P) 16.9%, **15.6%** (▼)	(P) 22.7%, **16.3%** (▼)
		(R) 5.8%, **5.8%** (=)	(R) 5.8%, **4.2%** (▼)	(R) 9.6%, **9.3%** (▼)
Multi-Krum†	**Nil**	(C) 84.3%, **83.7%** (▼)	(C) 93.3%, **74.8%** (▼)	(C) 83.5%, **73.9%** (▼)
		(P) 22.7%, **20.0%** (▼)	(P) 22.7%, **15.6%** (▼)	(P) 20.6%, **14.2%** (▼)
		(R) 28.8%, **27.3%** (▼)	(R) 28.9%, **19.5%** (▼)	(R) 22.3%, **20.0%** (▼)
	ERR+ LFR	(C) 84.6%, **83.8%** (▼)	(C) 79.2%, **68.3%** (▼)	(C) 83.5%, **75.4%** (▼)
		(P) 22.7%, **20.0%** (▼)	(P) 22.7%, **16.3%** (▼)	(P) 20.6%, **14.2%** (▼)
		(R) 28.7%, **22.3%** (▼)	(R) 27.9%, **21.0%** (▼)	(R) 28.1%, **17.9%** (▼)
Test Accuracy (Higher is Better)*				
FedAvg	**Nil**	10.3%, **10.9%** (▲)	10.1%, **10.1%** (=)	9.0%, **9.2%** (▲)
Trimmed Mean†	**Nil**	(C) 11.5%, **14.6%** (▲)	(C) 13.5%, **17.3%** (▲)	(C) 15.8%, **18.4%** (▲)
		(P) 92.1%, **97.8%** (▲)	(P) 92.3%, **93.2%** (▲)	(P) 15.8%, **18.4%** (▲)
		(R) 94.6%, **95.1%** (▲)	(R) 93.9%, **94.8%** (▲)	(R) 92.5%, **92.1%** (▼)
	ERR+ LFR	(C) 11.2%, **11.0%** (▼)	(C) 11.3%, **16.5%** (▲)	(C) 12.8%, **14.1%** (▲)
		(P) 93.4%, **93.0%** (▼)	(P) 93.4%, **93.9%** (▲)	(P) 93.4%, **93.2%** (▼)
		(R) 95.8%, **96.0%** (▲)	(R) 94.6%, **94.9%** (▲)	(R) 95.6%, **95.6%** (=)
Multi-Krum†	**Nil**	(C) 34.5%, **56.0%** (▲)	(C) 35.4%, **56.2%** (▲)	(C) 37.6%, **52.2%** (▲)
		(P) 35.6%, **43.5%** (▲)	(P) 36.0%, **44.7%** (▲)	(P) 35.5%, **44.2%** (▲)
		(R) 35.3%, **44.2%** (▲)	(R) 36.0%, **45.9%** (▲)	(R) 35.6%, **46.9%** (▲)
	ERR+ LFR	(C) 34.5%, **56.1%** (▲)	(C) 34.5%, **56.2%** (▲)	(C) 38.8%, **60.9%** (▲)
		(P) 35.5%, **43.4%** (▲)	(P) 36.0%, **44.7%** (▲)	(P) 35.5%, **45.9%** (▲)
		(R) 35.3%, **44.2%** (▲)	(R) 35.3%, **47.7%** (▲)	(R) 35.4%, **46.9%** (▲)

* We use "(▼)", "(=)" and "(▲)" to represent a decrease, no change, and growth, respectively.
† Three parametric settings are adopted in both trimmed mean and multi-Krum aggregation algorithms: (C), (P), and (R) stand for Conservative, Perfect, and Radical modes, respectively.

of malicious clients, and the P mode defines that the server estimates the exact population of malicious clients. These three modes estimate the percentage of malicious clients to be 10%, 30%, and 20%, respectively.

Our results show that the adoption of Byzantine-resilient aggregations helps reduce the error rate and improve the test accuracy, however, only when the server sufficiently estimates the presence of malicious clients (i.e., the P mode). We learn that any mis-estimiation of the population of malicious clients causes a negative impact on the FL in terms of test accuracy and error rate. This highlights the necessity of defensive techniques that are independent of the server's knowledge regarding the attackers' population.

Our evaluation demonstrates that FLAP can improve the FL in all three modes of the two aggregation algorithms. The adoption of FLAP reduces the error rate by up to 12.5% for the trimmed mean model (P mode) and 2.7% for the multi-Krum model (P mode). On this basis, FLAP also helps FL to better converge as we record a growth of test accuracy at up to 5.7% from the trimmed mean model (P mode) and 21.5% from the multi-Krum model (C mode).

Advanced Adversarial Models. Our next set of experiments aims to investigate *whether* FLAP *makes FL more Byzantine-robust against the SOTA adversarial models*. We simulate the two adversarial models specially designed for FL [5] and evaluate our approach against them. Moreover, we also compare our

approach with the prediction-based defenses proposed in the same paper. We take the most radical defense named *ERR+LFR* as the baseline[8].

From Table 1, we observe that the two adversarial models overall stimulate the adversarial effectiveness when a defensive aggregation algorithm is deployed. They incur a higher error rate without significantly impairing the test accuracy and therefore can more stealthily poison the global model. Besides that, the ERR+LFR defense is shown to be effective in most cases, especially when the aggregation algorithms are deployed in P mode and R mode.

We also find that our approach can boost the ERR+LFR as a post-aggregation defense. We record an error rate reduction at up to 10.2% (multi-Krum R mode) and a test accuracy increment at up to 22.1% (multi-Krum C mode) by applying FLAP to the ERR+LFR defense against the full-knowledge attack. Even if we compare the models that are either equipped with ERR+LFR only (i.e., without FLAP) or FLAP only (i.e., absence of ERR+LFR), we find our approach still achieves a lower error rate than the ERR+LFR defense in all scenarios of the multi-Krum setting and two out of six scenarios of the trimmed mean setting. FLAP also manages to outperform the ERR+LFR defense in 14 out of 18 scenarios of two settings with regard to the test accuracy, indicating FLAP better assists the FL towards the learning target.

Summary. FLAP is shown effective towards Byzantine-robust FL in both benign and adversarial environments. It can co-exist with existing defenses including Byzantine-resilient aggregations and auxiliary prediction-based techniques and even outperforms them in most cases. More importantly, FLAP can boost those defenses to achieve a higher degree of Byzantine robustness, especially when the server underestimates the presence of malicious clients.

6 Discussion

Limitations and Future Work. First, FLAP supports pruning for both FC and Conv layers. For that reason, we may achieve a higher degree of Byzantine robustness if we broaden the FLAP's support for pruning residual blocks. In addition, the server has no access to the training set but owns some data in a similar distribution for testing purposes. That gives us a chance to prune the model in a supervised manner with the testing data. We aim to explore the test set guided pruning in the future.

Broader Impacts. To the best of our knowledge, this is the first work that explores the pruning by the FL server without the reliance on clients' contribution. FLAP does not request any training data or training outputs from clients, therefore it is difficult to be manipulated by malicious participants. It takes place after the aggregation so that it can co-exist with the existing defenses and boost their effectiveness. This paper will help the research community's exploration of more defense techniques to be adopted in FL, and contribute to achieving efficient privacy-preserving machine learning [23, 29].

[8] We assume the perfect estimation that 20% of clients are excluded due to high loss function value and another 20% of clients are excluded due to low accuracy.

7 Conclusion

In this paper, we propose FLAP, a post-aggregation pruning technique to boost the Byzantine robustness of FL, based on our insight that pruning can effectively mitigate the unfavorable and malicious parameters learned in adversarial training. We evaluate the proposed FLAP with different models, assess its effectiveness against different adversarial models, and compare it with existing defensive techniques. Our empirical study demonstrates that FLAP can reduce the error rate and preserve the fidelity of FL equipped with different aggregation algorithms under various adversarial settings. FLAP also shows a promising capacity to reinforce the existing defensive techniques against the SOTA adversarial models to achieve a higher degree of Byzantine robustness.

Acknowledgment. This work was supported by The University of Queensland under the NSRSG grant 4018264-617225, Cyber Research Seed Funding, the Global Strategy and Partnerships Seed Funding, and Agency for Science, Technology and Research (A*STAR) Singapore under the ACIS scholarship.

References

1. Bhagoji, A.N., Chakraborty, S., Mittal, P., Calo, S.: Analyzing federated learning through an adversarial lens. In: International Conference on Machine Learning (2019)
2. Blanchard, P., Mhamdi, E.M.E., Guerraoui, R., Stainer, J.: Machine learning with adversaries: Byzantine tolerant gradient descent. In: Guyon, I., von Luxburg, U., Bengio, S., Wallach, H.M., Fergus, R., Vishwanathan, S.V.N., Garnett, R. (eds.) Advances in neural information processing systems. pp. 119–129 (2017)
3. Cao, X., Fang, M., Liu, J., Gong, N.Z.: Fltrust: Byzantine-robust federated learning via trust bootstrapping. In: Network and Distributed System Security Symposium. The Internet Society (2021)
4. Chen, X., Liu, C., Li, B., Lu, K., Song, D.: Targeted backdoor attacks on deep learning systems using data poisoning. arXiv preprint arXiv:1712.05526 (2017)
5. Fang, M., Cao, X., Jia, J., Gong, N.: Local model poisoning attacks to byzantine-robust federated learning. In: 29th USENIX Security Symposium (2020)
6. Fang, U., Li, J., Akhtar, N., Li, M., Jia, Y.: Gomic: Multi-view image clustering via self-supervised contrastive heterogeneous graph co-learning. World Wide Web pp. 1–17 (2022)
7. Guan, H., Xiao, Y., Li, J., Liu, Y., Bai, G.: A comprehensive study of real-world bugs in machine learning model optimization. In: Proceedings of the International Conference on Software Engineering (2023)
8. Guerraoui, R., Rouault, S., et al.: The hidden vulnerability of distributed learning in byzantium. In: International Conference on Machine Learning (2018)
9. Huang, L., Joseph, A.D., Nelson, B., Rubinstein, B.I., Tygar, J.D.: Adversarial machine learning. In: Proceedings of the 4th ACM workshop on Security and artificial intelligence. pp. 43–58 (2011)
10. Jin, C., Wang, J., Teo, S.G., Zhang, L., Chan, C., Hou, Q., Aung, K.M.M.: Towards end-to-end secure and efficient federated learning for xgboost (2022)
11. Li, T., Hu, S., Beirami, A., Smith, V.: Ditto: Fair and robust federated learning through personalization. In: International Conference on Machine Learning (2021)

12. Liu, K., Dolan-Gavitt, B., Garg, S.: Fine-pruning: Defending against backdooring attacks on deep neural networks. In: International Symposium on Research in Attacks, Intrusions, and Defenses. pp. 273–294. Springer (2018)
13. Mahalle, A., Yong, J., Tao, X., Shen, J.: Data privacy and system security for banking and financial services industry based on cloud computing infrastructure. In: IEEE International Conference on Computer Supported Cooperative Work in Design (2018)
14. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: International Conference on Artificial Intelligence and Statistics (2017)
15. Meng, M.H., Bai, G., Teo, S.G., Dong, J.S.: Supervised robustness-preserving data-free neural network pruning. In: International Conference on Engineering of Complex Computer Systems (2023)
16. Meng, M.H., Bai, G., Teo, S.G., Hou, Z., Xiao, Y., Lin, Y., Dong, J.S.: Adversarial robustness of deep neural networks: A survey from a formal verification perspective. IEEE Transactions on Dependable and Secure Computing (2022)
17. Panda, A., Mahloujifar, S., Bhagoji, A.N., Chakraborty, S., Mittal, P.: Sparsefed: Mitigating model poisoning attacks in federated learning with sparsification. In: International Conference on Artificial Intelligence and Statistics (2022)
18. Salem, A., Zhang, Y., Humbert, M., Berrang, P., Fritz, M., Backes, M.: Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. In: Network and Distributed System Security Symposium (2019)
19. Shaik, T., Tao, X., Higgins, N., Gururajan, R., Li, Y., Zhou, X., Acharya, U.R.: Fedstack: Personalized activity monitoring using stacked federated learning. Knowledge-Based Systems **257**, 109929 (2022)
20. Shejwalkar, V., Houmansadr, A., Kairouz, P., Ramage, D.: Back to the drawing board: A critical evaluation of poisoning attacks on production federated learning. In: IEEE Symposium on Security and Privacy. pp. 1354–1371. IEEE (2022)
21. Song, X., Li, J., Cai, T., Yang, S., Yang, T., Liu, C.: A survey on deep learning based knowledge tracing. Knowledge-Based Systems **258**, 110036 (2022)
22. Srinivas, S., Babu, R.V.: Data-free parameter pruning for deep neural networks. In: Proceedings of the British Machine Vision Conference (2015)
23. Teo, S.G., Cao, J., Lee, V.C.: Dag: a general model for privacy-preserving data mining. IEEE Transactions on Knowledge and Data Engineering **32**(1), 40–53 (2018)
24. Wang, B., Yao, Y., Shan, S., Li, H., Viswanath, B., Zheng, H., Zhao, B.Y.: Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In: IEEE Symposium on Security and Privacy (SP). pp. 707–723. IEEE (2019)
25. Wang, K., Zhang, J., Bai, G., Ko, R., Dong, J.S.: It's not just the site, it's the contents: intra-domain fingerprinting social media websites through cdn bursts. In: Proceedings of the Web Conference (2021)
26. Wu, C., Yang, X., Zhu, S., Mitra, P.: Mitigating backdoor attacks in federated learning. arXiv preprint arXiv:2011.01767 (2020)
27. Yin, D., Chen, Y., Ramchandran, K., Bartlett, P.L.: Byzantine-robust distributed learning: Towards optimal statistical rates. In: International Conference on Machine Learning (2018)
28. Yin, H., Song, X., Yang, S., Li, J.: Sentiment analysis and topic modeling for covid-19 vaccine discussions. World Wide Web **25**(3), 1067–1083 (2022)
29. Zhang, Y., Bai, G., Li, X., Curtis, C., Chen, C., Ko, R.K.: Privcoll: Practical privacy-preserving collaborative machine learning. In: European Symposium on Research in Computer Security (2020)