OBLIVION: Poisoning Federated Learning by Inducing Catastrophic Forgetting

Chen Zhang[†*], Boyang Zhou[‡*], Zhiqiang He[‡], Zeyuan Liu[‡], Yanjiao Chen[‡], Wenyuan Xu[‡], and Baochun Li[§]

[†]Department of Computing, The Hang Seng University of Hong Kong, Hong Kong, China
[‡]College of Electrical Engineering, Zhejiang University, Hangzhou, China
[§]Department of Electrical and Computer Engineering, University of Toronto, Toronto, Canada
czhang@hsu.edu.hk, {zhouboyang, zhiqiang_ee, zeyuan_liu, wyxu}@zju.edu.cn, bli@ece.toronto.edu

***Abstract*—Federated learning is exposed to model poisoning attacks as compromised clients may submit malicious model updates to pollute the global model. To defend against such attacks, robust aggregation rules are designed for the centralized server to winnow out outlier updates, and to significantly reduce the effectiveness of existing poisoning attacks. In this paper, we develop an advanced model poisoning attack against defensive aggregation rules. In particular, we exploit the *catastrophic forgetting* phenomenon during the process of continual learning to destroy the memory of the global model. Our proposed framework, called OBLIVION, features two special components. The first component prioritizes the weights that have the most influence on the model accuracy for poisoning, which induces a more significant degradation on the global model than equally perturbing all weights. The second component smooths malicious model updates based on the number of selected compromised clients in the current round, adjusting the degree of poisoning to suit the dynamics of each training round. We implement a fully-functional prototype of OBLIVION in PLATO, a real-world scalable federated learning framework. Our extensive experiments over three datasets demonstrate that OBLIVION can boost the attack performance of model poisoning attacks against unknown defensive aggregation rules.**

I. INTRODUCTION

Federated learning is a popular distributed learning paradigm that has already been adopted by many well-known companies such as Google [1], Apple [2], and Facebook [3]. In federated learning, a central server aggregates local updates submitted by edge devices (referred to as clients) to establish a global model. The fact that local datasets are kept private by clients gives rise to the concern of model poisoning attacks, where a fraction of compromised clients submit malicious model updates to pollute the global model, as shown in Fig. 1. To deal with such risks, a line of defensive aggregation rules (AGRs) have been developed to sift through the model updates for outliers [4]–[9]. Although adaptive model poisoning attacks [10]–[12] have been proposed to constrain the poisoning perturbations so as to avoid being discarded by the AGR, it is shown in a recent study that the effectiveness of these attacks was greatly suppressed by state-of-the-art defensive AGRs.

* indicates equal contribution
Corresponding author: Yanjiao Chen, chenyanjiao@zju.edu.cn

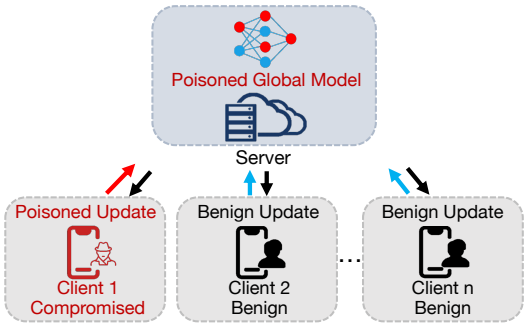

Fig. 1: Model poisoning attacks in federated learning.

Inspired by the *catastrophic forgetting* phenomenon observed in continual learning [13], [14], we seek to boost the effectiveness of model poisoning attacks by concentrating on destroying the model weights that are most important to the learning task. More specifically, we develop a model poisoning framework, called OBLIVION, that aims to induce amnesia of the global model by offsetting the memory of benign updates. OBLIVION features two novel components to solve the following corresponding challenges:

▷ *How do we adjust the amount of poison perturbation over different weights?*

Not all model weights are created equal. However, to our best knowledge, all existing works treat the weights in the global model as equal; i.e., the total budget of poisoning perturbations is equally divided across the model weights. Unfortunately, to evade being abandoned by defensive aggregation rules, the total budget of poisoning perturbations is bounded, usually by a tight constraint. As the global model in federated learning normally has a large scale (e.g., millions of weights), the poisoning perturbation assigned to each weight is infinitesimal, imposing a minuscule impact on the global model. To deal with this issue, we propose to prioritize the weights that are most critical in learning for poisoning, inspired by explanatory work that only a few weights memorize the most useful information during training, while others are redundant. To assess the importance of weights, we leverage the Elastic Weight Coalescing (EWC) [15] method, which is used to score the importance of weights based on their changes during training. With the weight prioritization component, we

can carefully channel the poison budget to achieve the highest poisoning effect.

▷ *How can we continually degrade the influence of previously learned knowledge?*

Existing works compute the current amount of poisoning perturbations to disturb a benign update in the current round. Nonetheless, only considering the current training round limits the poisoning effect for two reasons. *First*, federated learning is a continual iterative learning process where the training results of previous rounds will continue to influence the global model. *Second*, training results in each round have large variations as different sets of benign and compromised clients are selected by the server to participate in training. To account for these issues, we design a dynamic smoothing mechanism that considers the momentum history of benign and malicious updates during the model poisoning process.

Equipped with both weight prioritization and dynamic smoothing components, OBLIVION intensifies the poisoning effect in federated learning even if defensive aggregation rules were adopted and unknown to the attacker. We have conducted an extensive array of experiments on three datasets to evaluate the effectiveness of OBLIVION against 4 defensive aggregation rules, in comparison with 3 state-of-the-art poisoning attacks. We show that OBLIVION enhances the poisoning effect by as much as 39.2%, which is a significant degradation to the federated learning model considering the scale of the global model. By proposing OBLIVION, we reveal the vulnerability of federated learning to potential advanced poisoning attacks, which may inspire further research into more robust federated learning aggregation rules.

Our original contributions in this paper are as follows.

- ▷ We design OBLIVION, an advanced model poisoning attack framework that prioritizes critical weights for poisoning to induce catastrophic forgetting of the global model under defensive aggregation rules.
- ▷ We optimize the poisoning perturbations by considering the history of model updates with a carefully-designed dynamic smoothing mechanism, and by intensifying the poisoning perturbations of important weights determined by a prioritization algorithm.
- ▷ We conduct an extensive array of performance evaluations in a real-world scalable framework, PLATO, and verify the effectiveness of OBLIVION on 3 datasets under 4 defensive aggregation rules.

II. PRELIMINARIES AND RELATED WORK

A. Federated Learning

Federated learning is a popular form of distributed machine learning in which many data owners (referred to as clients) collaborate to train a global model without sharing private local data. Consider a federated learning scenario with a set $\mathcal{C}$ of clients and a server $\mathcal{S}$. Each client $c_i \in \mathcal{C}$ has a local dataset $\mathcal{D}_i$. The local datasets of clients may not be independent and identically distributed (i.i.d.). Client c_i can use its local dataset $\mathcal{D}_i$ to fine-tune the received global model. Let $\mathcal{L}(\theta_i, \mathcal{D}_i)$ denote the loss function of client c_i, where θ_i denotes the set of model weights. The server maintains a global model θ_G via aggregating the local models uploaded by clients. Specifically, the pipeline for federated learning has three major steps.

1) *Initialization.* The server initializes the global model weights as θ_G^0. It then randomly selects a subset of clients $\mathcal{C}^0 \subseteq \mathcal{C}$ to participate in the first round of training.
2) *Local training.* At the t-th round of training, the server randomly selects a subset of clients $\mathcal{C}^t \subseteq \mathcal{C}$ to receive and fine-tune the global model θ_G^{t-1} using their local datasets. Specifically, a selected client $c_i \in \mathcal{C}^t$ aims to solve the optimization problem $\min_{\theta_i^t} \mathcal{L}(\theta_i^t, \mathcal{D}_i | \theta_G^{t-1})$, where θ_i^t is the local model weights of client c_i at the t-th training round initialized by θ_G^{t-1}. The optimization problem is usually solved by stochastic gradient descent. The model update of client c_i, calculated as $\nabla_i^t = \theta_i^t - \theta_G^{t-1}$, is uploaded to the server.
3) *Server aggregation.* Having received the local updates by all clients in $\mathcal{C}^t$, the server applies a model aggregation rule f_{agr} to aggregate the local updates and obtain the global model update$\nabla_G^t = f_{\text{agr}}(\{\nabla_i^t\}_{c_i \in \mathcal{C}^t})$. The server updates the global model as $\theta_G^t = \theta_G^{t-1} + \eta \nabla_G^t$, where η is the learning rate adjusted to reach stable and fast convergence. The *local training* and *server aggregation* steps are then iterated until convergence, or until the maximum number of training rounds is reached.

B. Model Poisoning Attacks in Federated Learning

In model poisoning attacks, the attacker manipulates model updates sent from compromised clients to the server, aiming at degrading the performance of the global model. According to the goal of the attacker, poisoning attacks can be divided into *targeted poisoning attacks* [17]–[19] and *untargeted poisoning attacks* [11], [20], [21]. Targeted poisoning attacks aim to minimize the accuracy of the global model on specific samples while ensuring high accuracy on the other samples. In contrast, untargeted poisoning attacks aim to minimize the accuracy of the global model indiscriminately on any sample. As untargeted poisoning attacks pose a wider threat to federated learning, we focus on untargeted model poisoning attacks in this work.

Based on the attacker's knowledge about the aggregation rule adopted by the server, existing untargeted model poisoning attacks [10], [12], [16], [22] can be categorized into *AGR-tailored* attacks [22] and *AGR-agnostic* attacks [16].

- ▷ *AGR-tailored attacks*. The attacker is assumed to know the aggregation rule adopted by the server; thus the attacker is able to design its strategies to achieve the best performance under the specific aggregation rule. Nevertheless, the attack strategy may be ineffective if the server changes the aggregation rule.
- ▷ *AGR-agnostic attacks*. The attacker does not have to know the server's aggregation rule. The attack strategies can be generalized to various aggregation rules.

AGR-agnostic attacks are more practical as the server will not reveal its aggregation rule in normal cases. Let ∇^m

TABLE I: Representative AGR-agnostic attack methods.

Attack Methods	Objective Function	Perturbation Function
LIE [16]	$\underset{\gamma}{\text{argmax}}\, \|\|\nabla^b - \nabla^m\|\|_2 \leq \|\|\nabla^b - \nabla^+\|\|_2$ & $\|\|\nabla^- - \nabla^b\|\|_2 \leq \|\|\nabla^- - \nabla^m\|\|_2$	$\nabla^{\text{p}}_{\text{std}}$
Min-Max [12]	$\underset{\gamma}{\text{argmax}} \max_{i\in[n']} \|\|\nabla^m - \nabla_i\|\|_2 \leq \max_{i,j\in[n']} \|\|\nabla_i - \nabla_j\|\|_2$	$\nabla^{\text{p}}_{\text{uv}}$ || $\nabla^{\text{p}}_{\text{std}}$ || $\nabla^{\text{p}}_{\text{sgn}}$
Min-Sum [12]	$\underset{\gamma}{\text{argmax}} \sum_{i\in[n']} \|\|\nabla^m - \nabla_i\|\|_2^2 \leq \max_{i\in[n']} \sum_{j\in[n']} \|\|\nabla_i - \nabla_j\|\|_2^2$	$\nabla^{\text{p}}_{\text{uv}}$ || $\nabla^{\text{p}}_{\text{std}}$ || $\nabla^{\text{p}}_{\text{sgn}}$

denote the crafted malicious update for each compromised client controlled by the attacker. ∇^m is usually formed as the summation of the estimated average benign update ∇^b and the scaled poisoning perturbation $\gamma\nabla^p$, where the poisoning perturbation is generated as $\nabla^p = \mathcal{P}(\nabla^b)$ and γ is a scaling factor. The scaling factor γ scales up ∇_p to intensify the attack performance, lest the benign updates from other clients dilute the poisoning effect. However, γ cannot be too large; otherwise the malicious update may be discarded by the aggregation rule. ∇^b is the average of all benign model updates $\nabla_{\{i\in[n']\}}$ available to the attacker. AGR-agnostic attacks are conducted by designing the perturbation function $\mathcal{P}(\cdot)$ and choosing the scaling factor γ to achieve certain optimization goals.

We now introduce three of the most popular AGR-agnostic attacks that will be compared with our designs in Section V. Table I summarizes the objective and perturbation functions adopted by these attacks.

Little Is Enough (LIE). LIE [16] assumes that benign updates $\nabla_{\{i\in[n']\}}$ are symmetrically distributed around ∇^b. Benign updates lying in the desirable and undesirable directions of the attacker are treated as supporters (∇^+) and opposers (∇^-). The malicious update is crafted between ∇^b and ∇^+ to mislead the server to regard ∇^- as outliers. Specifically, LIE perturbs each dimension of ∇^b by adding the inverse standard deviation $\nabla^{\text{p}}_{\text{std}} = -\text{std}(\nabla_{\{i\in[n']\}})$ scaled by γ.

Minimize Maximum Distance (Min-Max). Min-Max [12] applies three different perturbation functions to optimize the attack performance, i.e., inverse unit vector $\nabla^{\text{p}}_{\text{uv}} = -\frac{\nabla^b}{||\nabla^b||_2}$, inverse standard deviation $\nabla^{\text{p}}_{\text{std}}$, and inverse sign perturbation $\nabla^{\text{p}}_{\text{sgn}} = -\text{sign}(\nabla^b)$. It searches the largest γ such that the distances between the malicious update and any benign update ($||\nabla^m - \nabla_i||_2$) are upper bounded by the maximum distance between any two benign updates ($\max_{i,j\in[n']} ||\nabla_i - \nabla_j||_2$).

Minimize Sum of Distances (Min-Sum). Min-Sum [12] is similar to Min-Max, but uses a different strategy to evade being detected as outliers. γ is computed such that the sum of distances of the malicious update from all benign updates ($\sum_{i\in[n']} ||\nabla^m - \nabla_i||_2^2$) is upper bounded by the sum of distance between any two benign updates ($\max_{i\in[n']} \sum_{j\in[n']} ||\nabla_i - \nabla_j||_2^2$).

III. Problem Formulation

A. System Model

We consider a typical federated learning scenario with a server $\mathcal{S}$ and a set of clients denoted as $\mathcal{C}$. The server is assumed to be a trustworthy entity that aims to aggregate a global model based on the contributions of the clients. There is an attacker who conducts model poisoning attacks via a small subset of controlled clients (referred to as *compromised clients*), denoted as $\mathcal{C}^m$. In each training round, the server dispatches the latest global model to a subset of randomly-selected clients. A benign client who receives the global model will fine-tune the model with its local dataset and then send the model update back to the server. A compromised client who is selected, in contrast, will follow the instructions of the attacker to send malicious model updates to the server. After receiving the model updates from all selected clients, the server updates the global model according to a certain aggregation rule. This process repeats until the global model converges or the maximum number of training rounds is reached.

B. Threat Model

We define the threat model in terms of the knowledge, capability, and goal of the attacker.

Knowledge. The attacker knows all the information about the compromised clients (e.g., local datasets) but knows nothing about the benign clients. In a certain training round, the attacker is able to know the current global model if any compromised client is selected; otherwise, the attacker does not know the current global model. We assume that the attacker does not know the aggregation rule adopted by the server, which is a challenging but practical setting.

Capability. The attacker can manipulate the model updates of all compromised clients. The attacker cannot interfere with either the benign clients or the server.

Goal. The attacker aims to degrade the accuracy of the global model as much as possible. In particular, we consider an untargeted poisoning attack, which aims to indiscriminately minimize the accuracy of the global model on any test sample.

C. Crafting Model Poisoning Attacks: Problem Formulation

Given the goal of sabotaging the global model, the attacker carefully designs its poisoned model updates in each training round. Existing works on AGR-agnostic model poisoning attacks adopt a variety of objective functions $\mathcal{F}$ and perturbation functions $\mathcal{P}$, as summarized in Table I.

We define a generic AGR-agnostic model poisoning attack as

$$\underset{\gamma,\nabla^p}{\text{argmax}}\, \mathcal{F}(\nabla^m, \nabla_{\{i\in[n']\}}) \tag{1}$$

$$\text{s.t.} \quad \nabla^b = f_{avg}(\nabla_{\{i\in[n']\}}), \tag{2}$$

$$\nabla^p = \mathcal{P}(\nabla^b), \tag{3}$$

$$\nabla^m = \nabla^b + \gamma\nabla^p. \tag{4}$$

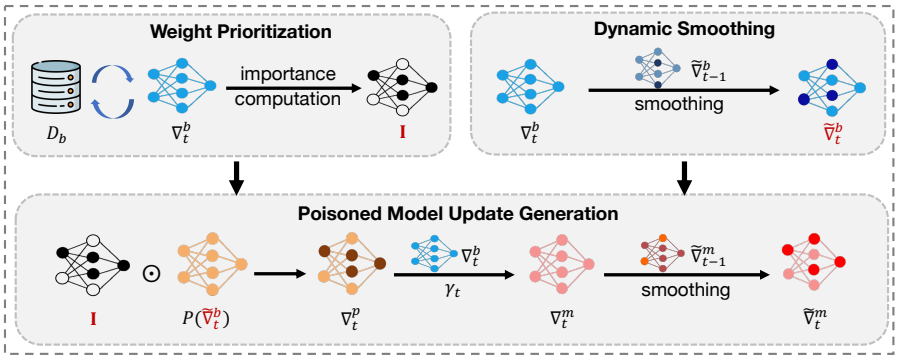

Fig. 2: OBLIVION: an architectural overview.

∇^b is the average aggregate of all selected benign clients in a given round. In fact, the attacker does not know ∇^b as it does not have access to the information of benign clients. We assume that the attacker can estimate ∇^b as it can leverage the local datasets of compromised clients to train the current global model and obtain a proximate ∇^b. In other words, ∇^b is calculated as the average benign model updates $\nabla_{\{i\in[n']\}}$ of compromised clients. ∇^m is the poisoned model update of each compromised client to be uploaded to the server. The perturbation ∇^p is obtained through the function $\mathcal{P}$ based on the benign update ∇^b.

IV. OBLIVION: DETAILED CONSTRUCTION

A. Design Rationale

The key to solving the optimization problem in Eq. (1) is to determine the optimal poisoned model update ∇^m by finding the optimal combination of the perturbation ∇^p and the scaling factor γ. Existing studies solve the optimization problem by first determining the perturbation function $\mathcal{P}$, e.g., the inverse standard deviation $\nabla^{\text{p}}_{\text{std}}$ or the inverse sign of the benign update $\nabla^{\text{p}}_{\text{sgn}}$, and then computing the optimal γ that conforms to the constraint. However, existing methods did not consider the importance of different weights and the influence of the history of model updates.

To enhance the effectiveness of model poisoning attacks, we make full use of the available knowledge of the attacker to refine poisoning perturbations with two specially designed components in OBLIVION, as shown in Fig. 2.

Weight prioritization. According to explanatory analysis on learning models [23], [24], only a fraction of all weights is critical in determining the prediction results, while most of the weights in a large-scale learning model are redundant. Therefore, given the constraint on the perturbations, rather than poisoning all the weights, it is better to focus on perturbing the most important ones. To achieve this goal, we exploit the *catastrophic forgetting* effect [25], [26] in continual learning, which states that the model will easily forget what it has learned in previous rounds, especially if the critical weights are altered. Inspired by this fact, we intentionally incur amnesia of the global model by perturbing the most important weights, which is also the provenance of the name OBLIVION.

Dynamic smoothing. The history of benign model updates in previous rounds will have a continual influence on the global model. Rather than only perturbing the current benign model updates, we come up with a dynamic smoothing strategy that incorporates the history of benign model updates in calculating malicious model updates in the current round.

B. Weight Prioritization

The weight prioritization component in OBLIVION aims to pinpoint the most important weights towards inducing catastrophic forgetting based on the currently received global model θ_G^{t-1}. Elastic Weight Coalescing (EWC) is a typical method that is designed to protect neural networks from catastrophic forgetting. It proposes an efficient way to calculate the importance of weights and avoid catastrophic forgetting by penalizing changes in the weights with large importance scores.

Specifically, EWC deems a weight as critical if the weight plays a significant role in minimizing the loss function during training. Suppose that at the t-th round of training, at least one compromised client is selected by the server to participate in updating the current global model. Let $\mathcal{D}_b$ denote the local benign datasets available to the attacker. Following EWC, we retrain the current global model θ_G^{t-1} with $\mathcal{D}_b$ until convergence. The importance score of weight w in the updated global model θ_G^t is calculated as

$$I_w = \frac{\partial^2 \mathcal{L}(\theta_G^t, \mathcal{D}_b | \theta_G^{t-1})}{\partial w^2}, \tag{5}$$

where $\mathcal{L}$ is the loss function of fine-tuning the current global model θ_G^{t-1} with the local dataset $\mathcal{D}_b$.

Rather than avoiding catastrophic forgetting, our goal is to manipulate model updates to prompt the global model to forget what it has learned from benign updates. Hence, instead of penalizing changes in important weights, we promote forgetting by reinforcing modifications to important weights. To achieve this, we rank all weights in a non-ascending order of their importance scores. Let ϵ denote the fraction of weights to prioritize and $|\theta|$ denote the total number of weights in the global model. We construct a $|\theta|$-dimensional priority vector $\mathbf{I}$. For the top $\epsilon|\theta|$ weights with the highest importance score, we assign a priority of α_1 to the corresponding elements in $\mathbf{I}$.

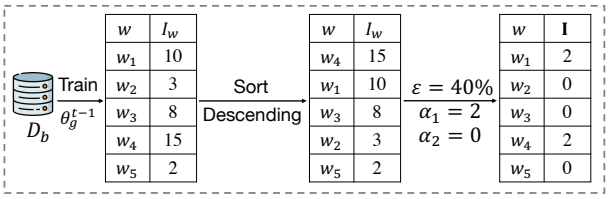

Fig. 3: Toy example of weight prioritization.

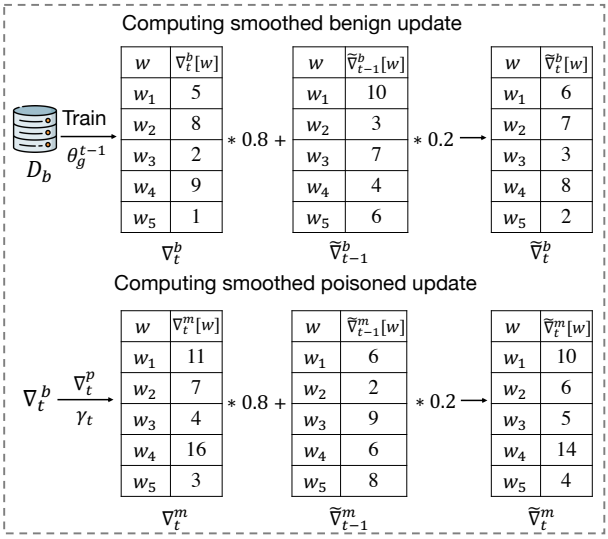

Fig. 4: Toy example of dynamic smoothing.

The remaining elements are assigned a priority of $\alpha_2 < \alpha_1$. After that, when calculating the poisoned model update, the weights with a high priority will be given a larger poisoning perturbation. The detailed process of using the priority vector $\mathbf{I}$ to intensify the poisoning perturbation of important weights is given in Section IV-D.

Toy example. As shown in Fig. 3, given a global model with 5 weights, we fine-tune the global model with a local dataset and then compute the importance score as $I_1 = 10, I_2 = 3, I_3 = 8, I_4 = 15, I_5 = 2$. We aim to select a fraction of $\epsilon = 0.4$ of the weights to prioritize. Therefore, the top $0.4 \times 5 = 2$ weights are given a priority of $\alpha_1 = 2$, and the remaining weights are given a priority of $\alpha_2 = 0$. The priority vector is $\mathbf{I} = [2, 0, 0, 2, 0]$.

C. Dynamic Smoothing

Existing works [12], [16] directly use the current perturbation $\mathcal{P}(\nabla^b)$ as the poisoning perturbation, which ignores the influence of the history of benign updates in previous training rounds. To tackle this problem, we apply a dynamic smoothing algorithm to the current benign update as

$$\widetilde{\nabla}_t^b = \beta_t \nabla_t^b + (1 - \beta_t)\widetilde{\nabla}_{t-1}^b, \tag{6}$$

where ∇_t^b is the benign update of the t-th round, $\widetilde{\nabla}_t^b$ is the smoothed benign update of the t-th round, and β_t is the smoothing factor determined by the number of selected compromised clients in the t-th round.

Algorithm 1: OBLIVION

Input: $\mathcal{P}, \mathcal{F}$ of a specific model poisoning attack, global model θ_G^{t-1}, local dataset $\mathcal{D}_i$, estimated benign update ∇_t^b, prioritization fraction ϵ, priority score α_1 and α_2, smoothing factors β_t and ξ_t, dynamic smoothing threshold κ.

Output: smoothed poisoned model update $\widetilde{\nabla}_t^m$.

1 Train the current global model θ_G^{t-1} on the local datasets $\mathcal{D}_b$ until convergence;
2 Compute the importance score I_w of all weights;
3 Set the prioritization vector $\mathbf{I}$ by giving the top $\epsilon|\theta|$ weights a priority of α_1 and the remaining weights a priority of α_2;
4 Smooth the benign update to obtain $\widetilde{\nabla}_t^b$;
5 Compute the perturbation vector $\nabla_t^p = \mathbf{I} \odot \mathcal{P}(\widetilde{\nabla}_t^b)$;
6 Compute the optimal γ_t based on $\mathcal{F}$, ∇_t^p;
7 Compute the poisoned model update as $\nabla_t^m = \nabla_t^b + \gamma_t \nabla_t^p$;
8 Smooth the poisoned model update to obtain $\widetilde{\nabla}_t^m$.

Moreover, we can also apply the dynamic smoothing algorithm to the entire poisoned model update as

$$\widetilde{\nabla}_t^m = \xi_t \nabla_t^m + (1 - \xi_t)\widetilde{\nabla}_{t-1}^m, \tag{7}$$

where $\widetilde{\nabla}_t^m$ is the smoothed poisoned update of the t-th round and ξ_t is the smoothing factor.

Intuitively, if more compromised clients are selected in the t-th FL round, the model poisoning attack can be more powerful, thus we may choose larger β_t and ξ_t to intensify the effect of the current round of model updates; otherwise, we may choose smaller β_t and ξ_t to be more conservative about the attack. In our experiments, the values of β_t and ξ_t are determined by a threshold κ. If the number of compromised clients selected in the t-th round is greater than κ, we set $\beta_t = \beta_t^h \in (0.5, 1]$ and $\xi_t = \xi_t^h \in (0.5, 1]$, respectively. Otherwise, we set $\beta_t = 1 - \beta_t^h$ and $\xi_t = 1 - \xi_t^h$.

Toy example. As shown in Fig. 4, our dynamic smoothing algorithm is applied to the benign update and the entire poisoned model update. To smooth the benign update, the current benign update ∇_t^b scaled by the smoothing factor $\beta_t = 0.8$ is summed with the smoothed benign update of the $(t-1)$-th federated learning round $\widetilde{\nabla}_{t-1}^b$ scaled by $(1 - \beta_t) = 0.2$ to obtain the smoothed benign update of the current round $\widetilde{\nabla}_t^b = [6, 7, 3, 8, 2]$. The smoothed poisoned model update $\widetilde{\nabla}_t^m = [10, 6, 5, 14, 4]$ can be computed in a similar way.

D. OBLIVION: An Integrated Framework

To prompt the global model to forget what it has learned from benign updates, our intuition here is to reinforce the modifications to important weights. Recall that the priority vector $\mathbf{I}$ is computed in the weight prioritization component, where important weights are assigned a larger priority than unimportant weights. By using $\mathbf{I}$ to adjust the primitive perturbation vector $\mathcal{P}(\widetilde{\nabla}_t^b)$, we can perturb ∇_t^b in a smarter way.

With $\mathbf{I}$ and $\mathcal{P}(\widetilde{\nabla}_t^b)$, the perturbation vector ∇_t^p is computed as

$$\nabla_t^p = \mathbf{I} \odot \mathcal{P}(\widetilde{\nabla}_t^b), \tag{8}$$

where $\odot$ is the Hadamard product.

We summarize the integrated framework of OBLIVION in Algorithm 1. Given the perturbation function $\mathcal{P}$ and the objective function $\mathcal{F}$ of a specific model poisoning attack (e.g., Min-Max), OBLIVION equips the attack with the weight prioritization and the dynamic smoothing components to intensify the model poisoning effect. As shown in Algorithm 1, in line 1∼3, we calculate the importance score of each weight given the current global model. In line 4∼5, we compute the forgetting-augmented perturbation ∇_t^p based on the smoothed benign update $\widetilde{\nabla}_t^b$ and the prioritization vector $\mathbf{I}$. Note that in this way the perturbation budget of the primitive perturbation vector $\mathcal{P}(\widetilde{\nabla}_t^b)$ will be reallocated by $\mathbf{I}$ to amplify the catastrophic forgetting phenomenon. In line 6, we find the optimal scaling factor γ_t by using the corresponding objective function $\mathcal{F}$ and the calculated perturbation vector ∇_t^p. In line 7∼8, we further smooth the poisoned model update and send the result $\widetilde{\nabla}_t^m$ to the server.

V. EXPERIMENTAL EVALUATION

A. Experimental Setup

Datasets and models. We consider three datasets: `FEMNIST` [27], `CIFAR-10` [28], and `Purchase` [29]. All datasets are partitioned in a non-i.i.d. manner to simulate the production federated learning practice. To make a fair evaluation, we follow the experiment settings adopted in the existing literature [20], [21].

- ▷ `FEMNIST` is a grayscale image dataset built for the character classification task with 62 classes, 3,550 users, and 805,263 images of size 28×28 pixels. Each of the 3,550 users has personal data of handwritten digits or letters. We consider a total of 3,400 clients participating in federated learning and assign the first 3,400 pre-divided user data to these clients. LeNet5 [30] is used for the global model.
- ▷ `CIFAR-10` is a 10-class *class-balanced* image dataset with 60,000 images of size 32×32 pixels (50,000 for training and 10,000 for testing). We consider 500 clients participating in federated learning and divide the 50,000 training samples using the Dirichlet distribution [31] with concentration parameter $\alpha = 1$ among the clients. VGG-11 [32] is used for the global model architecture.
- ▷ `Purchase` is a *class-imbalanced* dataset built for the customer classification task with 100 classes and 197,324 binary feature vectors. The feature vector of each sample has a length of 600. The first 180,000 samples are used for training and the remaining are used for testing. We consider 1,000 clients participating in federated learning. We divide the 180,000 training data samples using the Dirichlet distribution with $\alpha = 1$ among the clients. A fully-connected network with a layer size of {600, 1024, 100} is used for training the global model.

TABLE II: Parameter settings.

Datasets	Baseline	ϵ	α_1	α_2	$\beta_1 = \xi_1$	κ	$\mathcal{P}$
`FEMNIST`	LIE	10%	2	0	0.7	6	∇_{std}^p
	Min-Max	30%	2	0	0.7	6	∇_{sgn}^p
	Min-Sum	30%	1.5	0.5	0.7	6	∇_{uv}^p
`CIFAR-10`	LIE	10%	2	0	0.7	6	∇_{std}^p
	Min-Max	10%	2	0	0.7	6	∇_{std}^p
	Min-Sum	10%	2	0	0.7	6	∇_{std}^p
`Purchase`	LIE	10%	2	0	0.7	6	∇_{std}^p
	Min-Max	10%	2	0	0.7	6	∇_{uv}^p
	Min-Sum	10%	2	0	0.7	6	∇_{uv}^p

Aggregation rule. Aggregation rules are designed to defend against poisoning attacks in federated learning. We evaluate the performance of OBLIVION under one non-defensive aggregation rule, FEDAVG (the most popular and effective aggregation rule) and four popular defensive aggregation rules, namely MULTI-KRUM, TRIMMED-MEAN, MEDIAN, and AFA. Suppose that at most m of a total of n clients are compromised by the attacker.

- ▷ MULTI-KRUM [4] is an advanced version of KRUM [4], which selects the model update closest to its $n - m - 2$ neighboring model updates in terms of Euclidean distance as the global model. In MULTI-KRUM, the server maintains a selection set (initialized to the set of model updates received in the current round of iteration) and a remaining set (initialized to empty). The server selects a model update using KRUM from the remaining set and adds it into the selection set. This process repeats until the selection set includes b model updates such that $n - b > 2m + 2$. The final aggregate is the average of the model updates in the selection set.
- ▷ TRIMMED-MEAN [5] aggregates each dimension of model updates separately. For each dimension, it first sorts the value of the dimension of the received model updates and then removes p largest and p smallest values in each dimension. The average of the rest values is set as the aggregate of this dimension.
- ▷ MEDIAN [5] also aggregates each dimension of model updates separately. Different from TRIMMED-MEAN, MEDIAN directly sets the average value of the model updates in one dimension as its aggregate.
- ▷ AFA [8] utilizes the difference of cosine similarity between malicious and benign updates to remove the malicious updates. AFA computes the ranges in terms of the mean, median, and standard deviation of cosine similarity and discards the out-of-range updates.

Federated learning framework. We implement and evaluate a fully-functional prototype of OBLIVION on PLATO[1], an open-source real-world framework for scalable federated learning. To the best of our knowledge, we are the first to implement model poisoning attacks in real-world federated learning scenarios. The same federated learning setting is used for all experiments. Specifically, the maximum number of

[1]Online at: https://github.com/TL-System/plato

TABLE III: Attack performance of OBLIVION. ⋆ denotes the highest accuracy of the global model in a non-convergence case.

Datasets	Attack Methods	Non-Robust AGR	Robust AGR				
		FEDAVG	TRIMMED-MEAN	MULTI-KRUM	AFA	MEDIAN	Average I_o
Purchase	LIE	3.5%⋆	39.3%⋆	60.2%	55.7%	29.8%	**6.2%**
	OBLIVION-LIE	**2.4%**⋆	**30.6%**⋆	**58.9%**	**48.4%**	**22.2%**⋆	
	Min-Max	2.6%⋆	36.2%	59.8%	62.2%	26.2%	**5.3%**
	OBLIVION-Min-Max	**2.4%**⋆	**29.5%**	**58.0%**	**53.6%**	**22.3%**	
	Min-Sum	3.8%⋆	37.7%	51.4%	59.2%	30.0%	**7.7%**
	OBLIVION-Min-Sum	**2.4%**⋆	**33.2%**	**39.3%**	**53.3%**	**21.9%**	
FEMNIST	LIE	2.6%⋆	29.0%	69.0%	7.0%⋆	49.5%⋆	**19.5%**
	OBLIVION-LIE	**2.4%**⋆	**19.0%**⋆	**29.8%**⋆	**5.4%**⋆	**22.5%**⋆	
	Min-Max	2.6%⋆	55.4%	67.8%	64.2%	51.4%	**6.3%**
	OBLIVION-Min-Max	**2.4%**⋆	**53.2%**	**55.4%**	**56.9%**	**48.1%**	
	Min-Sum	2.6%⋆	59.6%	68.2%	62.4%	51.8%	**6.2%**
	OBLIVION-Min-Sum	**2.4%**⋆	**56.5%**	**62.2%**	**55.6%**	**43.1%**	
CIFAR-10	LIE	11.6%⋆	57.8%	61.9%	59.3%	54.5%	**3.5%**
	OBLIVION-LIE	**17.9%**⋆	**56.0%**	**58.6%**	**52.9%**	**51.9%**	
	Min-Max	16.5%⋆	58.2%	59.2%	63.9%	52.5%	**8.1%**
	OBLIVION-Min-Max	**12.0%**⋆	**55.8%**	**59.0%**	**35.3%**	**51.5%**	
	Min-Sum	16.8%⋆	59.7%	62.8%	63.9%	51.2%	**3.7%**
	OBLIVION-Min-Sum	**18.6%**⋆	**56.7%**	**60.2%**	**58.9%**	**47.2%**	

training rounds is set as 200. In each round of training, the server randomly selects 30 clients to participate in the training. For local training by each client, we use a batch size of 10, a local training epoch number of 1, and the SGD optimizer with a learning rate of 0.05. Unless specified otherwise, we assume that the attacker compromises 20% of the clients.

OBLIVION settings. We apply the weight prioritization and the dynamic smoothing components of OBLIVION to existing AGR-agnostic model poisoning attacks to form their enhanced version. The parameters of OBLIVION include the percentage of important parameters ϵ, the priority parameters α_1 and α_2, the threshold κ, the smoothing factors β and ξ, and the perturbation function $\mathcal{P}$. In the experiments, we set the values of β and ξ to be the same. We integrate OBLIVION with three AGR-agnostic model poisoning attacks to obtain the enhanced version. Unless specified otherwise, the default parameter settings of OBLIVION are shown in Table II. Recall that to get the priority vector $\mathbf{I}$ in OBLIVION, we need to use $\mathcal{D}_b$ (the local benign datasets of clients available to the attacker) to train the global model until convergence, which will take a relatively long time if the dataset is large. In the experiments, we utilize the local datasets of all compromised clients to compute the priority vector $\mathbf{I}$ in the initial round and use the computed $\mathbf{I}$ in subsequent training.

Evaluation metrics. For a given federated learning setting, we evaluate the accuracy of the global model under the existing attack (denoted as A_g) and under the OBLIVION-enhanced attack (denoted as A_g^o). We find in our experiments that the global model may fail to converge in the face of powerful poisoning attacks. In this case, the accuracy of the global model first increases and then keeps decreasing. We report the highest accuracy achieved during the process as A_g and A_g^o under these circumstances. We define *attack improvement*, $I_o = A_g - A_g^o$, as the reduction in the accuracy of the global model due to the introduction of OBLIVION.

B. Overall Attack Performance

To evaluate the performance of OBLIVION, we first integrate OBLIVION with three state-of-the-art AGR-agnostic model poisoning attacks (LIE, Min-Max, and Min-Sum) and compare the performance of the three attacks with and without OBLIVION. Table III shows that OBLIVION can enhance existing AGR-agnostic model poisoning attacks under any combination of aggregation rules and datasets. The results followed by ⋆ indicate that the global model does not converge.

For Purchase, we can observe that OBLIVION enhances the attack effect of Min-Sum by an average 7.7% reduction in the accuracy of the global model. It is obvious that such a drop in global model accuracy can impose huge losses on federated learning, especially when the dataset is large. Similarly, OBLIVION also boosts the attack effect of Min-Max and Min-Sum against various aggregation rules. For CIFAR-10, we can observe that OBLIVION dramatically reduces the accuracy of the global model under AFA by 28.6% compared to the original Min-Max. For the given four defensive aggregation rules, OBLIVION-AFA achieves an average attack improvement of 19.5%. Similar effects can be seen in FEMNIST. The introduction of OBLIVION to LIE can even change the global model from convergence to non-convergence under MULTI-KRUM and TRIMMED-MEAN, and increase the attack performance by 39.2% and 10.0% respectively in regard of the highest global model accuracy, which further demonstrates the effectiveness of OBLIVION.

C. Ablation Study

OBLIVION consists of two key components, i.e., *weight prioritization* and *dynamic smoothing*. To investigate the contribution of weight prioritization and dynamic smoothing components to the attack performance of OBLIVION, we conduct ablation studies to test the contribution of the two components to the attack performance.

TABLE IV: Ablation study of the weight prioritization and the dynamic smoothing components. The dataset is Purchase and results report attack improvement I_o. * denotes the highest achieved accuracy of the global model in a non-convergence case.

Baseline	Components	TRIMMED-MEAN	MULTI-KRUM	AFA	MEDIAN
LIE	weight prioritization	4.1%*	0.3%	6.3%	4.4%*
	dynamic smoothing	3.4%*	22.1%	6.2%	5.8%*
	weight prioritization & dynamic smoothing	8.7%*	1.3%	7.3%	7.6%*
Min-Max	weight prioritization	2.0%	0.8%	7.8%	3.0%
	dynamic smoothing	0.6%	0.4%	7.8%	2.1%
	weight prioritization & dynamic smoothing	6.7%	1.8%	8.6%	3.9%
Min-Sum	weight prioritization	4.9%	0.6%	4.3%	6.7%
	dynamic smoothing	0.6%	11.2%	3.4%	7.4%
	weight prioritization & dynamic smoothing	4.5%	12.1%	5.9%	8.1%

The attacks with weight prioritization or dynamic smoothing can be seen as two variants of the OBLIVION by fine-tuning the default parameter settings. For OBLIVION that only activates the weight prioritization component, denoted as OBLIVION-P, we set $\kappa = 0$ and $\beta = \xi = 1$. For OBLIVION that only activates the dynamic smoothing component, denoted as OBLIVION-S, we set $\alpha_1 = \alpha_2 = 1$ to stop the forgetting augmentation with the priority vector $\mathbf{I}$. We use OBLIVION to denote the integrated attack with both weight prioritization and dynamic smoothing components.

Table IV reports the attack improvement I_o. It can be seen that both OBLIVION-P and OBLIVION-S promote the performance of AGR-agnostic attacks, since the attacks, after adding either weight prioritization or dynamic smoothing component, outperform the original attacks. We can also observe that, in most cases, OBLIVION with both weight prioritization and dynamic smoothing components leads to a stronger attack than merely using one of them. For instance, when facing TRIMMED-MEAN, OBLIVION enhances Min-Max by reducing the accuracy of the global model by 4.7% and 6.1% compared to OBLIVION-P and OBLIVION-S, respectively.

We also compute the expectation and standard deviation of the attack improvement I_o for OBLIVION-P, OBLIVION-S, and OBLIVION to provide more insight into the ablation study results. The results show that OBLIVION has the largest expectation of $E(I_o) = 6.4\%$. OBLIVION-S has a slightly lower expectation than OBLIVION with $E(I_o) = 5.9\%$, and OBLIVION-P has the lowest expectation $E(I_o) = 3.8\%$. When it comes to the standard deviation, OBLIVION-S has the most fluctuating performance with $Std(I_o) = 6.1\%$, while OBLIVION-P shows the most stable attack enhancement with $Std(I_o) = 2.5\%$, and OBLIVION has a similar stability with $Std(I_o) = 3.1\%$. From the analysis of the statistics, we can know that using dynamic smoothing brings a higher reward compared to weight prioritization, but the gains come at a price of instability. Weight prioritization is the most steady method to improve attack performance. OBLIVION combines the advantages of weight prioritization and dynamic smoothing to find a balance between attack performance and generality.

D. Robustness

We now evaluate the impact of parameters on the attack performance of OBLIVION in terms of the proportion of compromised clients, the fraction of perturbed weights, and the smoothing factor. The threshold κ is set as the expectation of compromised clients selected in each round.

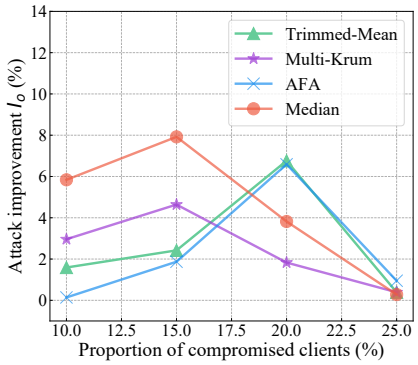

Fig. 5: Effect of compromised clients proportion on attack improvement I_o.

Effect of the proportion of compromised clients. Fig. 5 shows the attack improvement I_o by adding OBLIVION to Min-Max attack when the proportion of compromised clients varies from 10% to 25% for Purchase. We note that OBLIVION-Min-Max outperforms Min-Max for all the combinations of defensive aggregation rules and the proportion of compromised clients. Meanwhile, we can observe that as the proportion of compromised clients increases, the attack improvement I_o of OBLIVION with respect to Min-Max under all defensive aggregation rules first rises and then decreases. This is because our dynamic smoothing component is less effective in improving attack performance when many compromised clients participate in training. Recall that for a given federated learning round, the attacker uses the average aggregate of benign updates to approximate the average aggregate ∇_b of all selected benign clients. With more compromised clients, the attacker can better approximate the average benign update ∇_b, resulting in a more aggressive poisoned model update ∇_m. When the proportion of compromised clients is low, the dynamic smoothing component in OBLIVION can use the history benign model updates to help the attacker better approach ∇_b. When the proportion of compromised clients exceeds

TABLE V: The impact of the fraction of important weights ϵ on the attack performance of OBLIVION. The baseline is Min-Max. The dataset is Purchase. * denotes the highest achieved accuracy of the global model in a non-convergence case.

Attack Methods	Non-Robust AGR	Robust AGR			
	FEDAVG	TRIMMED-MEAN	MULTI-KRUM	AFA	MEDIAN
Baseline	2.6%*	36.2%	59.8%	62.2%	26.2%
$\epsilon = 0.1$	2.4%*	29.5%	58.0%	53.6%	22.3%
$\epsilon = 0.3$	2.3%*	30.3%	50.1%	49.6%	24.7%
$\epsilon = 0.5$	2.3%*	37.2%	52.3%	56.0%	26.0%

TABLE VI: Impact of smoothing factors β and ξ on the attack performance of OBLIVION. The baseline is Min-Max. The dataset is Purchase. * denotes the highest achieved accuracy of the global model in a non-convergence case.

Attack Methods	Non-Robust AGR	Robust AGR			
	FEDAVG	TRIMMED-MEAN	MULTI-KRUM	AFA	MEDIAN
Baseline	2.6%*	36.2%	59.8%	62.2%	26.2%
$\xi = \beta = 0.9$	2.4%*	31.5%	58.3%	56.0%	28.5%
$\xi = \beta = 0.7$	2.4%*	29.5%	58.0%	53.6%	22.3%
$\xi = \beta = 0.5$	2.6%*	32.1%	59.6%	59.9%	25.6%

a threshold, the role of historical information in improving attack performance is gradually weakening. The experimental results indicate that OBLIVION has a good improvement effect in attack performance when the proportion of compromised clients is relatively low, which is the most practical setting in production federated learning [22].

Impact of the fraction of important weights. We then investigate the impact of the value of ϵ on the performance of OBLIVION. Recall that ϵ is the fraction of the weights to prioritize. The larger ϵ is, the more weights are identified as important and given intensified perturbation. Then the resulting poisoned model update has a stronger attack capability, thereby weakening the dilution of the poisoning effect by updates from benign clients. However, as ϵ continues to increase, the more likely that the poisoned update will be identified as anomalous and be removed, which will lead to a decrease in attack performance. Hence, there is a sweet spot between perturbation strength and attack performance.

As shown in Table V, with the increase of ϵ, the accuracy of the global model under the defense of MULTI-KRUM and AFA first decreases and then increases. We can also observe that the accuracy of the global model under the defense of TRIMMED-MEAN and MEDIAN keeps increasing when the value of ϵ increases from 0.1 to 0.5. This is because the sweet spot of TRIMMED-MEAN and MEDIAN in terms of ϵ is smaller than 0.1. It indicates that TRIMMED-MEAN and MEDIAN are more sensitive to ϵ compared to MULTI-KRUM and AFA. It is known that TRIMMED-MEAN and MEDIAN are dimension-wise aggregation rules that aggregate each dimension of model updates separately. They can better detect OBLIVION since OBLIVION employs dimension-wise perturbation reinforcement. Thus, TRIMMED-MEAN and MEDIAN have better attack performance when ϵ is relatively small.

Impact of smoothing factors. We also evaluate the impact of β and ξ on the attack performance of OBLIVION. In the dynamic smoothing algorithm we designed, the larger the values of β and ξ, the greater the influence of the model update calculated in the current round on the final poisoned model update. Table VI depicts the change in the global model accuracy when varying the defense methods and the smoothing factor values. The results show that, with the increase of β, the accuracy of the global model under all defensive aggregation rules first decreases and then rises. This is because when the values of β and ξ are set too large, the attacker will focus too much on the current round of learning without using the historical information sensibly, thereby limiting the performance of model poisoning attacks. Conversely, if the values of β and ξ are set too small, the attack will lose too much information learned in the current round of training, resulting in the degradation of the attack performance. In Table VI, we can also observe that although the attack performance of OBLIVION-Min-Max varies with the values of β and ξ, it consistently outperforms the original Min-Max attack. This further demonstrates the effectiveness of OBLIVION.

Discussions. Our experimental results verify that OBLIVION can effectively enhance the poisoning effect of existing AGR-agnostic model poisoning attacks in federated learning. In this work, we consider a strong threat model that the attacker does not know the aggregation rule adopted by the server. Thus, we are more concerned about generalizing OBLIVION to enhance the poisoning effect under different unknown aggregation rules (transferability); i.e., the average attack improvement over all defensive aggregation rules. Under a weaker threat model, if the attacker knows the specific aggregation rule adopted by the server, we can launch a stronger version of OBLIVION by fine-tuning the parameters against the target aggregation rule.

VI. CONCLUDING REMARKS

In this paper, we present the motivation, design, and evaluation of OBLIVION, an AGR-agnostic model poisoning attack that enhances the attack performance in federated learning under defensive aggregation rules. OBLIVION intensifies the poisoning effect with weight prioritization and dynamic smoothing components. The former concentrates the poisoning perturbations on the most important weights, and the latter takes into account the influence of history benign updates. Extensive experimental results confirm the effectiveness of OBLIVION for corrupting the global model.

The promising results of OBLIVION may spur future research in various aspects. First, more powerful poisoning attacks may be designed based on new explanation theories that reveal the fundamental vulnerability of learning models. Second, to defend against model poisoning attacks in federated learning, more robust aggregation rules should be proposed to better differentiate anomalous model updates even if the attacker adjusts its strategies to evade being detected. Finally, other forms of attacks under the federated learning settings, e.g., membership inference attacks, may be further enhanced with similar strategies.

REFERENCES

[1] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[2] M. Paulik, M. Seigel, H. Mason, D. Telaar, J. Kluivers, R. van Dalen, C. W. Lau, L. Carlson, F. Granqvist, C. Vandevelde *et al.*, "Federated evaluation and tuning for on-device personalization: System design & applications," *arXiv preprint arXiv:2102.08503*, 2021.

[3] Facebook, Online at https://github.com/facebookresearch/flsim.

[4] 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.

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

[6] R. Guerraoui, S. Rouault *et al.*, "The hidden vulnerability of distributed learning in byzantium," in *Proc. International Conference on Machine Learning*. PMLR, 2018, pp. 3521–3530.

[7] 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.

[8] L. Muñoz-González, K. T. Co, and E. C. Lupu, "Byzantine-robust federated machine learning through adaptive model averaging," *arXiv preprint arXiv:1909.05125*, 2019.

[9] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "FLTrust: learning via trust bootstrapping," *arXiv preprint arXiv:2012.13995*, 2020.

[10] M. Fang, X. Cao, J. Jia, and N. Z. Gong, "Local model poisoning attacks to Byzantine-robust federated learning," in *Proc. 29th USENIX Security Symposium (USENIX Security)*, 2020.

[11] G. Baruch, M. Baruch, and Y. Goldberg, "A little is enough: Circumventing defenses for distributed learning," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[12] V. Shejwalkar and A. Houmansadr, "Manipulating the Byzantine: Optimizing model poisoning attacks and defenses for federated learning," in *Proc. 28th Annual Network and Distributed System Security Symposium (NDSS)*, 2021.

[13] Z. Li and D. Hoiem, "Learning without forgetting," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 12, pp. 2935–2947, 2017.

[14] R. Aljundi, F. Babiloni, M. Elhoseiny, M. Rohrbach, and T. Tuytelaars, "Memory aware synapses: Learning what (not) to forget," in *Proc. the European Conference on Computer Vision (ECCV)*, 2018.

[15] A. Aich, "Elastic weight consolidation (EWC): Nuts and bolts," *arXiv preprint arXiv:2105.04093*, 2021.

[16] G. Baruch, M. Baruch, and Y. Goldberg, "A little is enough: Circumventing defenses for distributed learning," in *Proc. Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

[17] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *Proc. International Conference on Artificial Intelligence and Statistics*, 2020.

[18] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Analyzing federated learning through an adversarial lens," in *Proc. International Conference on Machine Learning*, 2019, pp. 634–643.

[19] V. Tolpegin, S. Truex, M. E. Gursoy, and L. Liu, "Data poisoning attacks against federated learning systems," in *Proc. European Symposium on Research in Computer Security*. Springer, 2020.

[20] M. Fang, X. Cao, J. Jia, and N. Gong, "Local model poisoning attacks to Byzantine-robust federated learning," in *Proc. 29th USENIX Security Symposium (USENIX Security 20)*, 2020.

[21] V. Shejwalkar and A. Houmansadr, "Manipulating the Byzantine: Optimizing model poisoning attacks and defenses for federated learning," in *Proc. 28th Annual Network and Distributed System Security Symposium (NDSS)*, 2021.

[22] V. Shejwalkar, A. Houmansadr, P. Kairouz, and D. Ramage, "Back to the drawing board: A critical evaluation of poisoning attacks on federated learning," *arXiv preprint arXiv:2108.10241*, 2021.

[23] D. Bau, J.-Y. Zhu, H. Strobelt, A. Lapedriza, B. Zhou, and A. Torralba, "Understanding the role of individual units in a deep neural network," *Proceedings of the National Academy of Sciences*, vol. 117, no. 48, pp. 30 071–30 078, 2020.

[24] W. Samek, G. Montavon, S. Lapuschkin, C. J. Anders, and K.-R. Müller, "Explaining deep neural networks and beyond: A review of methods and applications," *Proceedings of the IEEE*, vol. 109, no. 3, pp. 247–278, 2021.

[25] R. M. French, "Catastrophic forgetting in connectionist networks," *Trends in cognitive sciences*, vol. 3, no. 4, pp. 128–135, 1999.

[26] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, "Continual lifelong learning with neural networks: A review," *Neural Networks*, vol. 113, pp. 54–71, 2019.

[27] S. Caldas, S. M. K. Duddu, P. Wu, T. Li, J. Konečnỳ, H. B. McMahan, V. Smith, and A. Talwalkar, "LEAF: A benchmark for federated settings," *arXiv preprint arXiv:1812.01097*, 2018.

[28] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[29] Acquire Valued Shoppers Challenge at Kaggle, Online at https://www.kaggle.com/c/acquire-valued-shoppers-challenge/data, 2019.

[30] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[31] T. Minka, "Estimating a dirichlet distribution," 2000.

[32] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.