Manipulating the Byzantine: Optimizing Model Poisoning Attacks and Defenses for Federated Learning

Virat Shejwalkar
University of Massachusetts Amherst
vshejwalkar@cs.umass.edu

Amir Houmansadr
University of Massachusetts Amherst
amir@cs.umass.edu

***Abstract*—Federated learning (FL) enables many data owners (e.g., mobile devices) to train a joint ML model (e.g., a next-word prediction classifier) without the need of sharing their private training data. However, FL is known to be susceptible to model poisoning attacks by malicious participants (e.g., adversary-owned mobile devices), who aim at hampering the accuracy of the jointly trained model through sending malicious inputs during the federated training process. In this paper, we present a general framework for model poisoning attacks on FL. We show that our framework leads to poisoning attacks that substantially outperform the state-of-the-art model poisoning attacks by large margins. For instance, our attacks result in $1.5\times$ to $60\times$ more reductions in the accuracy of FL compared to the strongest of existing poisoning attacks.**

Our work demonstrates that existing Byzantine-robust FL algorithms are significantly more susceptible to model poisoning than previously thought. Motivated by this, we design a defense against poisoning of FL, called *divide-and-conquer* (DnC). We demonstrate that DnC outperforms all existing Byzantine-robust FL algorithms in defeating model poisoning attacks, specifically, it is $2.5\times$ to $12\times$ more resilient in our experiments with different datasets and models[1].

I. Introduction

Federated learning (FL) is an emerging learning paradigm in which many data owners (called *clients*) collaborate in training a common machine learning model, without sharing their private training data. In this setting, a *central server* (e.g., a service provider) repeatedly collects some updates that the clients compute using their local private data, aggregates the clients' updates using an aggregation algorithm (AGR), and finally uses the aggregated client updates to tune the jointly trained model (called *global model*), which is broadcasted to all of the clients at the end of each FL training epoch.

While FL has emerged as a promising solution for many in-the-wild learning settings that involve mutually untrusting clients, FL mechanisms are prone to *poisoning attacks* [31]: *malicious* clients can attempt to degrade the utility (e.g., model accuracy) of the resulting global model by contributing malicious model updates during the FL training process. A poisoning attack can be either *untargeted* [31], [4], [17], [37], where the goal is to minimize the accuracy of the global model on *any* test input, or *targeted* [6], [3], where the goal is to minimize the accuracy on *specific* test inputs. To defeat poisoning attacks, a recent line of work has investigated the design of Byzantine-robust FL algorithms, where the central server uses some *robust aggregation algorithm* (AGR) [39], [8], [37], [31], [2], [12] to reduce the impact of malicious model updates while preserving model utility.

Note that while centralized (non-FL) models have long been known to be vulnerable to poisoning attacks through manipulation of training data (i.e., *data poisoning attacks*) [20], [7], recent works [17], [6], [3], [4], [31] have demonstrated advanced poisoning attacks tailored to FL, where malicious clients directly manipulate the model updates (e.g., gradients) that they share with the central server during FL training process. Such attacks are known as *model poisoning attacks* and are shown [4], [17] to be substantially more impactful on FL than the traditional data poisoning attacks. This paper studies untargeted model poisoning attacks on FL.

Our contribution: A general framework for model poisoning attacks on FL. In the first part of this work, we present a general framework for model poisoning on FL. Unlike previous works [17], [4], [31], [37], we consider a comprehensive set of possible threat models for model poisoning attacks along two dimensions of the adversary's knowledge: the knowledge of the updates shared by benign clients, and the knowledge of the AGR algorithm that the server uses. We demonstrate that the model poisoning attacks launched using our framework outperform state-of-the-art model poisoning attacks in defeating *all* Byzantine-robust FL algorithms.

The high-level approach of our attack is as follows. The adversary computes a benign reference aggregate using some benign updates she knows; then she computes a malicious perturbation (whose generation will be explained in detail), e.g., a unit vector in the opposite direction of the benign aggregate. Finally, the adversary computes her malicious model update by maximally perturbing the benign reference aggregate in the malicious direction, while also evading detection by robust aggregation algorithms. Based on this intuition, we provide a general optimization framework (Section IV-A) to mount optimal model poisoning attacks on different FL settings.

[1]This work was supported in part by NSF grant CPS-1739462.

Network and Distributed Systems Security (NDSS) Symposium 2021
21-24 February 2021
ISBN 1-891562-66-5
https://dx.doi.org/10.14722/ndss.2021.23xxx
www.ndss-symposium.org

However, due to its computational intractability, we introduce a novel modification to our general optimization problem: We fix the type of malicious perturbation, and search for the optimal coefficient of the perturbation vector. We give an algorithm to find the most effective coefficient for any model poisoning attack objective that is formulated using our general optimization framework (Section IV-D). We also show that selecting the appropriate malicious perturbation can significantly boost the impact of model poisoning on FL, and propose a simple yet effective approach to select the most effective perturbation vector for a given FL setting (Section VI-C).

We perform various optimizations to tailor our attacks to specific threat models. Specifically, we present attacks that are tailored to state-of-the-art AGR algorithms [8], [31], [39], which we call *AGR-tailored* attacks. Our AGR-tailored attacks (Section IV-B) aim to maximize the perturbation to a reference benign update, while also evading the detection by robust AGRs. We note that this also solves the fundamental objective of maximizing the distance between the aggregates with and without the attacks.

We also present *AGR-agnostic* attacks that work for adversaries with no knowledge of the underlying AGR algorithm (Section IV-C). However, the constraints of the server's AGR are unknown to our AGR-agnostic adversaries. We circumvent this challenge based on the key intuition behind robust AGRs: a robust AGR labels an update malicious if it wonders far away from a set of benign updates. Therefore, our AGR-agnostic attacks constrain their search of the most malicious updates to a ball of a small radius around the clique of the benign updates. Based on this intuition, we propose two novel AGR-agnostic attacks, both of which outperform state-of-the-art AGR-agnostic attack, i.e., LIE [4].

Evaluations. We have extensively evaluated our attacks using four benchmark classification datasets (Section VI). We show that the impact of our poisoning attacks (measured by the reduction in model accuracy) is significantly higher than that of state-of-the-art model poisoning attacks, i.e., Fang [17] and LIE [4], for all of the datasets. For instance, for CIFAR10 with Alexnet, without any knowledge of the updates of benign clients, the accuracy reductions due to Fang and LIE attacks are 11.8% and 30.0%, respectively, while the reductions due to our AGR-tailored and AGR-agnostic attacks are 45.6% and 44.5%, respectively, i.e., $\mathbf{1.5}\times$ and $\mathbf{4}\times$ **more accuracy reduction**. For Purchase, our AGR-tailored and AGR-agnostic attacks on Krum AGR [8] incur $\mathbf{15}\times$ and $\mathbf{60}\times$ **more accuracy reduction**. For FEMNIST dataset with Trimmed-mean (Median) AGR, the accuracy reductions due to our AGR-tailored and AGR-agnostic attacks are $\mathbf{2}\times$ to $\mathbf{3}\times$ ($\mathbf{4}\times$ to $\mathbf{18}\times$) more than the accuracy reductions due to Fang and LIE attacks.

Our contribution: Defending model poisoning on FL. In the second part of this work, we propose a novel robust aggregation algorithm (Section VII) to defend against model poisoning attacks. Our robust AGR, called the *divide-and-conquer* (DnC), is motivated from existing defenses against data poisoning attacks in centralized learning settings; these defenses use spectral analysis [16], [15], [28], [26], e.g., singular value decomposition, to detect and filter the outliers in poisoned data. The intuition behind DnC is that, a malicious model poisoning update is impactful if and only if it deviates significantly from the benign updates along a certain malicious direction in the updates' space. Therefore, DnC first computes the principle component, i.e., the direction of maximum variance, of the set of its input updates. Then it computes scalar products of the updates with the principal component, called *projections*. Finally, DnC removes a constant fraction of the total updates that have the largest projections. However, it is computationally impossible to perform spectral analysis of extremely high dimensional model updates in FL. Therefore, DnC performs dimensionality reduction that enables spectral analysis of input updates and also ensures the effective detection of malicious updates.

Our DnC AGR provides strong theoretical robustness guarantees of the removal of malicious updates, when benign updates are independently and identically distributed (iid) [16], [15], [28], [26], [36]. Furthermore, in order to provide empirical evidence of the robustness, we design an adaptive attack against DnC using our general framework (Section IV-A).

Evaluations. We evaluate DnC for four benchmark classification datasets. We show that for the three iid datasets, i.e., MNIST, CIFAR10, and Purchase, DnC significantly reduces the impacts of model poisoning attacks on FL when compared to previous defenses (Section VII-C1). For instance, compared to the existing robust AGRs, DnC reduces the maximum reduction in the accuracy of the global model due to model poisoning attacks from 36.8% to 6.1% for CIFAR10 with Alexnet, from 32.5% to 6.3% for CIFAR10 with VGG11, from 11.6% to 1.8% for Purchase, and from 4.4% to 1.9% for MNIST. Note that, in all of these cases, the most impactful attack against DnC is our adaptive attack on DnC, as expected. We also show the superiority of DnC in defeating model poisoning on FL is due to its effective filtering of the malicious updates with high poisoning impacts. For the non-iid FEMNIST dataset, we show that, compared to existing robust AGRs, DnC mitigates the model poisoning attacks substantially more effectively when the adversary has no knowledge of the benign update of benign clients.

II. Background

A. Federated Learning

We consider a standard federated learning (FL) [30], [21], [22] setting with a server and n clients with possibly disjoint private datasets drawn from a data distribution D; the datasets may not be independently and identically distributed (iid). In epoch t, the server selects a subset of clients and broadcasts the current global model parameters θ^t to the chosen clients. All the chosen clients then compute *stochastic gradients*, $\nabla_{t,i} = \frac{\partial L(b,\theta^t)}{\partial \theta^t}$, using a randomly sampled minibatch b of their private data and *synchronously* send it to the server. Here, $L(b,\theta^t)$ is the loss, e.g., cross-entropy loss, computed using minibatch b and model parameters θ^t. Then, the server aggregates the collected gradients using some aggregation algorithm, $f_{\mathsf{agr}}(\nabla_{\{t,i\in[n]\}})$, e.g., dimension-wise Average. Finally, the server computes a new model θ^{t+1} using the aggregate and SGD, and broadcasts it to a new subset of randomly selected clients. This process is repeated until the global model converges, i.e., has low loss $L(D_{\mathsf{val}},\theta)$ on validation data D_{val}. The FL output is the global model with the maximum accuracy on D_{val} across all FL epochs.

FL can either be *cross-device FL* with a very large (millions) number of clients and only a subset of them is chosen in an epoch, or *cross-silo FL* with a moderate number of clients (tens to hundreds) and all of them participate in every epoch. Unlike previous works which consider only cross-silo FL [4], [17], [37], *we consider both the FL settings*.

B. Poisoning attacks on FL

Federated learning is known to be vulnerable to various poisoning attacks [8], [4], [6], [3], [31], [17], [29], [38], [32], [20]. We divide these attacks based on the adversary's goal and capabilities. Based on the goal of adversary, there can be two types of attacks: *untargeted* and *targeted* attacks. In untargeted poisoning attacks, the goal is to minimize the accuracy of the global model on *any* test input [17], [4], [31], [29], [38]. In targeted poisoning attacks, the goal is to minimize the accuracy on *specific* test inputs, while maintaining high accuracies on the rest of the test inputs [6], [3]. *Backdoor* attacks [3] are a subset of targeted attacks, where the targeted test inputs have a *backdoor trigger*. Untargeted attacks can completely cripple the global model, and therefore we believe, pose a more severe threat to FL.

Based on the adversary's capabilities, there are two types of attacks: *model* and *data* poisoning attacks. In model poisoning attacks [17], [4], [31], [38], [6], [3], the adversary *can directly manipulate* the gradients on malicious devices before sharing them with the server in each epoch. While in data poisoning attacks [32], [20], the adversary *can only indirectly manipulate* the gradients on malicious devices *by poisoning training datasets* on the devices. Due to the direct manipulation of gradients, model poisoning attacks can achieve higher attack impacts on FL. Also, as data poisoning attacks cannot compute arbitrary gradients, it is not trivial to imitate the malicious gradients of model poisoning attacks via training data poisoning. Therefore, in order to understand the severity of the threat of poisoning to FL, *we focus on the stronger untargeted model poisoning attacks on FL.*

C. Existing robust aggregation algorithms for FL

In non-adversarial FL settings, dimension-wise Average [14], [23], [30] is an effective *aggregation algorithm (AGR)* to aggregate clients' gradients. However, even a single malicious client can manipulate the Average AGR based FL [8], [3], [5]. Therefore, multiple *Byzantine-robust* AGRs for FL [8], [31], [37], [39], [2], [19], [10] are proposed to defend against poisoning attacks by malicious clients.

1) Krum: Blanchard et al. [8] propose Krum AGR based on the intuition that the malicious gradients need to be far from the benign gradients in order to poison the global model. Hence, Krum selects the gradient from the set of its input gradients that is closest to its $n - m - 2$ neighboring gradients in the squared Euclidean norm space; here, m is an upper bound on the number malicious clients in FL.

2) Multi-krum: Blanchard et al. [8] modify Krum AGR to Multi-krum AGR in order to effectively utilize the knowledge shared by the clients in each FL epoch [8]. Multi-Krum selects a gradient using Krum from a remaining-set (initialized to the set of all the received gradients), adds it to a selection-set (initialized to an empty set), and removes it from the remaining-set. This way, Multi-krum selects c gradients such that $n - c > 2m + 2$. Finally, Multi-krum averages the gradients in the selection-set. Multi-krum significantly outperforms Krum in terms of the global model accuracy.

3) Bulyan: Mhamdi et al. [31] show that a malicious gradient can remain close to benign gradients while having a *single* gradient dimension with a very large value (on order of $\Omega(\sqrt[p]{d})$) and prevent convergence of the global model. As a remedy, they propose Bulyan AGR, which requires $n \geq 4m+3$ for its robustness guarantees to hold. Bulyan first selects $\theta(\theta \leq n - 2m)$ gradients in the same fashion as Multi-krum, and then computes Trimmed-mean of the selected gradients; please refer to [31] for more details.

4) Trimmed-mean: Trimmed-mean [39], [37] is a coordinate-wise AGR which aggregates each dimension of input gradients separately. Specifically, for a given dimension j, it sorts the values of j^{th}-dimension of all gradients, i.e., sorts $\nabla^j_{\{i \in [n]\}}$. Then it removes β largest and smallest values and computes average of the rest of the values as its aggregate of dimension j. We use Trimmed-mean where β equals m, the number of malicious clients. Yin et al. [39] show that Trimmed-mean achieves order-optimal error rates when $m \leq \beta \leq \frac{n}{2}$ for strongly convex objective function.

5) Median: Median [39], [37] is an another coordinate-wise AGR which aggregates its input gradients by computing median of the values of each of the dimensions of the gradients. Yin et al. [39] give theoretical guarantees on the robustness of Median AGR, while Fang et al. [17] empirically show that Median AGR has better robustness than the more sophisticated Krum AGR.

6) Adaptive federated average (AFA): AFA [33] removes malicious gradients based on their cosine-similarities with a benign gradient. More specifically, in each FL round, AFA computes a weighted average of collected gradients and computes cosine similarities between the weighted average and each of the collected gradients. Then, AFA discards the gradients whose similarities are out of a range; this range is a simple function of mean, median and standard deviation of the similarities.

7) Fang defenses: Fang et al. [17] propose defenses that are meta-AGRs and rely on existing robust AGRs to detect malicious gradients. More specifically, consider a robust AGR A. Given a set of gradients G, the corresponding Fang defense, called *Fang-A*, computes a score for each gradient in $\nabla_i \in G$ as follows. Fang-A computes two aggregates using A, one with ∇_i and one without ∇_i in G, i.e., $A(G)$ and $A(G - \nabla_i)$. Fang-A then computes losses and/or errors of the models obtained by updating the global model using the two aggregates. Then Fang-A assigns a score to each ∇_i such that the lower the negative impact of ∇_i on the loss and/or error of the corresponding model, the higher the score. Finally, Fang-A discards the gradients with the lowest scores. For any given AGR, [17] proposes three defenses, one based on the loss of model, one based on error of model, and one based on both the loss and error. The combination of loss and error works strictly better than either loss or error, hence, we consider the defense based on the combination of loss and error.

D. Comparison with related works

Comparing our attacks. We compare our attacks with two state-of-the-art model poisoning attacks, Fang [17] and LIE (short for 'A little is enough') [4]. Fang attacks require the knowledge of the server's AGR, and without the knowledge, they have no noticeable impact. Fang attack [17] formulates a general optimization problem, but tailors it only to Krum AGR. Furthermore, their solution of the optimization, even for Krum AGR, is far from optimal (Figure 1-(a)). Next, unlike our attacks on Trimmed-mean and Median AGRs, [17] proposes only a heuristic attack to manipulate each dimension of the collected gradients. Consequently, there are huge differences in the impacts of ours and Fang attacks. Furthermore, our attacks choose the most effective malicious direction tailored to the given FL setting, which boosts their poisoning impacts. Note that, Fang attacks are effective for the synthetic non-iid datasets generated in [17]. But, as we will show, for iid and highly imbalanced non-iid datasets, Fang attacks perform poorly.

LIE attacks [4] do not require the knowledge of AGR. Hence, in order to circumvent detection by any AGR, LIE attacks add a very small amounts of noises to a benign aggregate. Comparatively, our novel AGR-agnostic attacks leverage the knowledge of benign gradients in more principled ways to boost their poisoning impacts (Figures 1-(b, c)).

Comparing our defense. Our attacks show that the convergence guarantees of previous robust AGRs [8], [39] are insufficient to make FL robust to model poisoning. Hence, our robust AGR aims to provide guarantees of the removal of malicious updates. On a high level, our *divide-and-conquer* (DnC) AGR uses singular value decomposition, a radically different approach from the previous AGRs (Section II-C). Due to its effective filtering guarantees, the global model of DnC converges to a better local optima and achieves significantly higher accuracies even under attack (Table IV).

III. Threat Model of Model Poisoning Attacks

Here, we discuss various possible threat models of model poisoning attacks on FL.

Adversary's Objective. The goal of the adversary is to craft malicious gradients such that when the malicious clients share the malicious gradients with the central server, the accuracy of the resulting global model reduces indiscriminately, i.e., on any test input. This is also known as *untargeted model poisoning attack*.

Adversary's Capabilities. We assume that the adversary controls up to m out of n total clients, called *malicious clients*. We assume that the number of malicious clients is less than the number of benign clients, i.e., $(m/n) < 0.5$; otherwise, no Byzantine-robust AGR will be able to defeat poisoning attacks. Following the previous works [6], [4], [3], [17], [37], [19], we assume that the adversary can access the global model parameters broadcast in each epoch and can directly manipulate the gradients on malicious devices.

Adversary's Knowledge. We consider two important dimensions of FL setting: knowledge of the gradient updates (simply gradients) shared by the benign devices and knowledge of the AGR algorithm of the server. More specifically, we consider four adversaries as shown in Table I. `agr-updates` adversary is the strongest adversary who knows both the gradients of benign devices and the server's AGR. Although `agr-updates` adversary has limited practical significance, it has been commonly used in previous works [17], [37], [4] to understand the severity of the model poisoning threat. Furthermore, it allows the service provider (the server in this case) to evaluate the robustness of its AGR algorithms. `agr-only` adversary knows the server's AGR, but does not have the gradients of benign devices. To compute malicious gradients, `agr-only` adversary uses benign gradients computed using the benign data on malicious devices. `updates-only` adversary has the gradients of benign devices, but does not know the server's AGR. We consider this adversary in order to demonstrate *the empirical upper bound* of the severity of our AGR-agnostic attacks. Finally, the `agnostic` adversary does not have the gradients on benign devices or the server's AGR, and is *the weakest possible adversary in FL*.

Table I: Knowledge based classification of model poisoning adversaries in federated learning.

Type	Gradients of benign devices	Server's AGR algorithm
`agr-updates`	✓	✓
`agr-only`	✗	✓
`updates-only`	✓	✗
`agnostic`	✗	✗

Note that, none of the state-of-the-art untargeted model poisoning attacks thoroughly consider these two dimensions: Fang attacks [17] assume the complete knowledge of the server's AGR algorithm, while LIE attacks [4] assumes the complete knowledge of the gradients of benign devices.

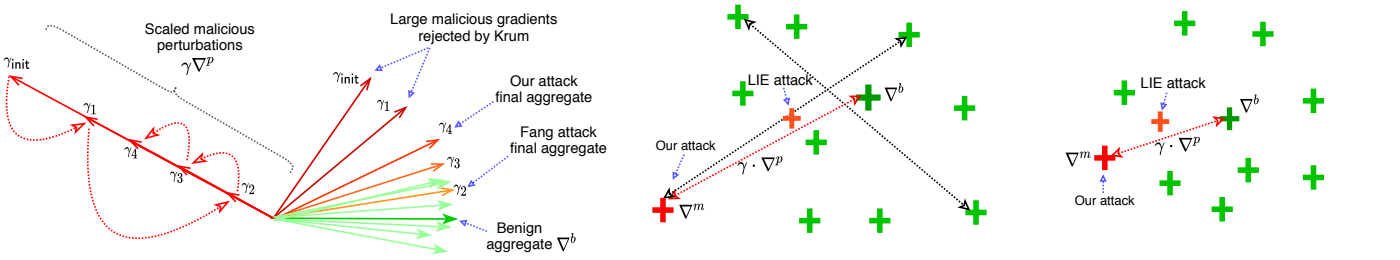
IV. Our General Framework for Model Poisoning

In this section, we describe our general framework to mount model poisoning attacks on FL, followed by specific optimizations for different AGRs and threat models, and finally give an algorithm to solve the optimizations.

A. General optimization formulation

In each FL training epoch, the malicious and benign clients share malicious and benign gradients, respectively, and then the server updates the global model using an aggregate of all of the gradients. To successfully mount an untargeted attack, our general optimization problem aims to maximize the damage to the global model in each FL epoch.

In order to maximize the damage to the global model, we craft the malicious gradients, denoted by $\nabla^m_{\{i\in[m]\}}$, such that the aggregate computed by the server is far from a *reference benign aggregate*, denoted by ∇^b. A possible ∇^b is the average of the benign gradients that the adversary knows. For instance, the `agr-only` adversary can compute m benign gradients using the benign data on malicious devices. The final malicious gradient ∇^m is a perturbed version of the benign aggregate ∇^b, i.e., $\nabla^m = \nabla^b + \gamma\nabla^p$, where ∇^p is a *perturbation vector* and γ is a *scaling coefficient*. Therefore, the objective of the full



(a) Our AGR-tailored attack (demonstrated for Krum) (b) Our AGR-agnostic Min-Max attack (c) Our AGR-agnostic Min-Sum attack

Figure 1: **Schematics of our attacks:** (a) Our AGR-tailored attack, unlike Fang attack, fine tunes the malicious gradient ($\nabla^b + \gamma\nabla^p$), using optimal γ and dataset-optimized ∇^p. (b) Our AGR-agnostic Min-Max attack finds its malicious gradient ∇^m (red cross) whose maximum distance from any other gradient is less than the maximum distance between any two benign updates (black arrows). (c) Our AGR-agnostic Min-Sum attack finds ∇^m (red cross) whose sum of distances from the other updates is less than the sum of distances of any benign gradient from the other benign updates. Due to stricter constraints, ∇^m of Min-Sum attack is closer to the benign aggregate, ∇^b, than ∇^m of Min-Max attack. LIE attack computes very suboptimal ∇^m due to extremely small amounts of noise additions.

knowledge `agr-updates` adversary is given by (1).

$$\underset{\gamma,\nabla^p}{\text{argmax}} \quad \|\nabla^b - f_{\text{agr}}(\nabla^m_{\{i\in[m]\}} \cup \nabla_{\{i\in[m+1,n]\}})\|_2 \qquad (1)$$
$$\nabla^m_{i\in[m]} = \nabla^b + \gamma\nabla^p; \quad \nabla^b = f_{\text{avg}}(\nabla_{\{i\in[n]\}})$$

where $\nabla_{\{i\in[n]\}}$ are the benign gradients that the adversary knows. Note that state-of-the-art robust AGRs [8], [39], [31] are generally not differentiable. Hence, solving (1), i.e., finding the optimal γ and ∇^p, using gradient descent based optimizations is not trivial. Our idea to overcome this challenge is to fix the perturbation vector ∇^p and find the optimal γ, i.e., solve the modified objective in (2). Algorithm 1 (Section IV-D) describes our algorithm to optimize γ.

$$\underset{\gamma}{\text{argmax}} \quad \|\nabla^b - f_{\text{agr}}(\nabla^m_{\{i\in[m]\}} \cup \nabla_{\{i\in[m+1,n]\}})\|_2 \qquad (2)$$
$$\nabla^m_{i\in[m]} = \nabla^b + \gamma\nabla^p; \quad \nabla^b = f_{\text{avg}}(\nabla_{\{i\in[n]\}})$$

Introducing perturbation vectors. A perturbation vector is any malicious direction in the space of gradients that the adversary can use to perturb ∇^b and find the malicious gradients $\nabla^m_{\{i\in[m]\}}$. In this work, we experiment with the following three types of ∇^p's.

Inverse unit vector (∇^p_{uv}). The intuition here is to compute the malicious gradient by perturbing ∇^b by a scaled unit vector that points in the opposite direction of ∇^b. Hence, we compute ∇^p_{uv} as $-\frac{\nabla^b}{\|\nabla^b\|_2}$.

Inverse standard deviation (∇^p_{std}). The intuition here is that the higher the variance of a dimension of benign gradients, the higher the perturbation that the adversary can introduce along the dimension. Hence, we compute ∇^p_{std} as $-\text{std}(\nabla_{i\in[n]})$.

Inverse sign (∇^p_{sgn}). We compute ∇^p_{sgn} as $-\text{sign}(f_{\text{avg}}(\nabla_{i\in[n]}))$. The intuition here is similar to that of (∇^p_{uv}), but we observe that (∇^p_{sgn}) is more effective for some classification tasks, e.g., MNIST.

As we will show in Section VI-C, the appropriate choice of perturbation vector ∇^p is the key to an effective attack. For instance, for Krum AGR, the attack using ∇^p_{uv} increases the accuracy of global model of MNIST, while the attack using ∇^p_{uv} reduces the accuracy to random guessing for Purchase. Finally, we note that our experiments show that our attacks destroy the global model accuracy and significantly outperform the existing model poisoning attacks using one of these ∇^p's. Hence, we leave investigating the optimal ∇^p to future work.

B. AGR-tailored attacks

In this section, we consider `agr-updates` and `agr-only` adversaries, who know the server's AGR algorithm and tailor the general attack objective in (2) to the known AGR. We consider the seven robust AGRs described in Section II-C. For the clarity of presentation, we provide the AGR-tailored optimizations for `agr-updates` adversary with all the benign gradients $\nabla_{\{i\in[n]\}}$. The only change in optimizations for `agr-only` adversary is to compute ∇^b using the benign gradients computed using the benign data of the m malicious devices, i.e., $\nabla_{\{i\in[m]\}}$.

1) Krum: Krum[2] selects a single gradient from its inputs as its aggregate. Hence, a successful attack requires Krum to select one of its malicious gradients, i.e., $\nabla^m_{i\in[m]} = f_{\text{krum}}(\nabla^m_{\{i\in[m]\}} \cup \nabla_{\{i\in[m+1,n]\}})$. Therefore, we modify (2) to (3) for Krum. For each of the input gradients, Krum computes a score that is the sum of distances of $n-m-2$ nearest neighbors of the gradient. Therefore, to maximize the chances of Krum selecting a malicious gradient, we keep all the malicious gradients the same.

$$\underset{\gamma}{\text{argmax}} \quad \nabla^m_{i\in[m]} = f_{\text{krum}}(\nabla^m_{\{i\in[m]\}} \cup \nabla_{\{i\in[m+1,n]\}}) \qquad (3)$$
$$\nabla^m_{i\in[m]} = f_{\text{avg}}(\nabla_{\{i\in[n]\}}) + \gamma\nabla^p$$

2) Multi-krum: Multi-krum uses Krum iteratively to construct a selection set $\mathcal{S}$ and computes average of the gradients in the selection set as its aggregate. Our attack on Multi-krum ensures that all of the malicious gradients are in selected $\mathcal{S}$, while maximizing the perturbation $\gamma\nabla^p$ used to compute the malicious gradients. This strategy minimizes the number of benign gradients in $\mathcal{S}$, while maximizing $\gamma\nabla^p$ increases the poisoning impact of malicious gradients on the final aggregate. Therefore, we modify (2) to (4) for Multi-krum; here $|A|$ is

[2]We omit suffix AGR, when it is clear from the context.

the cardinality of A.

$$\underset{\gamma}{\text{argmax}} \quad m = |\{\nabla \in \nabla^m_{\{i\in[m]\}} | \nabla \in \mathcal{S}\}| \qquad (4)$$
$$\nabla^m_{i\in[m]} = f_{\mathsf{avg}}(\nabla_{\{i\in[n]\}}) + \gamma\nabla^{\mathsf{p}}$$

3) Bulyan: Our attack on Bulyan is similar to that on Multi-krum, because Bulyan also computes a selection set in the exact same fashion as Multi-krum. Furthermore, as the distribution of perturbation $\gamma\nabla^{\mathsf{p}}$, and therefore, that of our malicious gradients, is very similar to the distribution of the benign updates. Hence, the Trimmed-mean based filtering in the second stage of Bulyan cannot effectively remove the contribution of our malicious gradients, which makes our attack effective.

4) Adaptive federated averaging: Our attack on AFA is similar to that on Multi-krum, because similar to Multi-krum, AFA computes a selection set and then computes their weighted average. In AFA, the weight of each gradient increases or remains constant if the gradient is selected, and decreases if it is discarded. Hence, our attack on AFA aims to maximize the number of malicious gradients in the final selection set of AFA.

The only change required to tailor our attack to AFA is to use f_{afa} to compute the selection set, $\mathcal{S}$, in the attack formulation in (4). Note that, to use f_{afa} to compute $\mathcal{S}$, we need the weights of clients in each epoch; in our experiments, we simply assume that all the clients have the same weights in each epoch, while computing malicious gradients. Even this loose assumption leads to highly impactful attacks. But, note that if the exact weights are available, the attack impact can improve further.

5) Trimmed-mean: For Trimmed-mean, we directly solve the optimization described by (2), by fixing the perturbation ∇^{p} and keeping all the malicious updates the same. Hence, our objective is to maximize the L_2-norm of the distance between the reference benign update ∇^b and the aggregate computed using Trimmed-mean on the set of benign and malicious updates. This is formalized in (5).

$$\underset{\gamma}{\text{argmax}} \quad \|\nabla^b - f_{\mathsf{trmean}}(\nabla^m_{\{i\in[m]\}} \cup \nabla_{\{i\in[m+1,n]\}})\|_2 \qquad (5)$$
$$\nabla^m_{i\in[m]} = f_{\mathsf{avg}}(\nabla_{\{i\in[n]\}}) + \gamma\nabla^{\mathsf{p}}$$

Note that in (5), we aim to compute γ that maximizes the required L_2-norm distance. As we demonstrate in our evaluations, this extremely simple approach of crafting malicious updates outperforms the complex approaches proposed by Fang [17] attacks by very large margins for all the datasets.

6) Median: Similar to Trimmed-mean, Median computes the aggregate of the collected updates for each dimension. Therefore, our attack on Median is similar to that on Trimmed-mean. The only change we introduce in (5) is that, for Median, our optimization aims to maximize $\|\nabla^b - f_{\mathsf{median}}(\nabla^m_{\{i\in[m]\}} \cup \nabla_{\{i\in[m+1,n]\}})\|_2$.

7) Fang defenses: As described in Section II-C7, Fang defenses are meta-AGRs that rely on existing defenses to detect and remove malicious gradients. We argue that the simple attack where the malicious gradients are computed by adding an arbitrarily large vector to the average of benign gradients suffices to completely cripple any of the Fang defenses. We illustrate this below.

Consider the Fang defense based on Multi-krum, called *Fang-Mkrum*, and a set of b benign and m malicious gradients, which we denote by G. For any malicious gradient ∇_m computed using our attack, Fang-Mkrum computes two aggregates: First, $f_{\mathsf{mkrum}}(G)$, which is the average of b benign gradients; this is because f_{mkrum} can detect and remove all the m malicious gradients due to their extremely large distances from the rest of the gradients. Second, $f_{\mathsf{mkrum}}(G - \nabla_m)$, which is the average of $b-1$ benign gradients, because f_{mkrum} removes all $m-1$ malicious gradients and a single benign gradient. For any benign gradient ∇_b, Fang-Mkrum computes two aggregates $f_{\mathsf{mkrum}}(G)$ and $f_{\mathsf{mkrum}}(G - \nabla_b)$. Similar to malicious gradients, these aggregates are averages of b and $b-1$ benign gradients.

Hence, in theory, when adversary mounts our attack, Fang-Mkrum assigns almost equal scores to all the benign and our malicious gradients. This forces Fang-Mkrum to accept at least few of the malicious gradients. As our malicious gradients are arbitrarily large, even when Fang-Mkrum selects a few of them, they significantly corrupt the global model. Observe that all the Fang defenses exhibit the same behavior, and hence, are fundamentally broken.

For brevity, in this work, we only consider the Fang defense that uses Trimmed-mean to discard malicious gradients. We note that, our AGR-tailored attack on Fang defenses does not depend on the robust AGR it uses.

C. AGR-agnostic attacks

Now, we consider the AGR-agnostic adversaries, `updates-only` and `agnostic`, who do not know the server's AGR algorithm. This is an important practical consideration, because the FL platforms can conceal the details and/or parameters of their robust AGRs to protect the security of the proprietary global models. Below, we first provide intuition behind our attacks and then propose two AGR-agnostics attacks to craft malicious gradients.

Intuition. All the robust AGRs for FL tend to remove/attenuate malicious gradients based on one or more of the following criteria: 1) distances from the benign gradients [8], [5], [31], [2], [39], 2) distributional differences with the benign gradients [5], [35], 3) difference in L_p-norms of the benign and malicious gradients [35]. Figures 1-(b, c) visualize the intuition behind our attacks based on the above criteria. The intuition is as follows. The distance based defenses work by removing the gradients that lie outside of the clique formed by the benign gradients. Therefore, our attacks maximize the distance of malicious gradient from a reference benign gradient, while ensuring that the malicious gradients lie within the clique of benign gradients. This also ensures that L_p-norms of the malicious and benign gradients are similar. To ensure distributional similarity, we use perturbations $\gamma\nabla^p$ with the similar distributions as the benign gradients.

Next, we present optimization for two novel AGR-agnostic attacks based on the intuition. We present the optimizations for `updates-only` adversary, who has all the benign gradients $\nabla_{\{i\in[n]\}}$. The extension to `agnostic` adversary is similar to

that explained at the beginning of Section IV-B for `agr-only` adversary.

Attack-1 (Min-Max): Minimize maximum distance attack. Our first attack ensures that the malicious gradients lie close to the clique of the benign gradients (Figure 1-(b)). Hence, we compute the malicious gradient such that its maximum distance from any other gradient is upper bounded by the maximum distance between any two benign gradients. (6) formalizes the corresponding optimization. Note that in order to maximize the impact of our attack, we keep all the malicious gradients the same. Hence, we formulate our attack objective in (6) for a single malicious gradient.

$$\underset{\gamma}{\text{argmax}} \quad \max_{i\in[n]} \|\nabla^m - \nabla_i\|_2 \leq \max_{i,j\in[n]} \|\nabla_i - \nabla_j\|_2 \qquad (6)$$
$$\nabla^m = f_{\mathsf{avg}}(\nabla_{\{i\in[n]\}}) + \gamma\nabla^p$$

Attack-2 (Min-Sum): Minimize sum of distances attack. Our second AGR-agnostic Min-Sum attack ensures that the sum of squared distances of the malicious gradient from *all* the benign gradients is upper bounded by the sum of squared distances of any benign gradient from the other benign gradients (Figure 1-(c)). (7) formalizes the corresponding optimization. We keep all malicious gradients the same for maximum attack impact. Hence, we formulate our objective in (7) for a single malicious gradient.

$$\underset{\gamma}{\text{argmax}} \quad \sum_{i\in[n]} \|\nabla^m - \nabla_i\|_2^2 \leq \max_{i\in[n]} \sum_{j\in[n]} \|\nabla_i - \nabla_j\|_2^2 \qquad (7)$$
$$\nabla^m = f_{\mathsf{avg}}(\nabla_{\{i\in[n]\}}) + \gamma\nabla^p$$

D. Solving for the most effective scaling factor γ

In previous sections, we formulated optimizations for various adversarial settings such that the final objective is to search for the optimal scaling coefficient, γ. Algorithm 1 describes our algorithm to optimize γ for any of the optimizations.

For clarity of presentation of Algorithm 1, we assume an oracle $\mathcal{O}$ that takes the set of benign gradients, $\nabla_{\{i\in[n]\}}$ and γ as inputs. Then, $\mathcal{O}$ computes malicious gradients as $\nabla^m_{\{i\in[m]\}} = \nabla^b + \gamma\nabla^p$, and outputs True if they satisfy the adversarial objective, otherwise outputs False. For instance, for our AGR-tailored attack on Krum, $\mathcal{O}$ outputs True if a malicious gradient is selected by f_{krum}, i.e., if (3) is satisfied. For our Min-Max attack (Section IV-C), $\mathcal{O}$ outputs True if the maximum distance of malicious gradient from any benign gradient is lower than the maximum distance between any two benign gradients, i.e., if (6) is satisfied.

Now, we describe Algorithm 1. The core idea of our algorithm is as follows: We start with a large γ value. We reduce γ in steps of size step until $\mathcal{O}$ returns True, e.g., for Krum, we reduce γ until a malicious gradient is selected by f_{krum}, i.e., (3) is satisfied for the first time. Our final γ is always *greater* than this *minimum* γ value that satisfies the objective. We halve the step size each time we update γ in order to make the search finer. From the minimum γ value, we increase γ using updated step sizes step, until $\mathcal{O}$ returns False, i.e., for Krum, we increase γ until f_{krum} *does not* select any malicious gradient, i.e., (3) is no more satisfied. Our final γ is always *lower* than this *maximum* γ value that satisfies the objective. Then we modify γ repeatedly and oscillate between the minimum and maximum γ values until the change in γ is below a threshold τ.

Algorithm 1 Algorithm to optimize γ

Input: γ_{init}, τ, $\mathcal{O}$, $\nabla_{\{i\in[n]\}}$
step $\leftarrow \gamma_{\mathsf{init}}/2$, $\gamma \leftarrow \gamma_{\mathsf{init}}$
while $|\gamma_{\mathsf{succ}} - \gamma| > \tau$ **do**
  if $\mathcal{O}(\nabla_{\{i\in[n]\}}, \gamma) ==$ True **then**
    $\gamma_{\mathsf{succ}} \leftarrow \gamma$
    $\gamma \leftarrow (\gamma + \mathsf{step}/2)$
  else
    $\gamma \leftarrow (\gamma - \mathsf{step}/2)$
  end if
  step $=$ step$/2$
end while
Output γ_{succ}

V. Experimental Setup

A. Datasets and model architectures

CIFAR10 [24] is a 10-class *class-balanced* classification task with 60,000 RGB images, each of size 32 × 32. 'Class-balanced' datasets have the same number of samples per class, e.g., each class of CIFAR10 has 6,000 images. We use 50 clients each with 1,000 samples and use validation and test data of sizes 5,000 each. We use Alexnet [25] and VGG11 [34] as the global model architectures.

MNIST [27] is a 10-class class-balanced classification task with 70,000 grayscale images, each of size 28 × 28. We use 100 clients each with 600 samples and use validation and test data of sizes 5,000 each. For MNIST, we use a fully connected network (FC) with layer sizes {784, 512, 10} as the global model architecture.

Purchase [1] is a 100-class class-imbalanced classification task with 197,324 binary feature vectors, each of length 600. We use 80 clients each with 2,000 training samples and use validation and test data of sizes 5,000 each. We use a fully connected network with layer sizes {600, 1024, 100}.

FEMNIST [9], [13] is a character recognition classification task with 3,400 clients, 62 classes, and a total of 671,585 grayscale images. Each of the 3,400 clients has her own data made of her own handwritten digits or letters (62 classes: 52 for upper and lower case letters and 10 for digits). The mean and standard deviation of the number of samples per client are 226.83 and 88.94, respectively. In each FL epoch, we randomly select 60 out of 3400 clients for FL training. FEMNIST is a non-iid, class-imbalanced dataset commonly encountered in cross-device FL settings [21], while the previous datasets are more common in cross-silo FL settings.

B. Learning and attacks settings

We train CIFAR10 with Alexnet using batch size of 250 and SGD optimizer with learning rates of 0.5 from epochs 0-1000 and 0.05 from 1000-1200. We train CIFAR10 with VGG11 using batch size of 200 and SGD optimizer with learning rates of 0.1 from epochs 0-1000 and 0.01 from 1000-1200. We train MNIST for 500 epochs using Adam optimizer

with 0.001 learning rate and batch size of 100. We train Purchase for 1000 epochs using SGD with learning rate of 0.5 and batch size of 500. We train FEMNIST for 1500 epochs using Adam optimizer with learning rate of 0.001 and use client's entire data in a each batch.

Unless specified otherwise, we assume 20% malicious clients for all adversarial settings, e.g., 20 malicious clients for MNIST. For most of our evaluation, we use independently and identically distributed (iid) CIFAR10, MNIST, and Purchase datasets, because poisoning FL with iid data is the hardest [17].

Measurement metrics. For a given FL setting, A_θ denotes the accuracy of the best globel model, over all the FL training epochs, in the benign setting without any attack, while A^*_θ denotes the accuracy under the given attack. We define *attack impact*, I_θ, as *the reduction in the accuracy of the global model due to the attack*, hence for a given attack, $I_\theta = A_\theta - A^*_\theta$.

C. Baseline model poisoning attacks

Below, we detail here two state-of-the-art model poisoning attacks, LIE [4] and Fang [17], that we compare against.

LIE: The LIE attack [4] adds small amounts of noises to each dimension of the average of the benign gradients. The small noises sufficiently large to adversely impact the global model and sufficiently small to evade detection by the Byzantine-robust AGRs. Specifically, the adversary computes the average μ and standard deviation σ of the benign gradients she has, computes a coefficient z based on the total number of benign and malicious clients, and finally computes the malicious update as $\mu + z\sigma$.

Fang: The Fang attack [17] is an optimization based model poisoning attack tailored to Krum AGR. The adversary computes the average μ of the benign gradients she has, computes a perturbation $\nabla^p = -\text{sign}(\mu)$, and finally computes a malicious update as $\nabla^m = (\nabla^b + \gamma \cdot \nabla^p)$ by solving for the coefficient γ. The attack starts from a reference γ and keeps halving it until Krum select the resulting ∇^m, therefore unlike our attacks, Fang attack does not optimize γ (Figure 1-(a)).

VI. Evaluation of Our Attacks

A. Comparison with the state-of-the-art attacks

In this section, we compare our attacks with state-of-the-art model poisoning attacks, Fang [17] and LIE[3] [4], for all the adversaries from Table I. The results are given in Table II; 'No attack' column shows accuracy A_θ of the global model in the benign setting, while the rest of the columns show the 'attack impact' I_θ, as defined in Section V-B.

For a fair comparison, we compare the attacks that use the knowledge of AGR, i.e., our AGR-tailored and Fang attacks under `agr-updates` and `agr-only` adversaries. We separately compare the attacks that do not use the knowledge of AGR, i.e., our AGR-agnostic and LIE attacks under `updates-only` and `agnostic` adversaries.

[3]We omit the suffix 'attack' when it is clear from the context.

1) Comparing AGR-tailored attacks: Table II shows that, ***our AGR-tailored attacks outperform Fang attacks for all the combinations of threat model, AGR, dataset, and model architecture by large margins***. For CIFAR10 with `agr-updates` adversary, our attacks are $2\times$ more impactful than Fang. While with `agr-only` adversary, our attacks are $2.5\times$ and $4.5\times$ more impactful than Fang for Alexnet and VGG11 models, respectively. For the rest of the AGRs, our attacks are $3\times$ to $7\times$ ($2\times$ to $4\times$) more impactful than Fang attacks on CIFAR10 with Alexnet (VGG11) for both `agr-updates` and `agr-only` adversaries.

Under `agr-updates` (`agr-only`) adversary, Fang and our attacks on Krum with MNIST have impacts of 20.5% (17.4%) and 33.9% (24.1%), respectively, i.e., our attack is $1.7\times$ ($1.5\times$) more effective than Fang. The impact of Fang attack on Trimmed-mean (Median) with MNIST is just 1.2% (1.7%), while that of our attack is 11.0% (4.4%), i.e., our attack is $10\times$ ($2.5\times$) more impactful. Even for AFA, which is the empirically most robust AGR for MNIST, our attack is $3\times$ more impactful than Fang.

For Purchase, our attacks reduce the accuracy of Krum to the random guessing, i.e., close to 1% for all the adversaries and, except for AFA, our attacks are at least $10\times$ more impactful than Fang attacks. Similarly, with `agr-updates` adversary, impacts of Fang attacks on Trimmed-mean and Median are 1.8% and 0.2%, respectively, while impacts of our attacks are 23.4% and 11.0%. We note similarly higher impacts of our attacks with `agr-only` adversary. For Multi-krum, Bulyan, and AFA, our attacks are $2\times$ more impactful than Fang attacks. For Fang-Trmean, the Fang defense that uses Trimmed-Mean to discard malicious gradients, our attacks reduce the global model accuracy to random guessing for all combinations of datasets and models; this is expected as discussed in Section II-C7.

For FEMNIST the impacts of our attacks with `agr-updates` adversary on AFA, Multi-krum, Trimmed-mean, and Median are respectively $12\times$, $2\times$, $3\times$, and $15\times$, that of Fang attack. For Krum and Bulyan also, the impacts of our attacks are moderately higher than that of Fang attacks.

Why our attacks are superior? For Krum AGR, although ours and Fang attacks have similar attack objectives, they differ in two instrumental aspects: First, instead of generalizing a single perturbation type across all datasets, our attacks tailor the perturbation to the given dataset (as we will explain in Section VI-C). Next, as Figure 1-(a) demonstrates, our Algorithm 1 carefully fine tunes γ of our objective (3), while Fang attack simply finds the first γ that satisfies its objective. Our attacks on AFA, Bulyan, and Multi-krum AGRs are also carefully tailored to the AGRs, while Fang attack uses the same objective as Krum for these AGRs.

Fang proposes the same attack for Trimmed-mean and Median AGRs, which crafts the values of each dimension of malicious gradients using the available benign gradients. But our attacks have more tailored and impactful objectives of diverging the final aggregate as far away from a benign aggregate as possible using the most malicious perturbation direction.

2) Comparing AGR-agnostic attacks: Table II shows that, ***both of our AGR-agnostic attacks significantly outperform***

Table II: Comparing state-of-the-art model poisoning attacks and our attacks under various threat models from Table I, when cross-silo FL is used. In all the settings, the impact of our AGR-tailored attacks is significantly higher than that of AGR-tailored Fang attacks. While both of our AGR-agnostic attacks outperform AGR-agnostic LIE attacks in most cases. We assume 20% malicious clients and, except for 'No attack' column, report *the attack impact* I_θ (Section V-B). For each adversary, we bold I_θ of the strongest attack.

Dataset (Model)	AGR	No attack (A_θ)	Gradients of benign devices are known					Gradients of benign devices are unknown				
			AGR tailored (`agr-updates`)		AGR agnostic (`updates-only`)			AGR tailored (`agr-only`)		AGR agnostic (`agnostic`)		
			Fang [17]	Ours	LIE [4]	Our attacks		Fang [17]	Ours	LIE [4]	Our attacks	
						Min-Max	Min-Sum				Min-Max	Min-Sum
CIFAR10 (Alexnet)	Krum	53.5	21.8	**43.6**	9.9	17.4	**30.1**	19.8	**43.1**	18.1	13.7	**30.2**
	MKrum	67.6	12.6	**36.8**	20.5	27.8	**30.8**	11.2	**35.3**	19.7	**31.7**	30.4
	Bulyan	66.9	12.3	**45.6**	33.8	35.5	**44.5**	11.8	**34.6**	30.0	40.6	**41.1**
	TrMean	67.7	15.8	**45.8**	22.8	**41.6**	33.5	12.9	**45.0**	19.4	**38.7**	27.9
	Median	65.5	12.9	**40.9**	20.7	34.7	**39.6**	12.6	**39.1**	19.7	34.1	**39.5**
	AFA	66.8	7.0	**47.0**	5.9	**31.5**	16.9	6.1	**46.8**	5.5	**22.2**	16.0
	FangTrmean	66.8	8.9	**56.3**	6.5	**42.9**	21.5	8.5	**56.0**	6.3	**42.1**	19.9
CIFAR10 (VGG11)	Krum	59.6	21.1	**49.1**	24.1	9.7	**28.7**	7.9	**32.2**	22.4	10.1	**25.9**
	MKrum	75.4	8.5	**32.5**	23.6	**32.0**	26.5	8.5	**32.3**	23.3	**32.3**	25.9
	Bulyan	75.0	25.9	**53.0**	36.4	34.2	**46.7**	24.5	**43.8**	33.2	42.7	**46.6**
	TrMean	75.5	25.2	**37.2**	28.4	**34.5**	23.6	22.2	**36.8**	24.2	**33.9**	20.4
	Median	73.2	24.7	**34.5**	28.9	**34.4**	30.3	24.8	**30.9**	28.3	**34.0**	29.7
	AFA	73.2	10.2	**42.3**	10.6	**21.9**	10.4	8.7	**28.3**	10.3	**19.9**	10.2
	FangTrmean	75.0	14.8	**64.9**	6.8	**35.8**	26.5	11.6	**64.9**	4.9	**34.1**	25.5
Purchase (FC)	Krum	62.1	6.0	**61.3**	-15.8	**60.6**	59.1	4.4	**60.8**	-17.7	**61.1**	61.0
	MKrum	91.9	13.7	**21.4**	1.5	**20.4**	18.4	12.2	**18.2**	1.7	**19.8**	16.4
	Bulyan	91.3	14.7	**28.7**	10.9	23.4	**30.0**	20.9	**28.4**	8.4	28.0	**30.3**
	TrMean	92.0	1.8	**23.4**	2.3	**16.9**	5.4	1.6	**22.2**	1.9	**26.8**	14.6
	Median	87.4	0.2	**11.0**	0.5	**11.6**	11.3	-1.0	**14.1**	-1.6	**13.4**	12.6
	AFA	91.7	1.5	**3.4**	0.7	**1.4**	0.7	1.4	**2.8**	0.2	**1.3**	0.5
	FangTrmean	91.9	1.2	**89.2**	0.5	**18.9**	8.4	0.9	**89.2**	0.5	**8.5**	7.8
MNIST (FC)	Krum	88.6	20.5	**33.9**	12.4	0.1	**28.5**	17.4	**24.1**	9.4	0.7	**25.3**
	MKrum	96.1	11.5	**18.6**	6.1	**16.3**	13.2	10.6	**15.6**	3.3	**15.0**	12.6
	Bulyan	95.4	6.9	**7.5**	**9.2**	4.8	8.2	7.1	**8.2**	6.7	5.1	**7.7**
	TrMean	96.2	1.8	**11.0**	6.4	**11.6**	9.3	1.7	**10.6**	5.1	**8.9**	8.5
	Median	93.2	1.7	**4.4**	1.9	**3.6**	2.2	1.5	**4.1**	1.8	**3.4**	2.0
	AFA	96.5	0.7	**2.5**	1.0	1.2	**2.2**	0.2	**1.8**	0.5	0.4	**1.6**
	FangTrmean	96.1	0.3	**84.2**	0.5	**23.4**	9.7	0.0	**84.2**	0.0	**21.8**	8.3
FEMNIST (CNN)	Krum	69.3	18.3	**30.0**	0.9	0.1	**9.8**	1.9	**2.9**	0.2	1.1	**8.0**
	MKrum	86.6	34.5	**78.8**	15.7	**79.5**	61.7	30.8	**57.1**	10.2	**79.5**	61.4
	Bulyan	86.1	38.9	**41.0**	32.0	20.1	**40.0**	35.6	**40.5**	20.5	18.7	**30.4**
	TrMean	86.7	7.2	**24.3**	19.1	**29.7**	26.8	7.9	**20.1**	14.4	24.7	**25.2**
	Median	77.1	2.7	**30.2**	12.0	**26.7**	17.1	0.8	**18.2**	5.8	**19.8**	16.6
	AFA	84.6	6.2	**77.0**	7.4	**74.4**	50.0	2.1	**75.3**	4.6	**74.0**	46.0
	FangTrmean	86.0	7.6	**83.1**	1.8	**81.6**	62.3	2.8	**83.0**	1.7	**78.3**	60.1

LIE, the state-of-the-art AGR-agnostic attack for most of the FL settings that we evaluate. For MNIST with Krum, the impact of Min-Sum attack (simply Min-Sum) is $3\times$ that of LIE, for both `updates-only` and `agnostic` adversaries. Except for CIFAR10 with VGG11, we note significantly higher impacts of Min-Sum on Krum than that of LIE. Note that *LIE, due to its small noise addition, regularizes and increases accuracy of the global model trained on Purchase using Krum*. For Bulyan, with `agnostic` and `updates-only` adversaries, Min-Sum significantly outperforms LIE by amounts varying from 1.8% (for MNIST) to 22.1% (for Purchase) depending on the classification task.

On the other hand, Min-Max is more effective against Multi-krum and outperforms LIE by amounts varying from 10.2% (MNIST) to 18.1% (Purchase) depending on the classification task. Min-Max is more effective against AFA, which also computes an average of gradients in a selection set. On AFA, Min-Max outperforms LIE for all datasets but MNIST dataset; for MNIST the two attacks have almost the same impacts. Min-Max is also more effective than LIE and Min-Sum attacks against Trimmed-mean, Median, and FangTrmean AGRs. For instance, depending on the classification task, Min-Max is almost $1.2\times$ (for CIFAR10 + VGG11) to $8\times$ (for Purchase) more impactful than LIE against Trimmed-mean, while it is almost $1.2\times$ (for CIFAR10 + VGG11) to $20\times$ (for Purchase) more impactful than LIE against Median.

LIE attack is ineffective, because it adds very small amounts of noises to compute its malicious gradients, while our AGR-agnostic attacks are much more impactful as they find the most malicious gradient within a ball formed by the benign gradients (Figure 1-(b,c)). For the same reason, for all the considered scenarios, except for the combination of Krum and FEMNIST dataset ***one or more of our AGR-agnostic attacks also outperform AGR-tailored Fang attacks***. Due to the extreme non-iid nature of FEMNIST, the malicious gradients of our AGR-agnostic attacks can be arbitrarily far

from benign gradients, which Krum can easily discard.

Reasons for the differences in the impacts of Min-Max and Min-Sum attacks: Min-Max finds the malicious gradient whose maximum distance from a benign gradient is less than the maximum distance between any two benign gradient. While, Min-Sum finds the malicious gradient such that the sum of its distances from all the other gradients is less than the sum of distances of any benign gradient from other benign gradients. Therefore, *as Figures 1-(b, c) demonstrate, the radius of search of malicious gradients of Min-Max is much larger than that of Min-Sum.* Therefore, the malicious gradients of Min-Sum more effectively circumvent the filtering of Krum and Bulyan AGRs, and therefore, are more impactful against these AGRs. For the same reason, Multi-krum selects a lesser number of malicious gradients of Min-Max than that of Min-Sum. But, as Multi-krum averages the selected gradients, Min-Max, with significantly more malicious gradients, damages the Multi-krum aggregate more effectively than Min-Sum.

Finally, we note that for AGR-agnostic attacks, we observe a few cases in Table II where the attacks with `agnostic` adversary have slightly more impact than those with `updates-only` adversary. For example, Min-Max attack on (CIFAR10 + Alexnet + Multi-krum) with `agnostic` adversary has 3.9% more impact than with `updates-only` adversary. The reason for this are various sources of randomness in our experiments. More specifically, we do not use the exact same set of gradients to compute malicious gradients under the two adversaries. Instead, we instantiate the whole FL training every time we compute the attack impact. Therefore, empirical randomness in running the two different instantiations may cause this behavior; the randomness can be in initial model parameters, in partitioning of dataset among clients, etc. Our experimental results are the average of three such instantiations for each of the presented result, and such empirical anomalies can be mitigated in various ways, including setting the seed for different random number generators and averaging over multiple runs of experiments.

B. Comparing different threat models

Our work is the first to comprehensively consider knowledge based threat models (or adversaries) while designing model poisoning attacks on FL. In this section, we compare the performances of our attacks by the adversaries given in Table I. For clarity of interpretation, we compare the two extreme adversaries—`agr-updates` and `agnostic`; the other adversaries can be compared using the bold impacts in Table II. Note that, ***even the weakest adversary,*** `agnostic`***, can effectively poison FL using our AGR-agnostic attacks***.

For Krum, we analyze the maximum reduction in the attack impact, I_θ, when the adversary changes from `agr-updates` to `agnostic`. I_θ reduces from 39.2% to 27.0% for MNIST, 43.6% to 30.2% for CIFAR10 with Alexnet, 49.1% to 25.9% for CIFAR10 with VGG11, and 30.0% to 1.9% for FEMNIST. This is expected, as Krum selects a single gradient as its aggregate, our attacks can fine tune the malicious gradient with the exact knowledge of all the benign gradients under `agr-updates` adversary. But such fine tuning is not possible with `agnostic` adversary. Nevertheless, for Purchase, I_θ due to both the adversaries are the same.

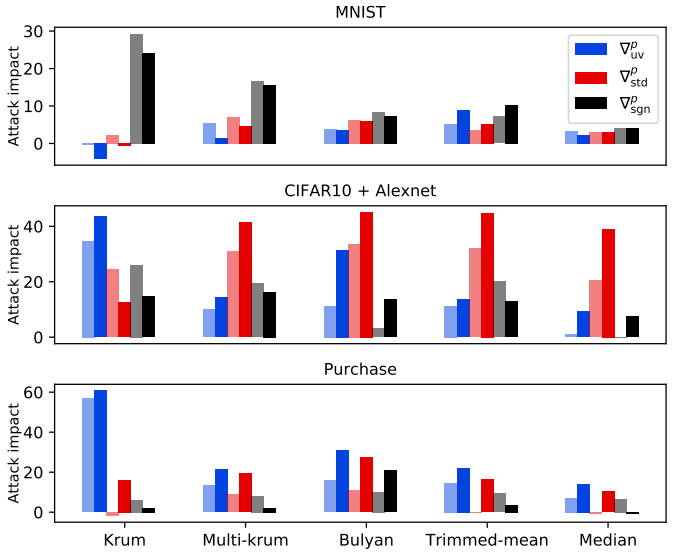


Figure 2: **Selecting an effective perturbation**: As explained in Section VI-C2, for a given FL setting, if the AGR is known, our adversary *emulates* attacks on the AGR using different ∇^p's and selects ∇^p with the highest attack impact (light bars). For unknown AGR, the adversary selects ∇^p which has the highest impact on the maximum number of AGRs. This selection method is reliable due to the transferability of attack impacts of ∇^p from the emulated settings (light bars) to actual FL (dark bars).

For all the other AGRs, the differences in the impacts of the two adversaries is significantly lesser than for Krum. For instance, for Multi-krum, changing adversary from `agr-updates` to `agnostic` reduces I_θ of our attack from 14.9% to 14.4% for MNIST, 36.8% to 31.7% for CIFAR10 with Alexnet, 32.5% to 32.3% for CIFAR10 with VGG11, 21.4% to 19.8% for Purchase, and 78.8% to 55.3% for FEMNIST. We observe similar differences in impacts for the rest of the AGRs across all the datasets. This is because, unlike Krum AGR, the other AGRs aggregate multiple gradients, including many benign gradients. Hence, neither of the adversaries can eliminate the good impact of these benign gradients on the final aggregates, which reduces the gap between their performances. Figure 4 depicts the differences in I_θ's of `agr-only` and `agnostic` adversaries.

C. Effect of perturbation vectors

In this section, we show the significant effect of the choice of perturbation vector, ∇^p, on the impacts of our attacks, I_θ. Then, we give a procedure to select the most effective ∇^p for a given FL setting.

Section IV-A proposes to fix the perturbation while optimizing the attack objectives and also introduces three types of perturbations. We assume 20% malicious clients, each with some data from the benign distribution, and use `agr-only` and `agnostic` adversaries.

1) Effect of perturbations on the attack impact: ***For a given dataset, model, and AGR, varying the perturbation ∇^p significantly changes the impact of our attacks***, as the dark bars in Figures 2 and 7 show. For instance, for MNIST with Krum, I_θ of ∇^p_{uv} is -4.7%, i.e., the global model accuracy increases under attack, while ∇^p_{sgn} increases I_θ significantly to 24.1%.

For Krum, we note such large differences in I_θ's of ∇^p's for the other datasets as well. For the other AGRs with MNIST as well, ∇^p_{sgn} is the most effective perturbation and outperforms the other perturbations by 3% to 15%. In case of CIFAR10 with either Alexnet or VGG11 model architectures, for all AGRs but Krum, standard deviation based ∇^p_{std} perturbation has the highest attack impact, while for Krum, ∇^p_{uv} has the highest impact. For instance, attacks on CIFAR10 + Alexnet + Multi-krum using ∇^p_{std}, ∇^p_{uv}, and ∇^p_{sgn} have impacts of 36.8%, 14.2%, and 16.3%, respectively. Similarly for Purchase, ∇^p_{uv} has the highest attack impact across all the AGRs.

Even for AFA AGR, we observe the similar behavior, e.g., for MNIST and CIFAR10, the most effective perturbations are ∇^p_{sgn} and ∇^p_{std}, respectively. For Fang-Trmean defense, all the perturbations can be equally effective, as far as the corresponding malicious gradients are sufficiently large. Hence, for a given dataset and Fang-Trmean, we simply choose the perturbation that works for most of the other AGRs. For instance, we use ∇^p_{sgn} and ∇^p_{std} for MNIST and CIFAR10, respectively. Due to space restrictions, we omit the figures for AFA and Fang-Trmean defenses.

2) How to select the most effective ∇^p?: Above, we showed that ***selecting the appropriate perturbation is the key to an effective model poisoning attack.*** However, as the adversary cannot know the end result of using a particular ∇^p, she must decide the ∇^p to use in each epoch of FL. Below, we provide a simple yet effective method to select ∇^p.

First, consider that the server's AGR is known. We propose that the adversary *emulate* AGR-tailored attacks (Section IV-B) on the given FL settings and select as its final ∇^p the most effective ∇^p in the emulated setting. An example of emulated AGR-tailored attack on MNIST with Krum is as follows: For MNIST we assume total of 100 clients including 20 malicious clients. Hence, the adversary emulates an FL setting with 20 benign and 4 malicious clients and mounts the AGR-tailored attack. In Figure 2, for each ∇^p, the lighter bars show the impacts of attacks on the emulated FL settings.

We compare the light and dark bars in Figure 2 and note that, ***the relative effects of different perturbations are the same across different AGRs, datasets, and models in both emulated FL settings (light bars) and actual FL (dark bars)***. In other words, the most effective perturbation in an emulated FL setting, is also the most effective perturbation in the corresponding actual FL. This transferability allows us to reliably select the most effective perturbation when the AGR is known. Finally, when the AGR is unknown, we simply pick the perturbation with the highest impact across maximum number of AGRs. For instance, we select ∇^p_{uv} for Purchase due to its highest attack impact on all AGRs. For CIFAR10, we choose ∇^p_{std} as it has the maximum impact on all but Krum AGR. We observe the same transferability for FEMNIST and CIFAR10 with VGG, as shown in Figure 7.

D. Effect of federated learning parameters

1) Effect of non-iid degrees of data distribution: In this section, we synthetically generate non-iid Purchase and MNIST datasets using the scheme proposed in [17]. We assume 20% malicious clients and plot the impacts of all the model poisoning attacks under `agr-only` and `agnostic` adversaries in Figure 3. Note that, ***as the non-iid degree of data increases, the impacts of all the model poisoning attacks increase***. This is because the higher degree of non-iid makes it difficult for AGRs to reliably detect and remove malicious gradients. This allows the adversaries to craft more malicious gradients without being detected and increase their attack impacts. Our attacks outperform previous attacks, especially at higher non-iid degrees of data. Our experiments with FEMNIST, a real world non-iid and imbalanced dataset, clearly show the significant superiority of our attacks over existing model poisoning attacks (Section VI-A).

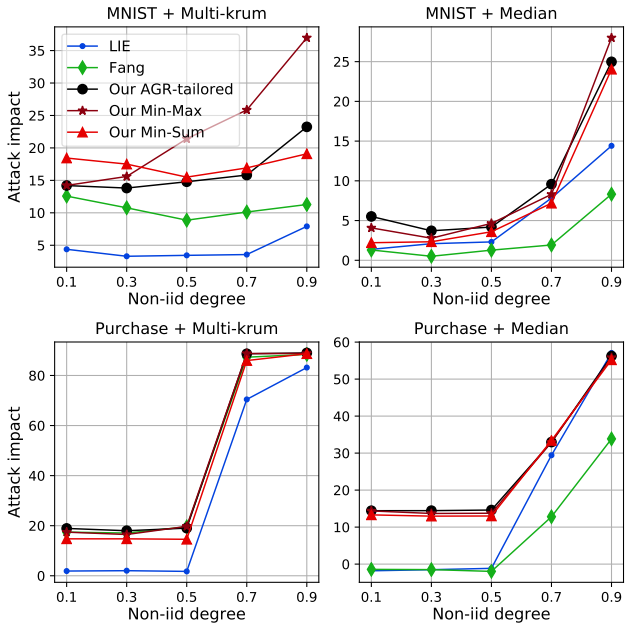


Figure 3: Effect of degree of non-iid nature of data on the impact of model poisoning attacks on FL. We use partial gradients knowledge, `agr-only` and `agnostic`, adversaries.

2) Effect of the percentage of malicious clients: Figure 4 shows the impacts of model poisoning attacks when the percentage of malicious clients in FL is varied from 5% to 24% for CIFAR10 with Alexnet and Purchase; Figure 6a shows the results for MNIST and FEMNIST datasets. We use partial gradients knowledge, `agr-only` and `agnostic`, adversaries. We note that, ***all of our attacks outperform the existing attacks for all the combinations of percentage of malicious clients, AGR algorithms, dataset, and model architectures***. For CIFAR10 with Krum, our AGR-tailored attack has impact of more than 43%, i.e., it reduces accuracy to random guessing, 10%, even with just 5% clients. For Purchase dataset, the distinction between impacts of ours and existing attacks is substantially higher than for CIFAR10. For Purchase, our AGR-tailored and AGR-agnostic Min-Sum attacks reduce the accuracy of Krum AGR to random guessing, i.e., close to 1%, with as few as 5% and 10% malicious clients, respectively. For all of the AGRs, ***with increasing percentage of malicious clients, the impacts of our attacks and the differences of the impacts of ours and existing attacks increase***. Interestingly for CIFAR10, LIE that uses ∇^p_{std} consistently outperforms Fang that uses ∇^b_{sgn}. This emphasizes our claim in Section VI-C that the most effective perturbation for model poisoning depends on the classification task.

E. Effect of cross-device setting

In this section, we evaluate the impact of our attacks when cross-device FL is used to learn on CIFAR10 dataset. More specifically, in each FL epoch, instead of processing all of the 50 clients, we process only 10 clients. As before, we evaluate for two model architectures, Alexnet and VGG11. The attack procedures for different AGRs do not change.

Table III shows the results. Similar to cross-silo setting, our AGR-tailored attacks outperform the state-of-the-art Fang attacks for both Alexnet and VGG11 architectures. For Alexnet with `agr-updates` adversary, our attack is $2\times$ (Trimmed-mean) to $11\times$ (Median) more impactful than Fang attack. We note similar results for `agr-only` adversary as well as VGG11 architecture in Table III.

For AGR-agnostic adversaries with Alexnet, we note that at least one of our Min-Sum and Min-Max attacks has up to $5\times$ more attack impact than the state-of-the-art LIE attack, for all but Krum AGR. For Krum AGR, LIE outperforms our attack by 0.2% and 1.2% under `updates-only` and `agnostic` adversaries, respectively. We note similar results for Alexnet with `agnostic adversary`. In case of VGG11 as well, at least one of our AGR-agnostic attacks has up to $10\times$ more impact than LIE, for all but Multi-krum and AFA AGRs. For Multi-krum and AFA, the LIE and Min-Max have almost equal attack impacts.

Finally we note that, overall the attack impacts are lower in cross-device setting than in cross-silo setting; the reduction in impacts varies widely based on AGR and model architecture. For instance, for Alexnet with Krum, Multi-krum, and Trimmed-mean, the impacts reduce by 9.5%, 14.6%, and 31.8%, respectively. The reason for this is that, in cross-device FL, the adversary cannot constantly corrupt the global model. Because, in many cross-device FL epochs, the number of malicious clients that the server selects can be negligible.

VII. The Divide and Conquer Defense (DnC)

Our strong attacks clearly motivate the need for more robust AGRs to defeat untargeted model poisoning attacks on FL. In this section, we first give the concrete lessons learned from our attacks that can guide the designs of future robust AGRs. Based on these lessons, we propose a novel robust AGR algorithm called *Divide-and-Conquer (DnC)*. Unlike state-of-the-art robust AGRs, which use distance based [31], [8] or simple pruning based filters [39], DnC performs dimensionality reduction using random sampling followed by spectral methods based outliers removal.

A. Lessons learned from our attacks

L1: The curse of dimensionality. Note that the theoretical error bounds of all of the robust AGRs [8], [31], [39], [16], [26], [2] depend on the dimensionality of their inputs. Hence, theoretical as well as empirical errors of these defenses explode for high dimensional gradients of neural networks [10] in FL. Therefore, ***dimensionality reduction of input gradients is necessary to address the curse of dimensionality***.

L2: Convergence is not enough. All the robust AGRs [8], [31], [39] give provably convergence guarantees for non-convex FL. However, for non-convex optimizations, such guarantees are meaningless due to large number of suboptimial local optima. This is indeed the case for all our attacks, where the global models converge, but to a suboptimial local optima. Therefore, ***robust AGRs should target the more robust criterion of malicious gradients removal***.

L3: Distance or dimension-wise pruning based filters are insufficient. Krum, Multi-krum, and Bulyan use ℓ_p distance based filtering, which, as [31], [4] point out and we show in our work, allows the malicious gradients to be close enough to the benign gradients to be undetected, while far enough to effectively poison the global model. Dimension-wise pruning in Trimmed-mean and Median allows adversary to craft gradients which significantly shift the aggregate in bad direction as ours and Fang [17] attacks show. Therefore, ***more principled approaches to filter malicious gradients are necessary for more robust AGRs***.

Algorithm 2 Our Divide-and-Conquer AGR Algorithm

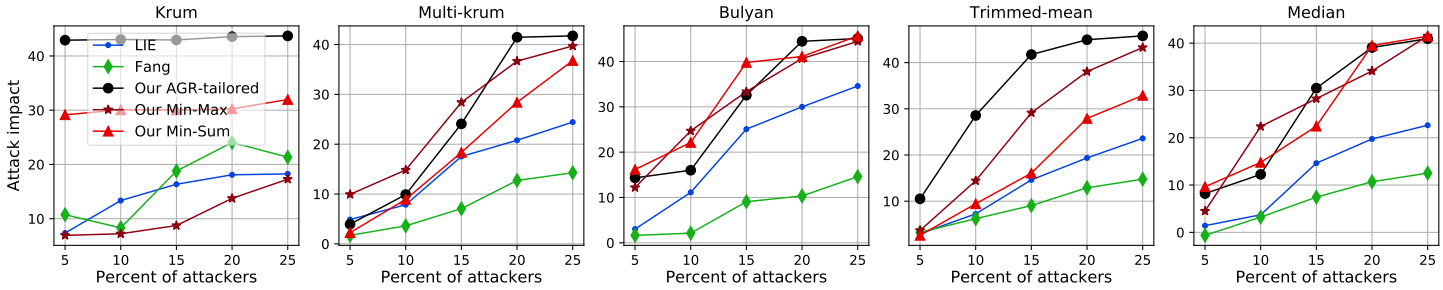
Input: Input gradients $\nabla_{\{i\in[n]\}}$, filtering fraction c, number of malicious clients m, niters, dimension of subsamples b, input gradients dimension d
$\mathcal{I}_{\text{good}} \leftarrow \emptyset$
while $i <$ niters **do**
  $r \leftarrow$ sorted set of size b of random dimensions $\leq d$
  $\widetilde{\nabla}_{\{i\in[n]\}} \leftarrow$ set of gradients subsampled using indices in r
  $\boldsymbol{\mu} = \frac{1}{n}\sum_{i\in[n]} \widetilde{\nabla}_i$ ▷ Compute mean of input gradients
  $\nabla^c = \widetilde{\nabla}_{\{i\in[n]\}} - \boldsymbol{\mu}$ ▷ ∇^c is a $n \times b$ matrix of centered input gradients
  Compute v, the top right singular eigenvector of ∇^c
  Compute *outlier scores* defined as $s_i = (\langle \nabla_i - \boldsymbol{\mu}, v\rangle)^2$
  $\mathcal{I} \leftarrow$ Set of $(n - c\cdot m)$ indices of the gradients with lowest outlier scores from s
  Append $\mathcal{I}$ to $\mathcal{I}_{\text{good}}$
  $i = i + 1$
end while
$\mathcal{I}_{\text{final}} \leftarrow \cap\, \mathcal{I}_{\text{good}}$ ▷ Compute intersection of sets in $\mathcal{I}_{\text{good}}$ as the final set of indices
$\nabla_a = \frac{1}{|\mathcal{I}_{\text{final}}|}\sum_{i\in\mathcal{I}_{\text{final}}} \nabla_i$
Output ∇_a

B. Our Defense

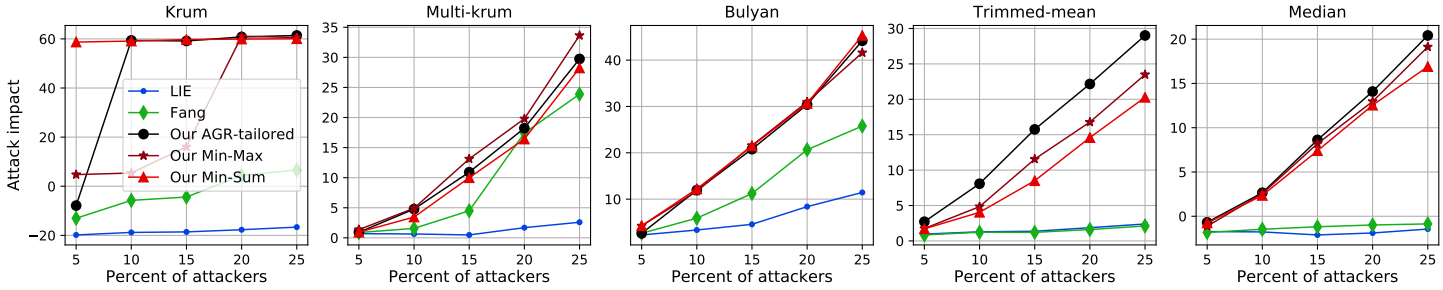
Intuition. Our intuition behind DnC is based on the lessons from Section VII-A. To address L3, DnC leverages singular value decomposition (SVD) based spectral methods for outliers detection and removal. These methods are shown to have state-of-the-art theoretical and empirical performance in mitigating data poisoning threats to the centralized learning settings [36], [15], [7]. To address L2, we provide theoretical analysis of our defense in Section VII-B that guarantees the removal of malicious gradients under certain conditions. Furthermore, we also construct adaptive attacks against DnC in Section VII-B to provide empirical evidence of the robustness of DnC. Note that SVD based defenses require $\mathcal{O}(d^3)$ memory and computational cost, hence, SVD cannot be performed directly on the high dimensional gradients in common FL settings [30] with dimensions of order on 10^6. To address this issue and the curse of dimensionality (L1), DnC performs random sampling

Table III: Comparing the state-of-the-art model poisoning attacks and our attacks under all threat models in Table I when *cross-device FL* is used. Our AGR-tailored attacks significantly outperform Fang attacks, while at least on of our AGR-agnostic attacks significantly outperforms LIE attack in most cases. Experimental setup is exactly the same as that of Table II.

Dataset (Model)	AGR	No attack (A_θ)	Gradients of benign devices are known					Gradients of benign devices are unknown				
			AGR tailored (`agr-updates`)		AGR agnostic (`updates-only`)			AGR tailored (`agr-only`)		AGR agnostic (`agnostic`)		
			Fang [17]	Ours	LIE [4]	Our attacks		Fang [17]	Ours	LIE [4]	Our attacks	
						Min-Max	Min-Sum				Min-Max	Min-Sum
CIFAR10 (Alexnet)	Krum	53.9	11.0	**34.0**	**19.9**	8.7	19.7	4.5	**15.8**	**14.9**	8.7	13.7
	MKrum	64.5	2.2	**22.2**	11.5	15.9	**17.2**	1.2	**14.4**	9.2	14.5	**15.5**
	Bulyan	63.9	2.0	**28.8**	13.0	**28.4**	26.4	1.9	**27.9**	9.9	**22.3**	8.6
	TrMean	64.9	8.3	**14.0**	9.3	**9.4**	7.3	3.2	**9.8**	4.9	**6.5**	4.2
	Median	62.4	1.8	**20.3**	4.1	**20.3**	17.8	0.2	**16.4**	-1.6	**15.8**	10.3
	AFA	66.2	1.6	**41.4**	3.8	**3.8**	2.6	1.0	**36.7**	3.4	**3.6**	1.9
	FangTrmean	64.5	7.2	**54.3**	3.6	**9.1**	6.5	4.3	**54.3**	2.3	**5.7**	5.3
CIFAR10 (VGG11)	Krum	59.3	3.8	**26.3**	15.0	6.7	**20.1**	1.2	**12.7**	**11.8**	1.8	10.7
	MKrum	72.0	1.4	**13.2**	**9.8**	9.0	8.5	1.0	**9.4**	**8.8**	7.3	8.3
	Bulyan	72.0	2.8	**18.8**	9.2	**24.0**	14.9	2.6	**17.6**	6.7	**16.6**	12.3
	TrMean	72.1	5.9	**8.7**	4.1	**6.0**	3.6	3.6	**8.1**	4.0	**5.2**	3.0
	Median	70.2	0.3	**11.2**	1.0	**12.8**	11.3	0.1	**10.7**	0.8	**9.9**	7.2
	AFA	71.8	2.1	**15.0**	**2.3**	1.8	1.5	2.0	**14.7**	**2.1**	1.7	1.1
	FangTrmean	71.9	1.9	**60.9**	3.6	**8.9**	4.4	0.6	**58.1**	2.7	**7.1**	4.0



(a) CIFAR10 with Alexnet architecture



(b) Purchase with fully connected network

Figure 4: Effect of increasing percentage of malicious clients on the impacts of model poisoning attacks on FL. We use adversaries who do not know the gradients on benign devices, i.e., `agr-only` and `agnostic` adversaries.

based dimensionality reduction of its input gradients. This way, DnC addresses the shortcomings of existing robust AGRs and makes SVD based more robust AGRs practical for FL.

Our DnC algorithm. Algorithm 2 describes the algorithm of our DnC AGR. First, DnC randomly picks a sorted set r of indices less than the dimensionality d of its input gradients (Line-4) and constructs a subsampled set $\widetilde{\nabla}$ of gradients using r (Line-5). For instance, if $d = 5$ and $r = [0, 3]$, a subsample of gradient $\nabla_i = \{\nabla_0, ...\nabla_4\}$ is $\widetilde{\nabla_i} = \{\nabla_0, \nabla_3\}$. Next, DnC computes a centered subsampled set ∇^c of $\widetilde{\nabla}$ using dimension-wise mean $\boldsymbol{\mu}$ of $\widetilde{\nabla}$ (Lines 6-7). Then DnC computes projections of centered gradients along their top right singular eigenvector v, computes a vector of outlier scores s, and removes $c \cdot m$ gradients with the highest scores (Lines 8-10). The remaining gradients are added to the set of good gradients. Such niters number of good sets are computed by randomizing r to reduce dependence on a single r. Finally, DnC computes its aggregate ∇_a as the average of the common gradients in all of the niters good sets (Lines 14-16).

Theoretical analysis. Our theoretical analysis of DnC provides guarantees on the removal of malicious gradients and leverages the analysis of SVD based defenses against data poisoning in the centralized settings [15], [36], [16], [28]. Definition 1 from [36], [28] defines a condition ϵ-*spectral separability*

under which two distributions can be separated using spectral methods, e.g., SVD.

Definition 1. *(ϵ-spectral separability) Consider $0 < \epsilon < 0.5$ and two finite covariance distributions, B and M. Let $U = (1-\epsilon)B + \epsilon M$ be a mixture of samples from B and M, and denote the top right singular eigenvector of U by v. Then B and M are ϵ-spectrally separable if there exists t such that*

$$\Pr_{X\sim B}[|\langle X - \mu_U, v\rangle| > t] < \epsilon$$
$$\Pr_{X\sim M}[|\langle X - \mu_U, v\rangle| < t] < \epsilon$$

If we consider that B and M, the distributions of benign and malicious gradients, respectively, are ϵ-spectrally separable, then by removing ϵ-fraction of gradients with maximum projections along the top eigenvector direction, we can remove malicious gradients from a set of benign and malicious gradients. Lemma 1 below gives the theoretical filtering guarantees of DnC, which are inspired from the guarantees of data poisoning defenses [36], [28].

Lemma 1. *Consider $0 < \epsilon < 0.5$ and two distributions B, M with means $\boldsymbol{\mu}_B, \boldsymbol{\mu}_M$ and covariances $\boldsymbol{\Sigma}_B, \boldsymbol{\Sigma}_M \preceq \sigma^2 I$. Let $U = (1-\epsilon)B + \epsilon M$ be a mixture of samples from B and M. Then B and M are ϵ-spectrally separable if $\|\boldsymbol{\mu}_B - \boldsymbol{\mu}_M\|_2^2 \geq \frac{6\sigma^2}{\epsilon}$.*

In the FL poisoning setting of ours, *Lemma 1 implies that if the means of poisoned and benign gradients are sufficiently separated, then the two types of gradients can be reliably separated using spectral methods*. Figure 5, demonstrates this exactly: the means of malicious gradients which effectively poison FL are sufficiently far from the means of benign gradients, and therefore, spectral methods can filter them. On the other hand, the malicious gradients which circumvent the criterion given in Lemma 1 have no impact on the accuracy of global model. We note that the result in Lemma 1 is common to SVD based outliers detection [15], [36], [16], [28], [11]; we provide it here for completeness and to give the intuition about the efficiency of DnC. Appendix B gives the formal proof of Lemma 1.

An adaptive attack against DnC. DnC provides provable theoretical guarantees on detection of malicious gradients. However, to provide empirical evidence on the robustness guarantees of DnC, we propose an adaptive attack by against the strongest `agr-updates` adversary who has the complete knowledge of the gradients of benign devices and of DnC.

Our adaptive attack is based on the general optimization framework proposed in Section IV-A. The attack is inspired from our AGR-tailored attack on Multi-krum AGR, because both DnC and Multi-krum compute a selection set and average the gradients in the final selection set. The intuition of the attack is to maximize the number of malicious gradients selected by DnC to maximize the bad impact on the final aggregate. This also ensures that the number of benign gradients selected and their good impact on the final aggregate are minimized. Consequently, the optimization problem for our adaptive attack is as follows:

$$\underset{\gamma}{\text{argmax}} \quad m = |\{\nabla \in \nabla^m_{\{i\in[m]\}} | \nabla \in \nabla^m_{\{i\in\mathcal{I}_{\text{final}}\}}\}| \qquad (8)$$
$$\nabla^m_{i\in[m]} = f_{\text{avg}}(\nabla_{\{i\in[n]\}}) + \gamma\nabla^p$$

where m is the number of malicious clients, $\mathcal{I}_{\text{final}}$ is the final set of candidate indices selected by DnC, ∇^p is perturbation, and γ is scaling factor. Note that, it is reasonable to assume that although the adversary knows DnC algorithm thoroughly, she cannot know the exact random indices r used for subsampling in Algorithm 2. Finally, we solve the optimization in (8) by finding the most impactful γ using Algorithm 1.

C. Evaluation of Our Defense

In this section, we first demonstrate the robustness of our DnC AGR against state-of-the-art [17], [4] and our model poisoning attacks from Sections IV and VII-B for iid datasets. We also analyze the effectiveness of spectral separability, and therefore of DnC, in defending against model poisoning on FL. Finally, we discuss the effectiveness of DnC for non-iid FEMNIST dataset.

1) Robustness of DnC for iid data: For iid datasets, i.e., MNIST, CIFAR10, and Purchase, we evaluate DnC against a strong adversarial setting with 20% malicious clients and the adversaries with complete knowledge of the gradients of benign clients, i.e., `agr-updates` when AGR is known and `updates-only` when AGR is unknown. We evaluate DnC using Fang and LIE, and our stronger AGR-tailored and AGR-agnostic attacks. For all these datasets, we set niters, c, and b in Algorithm 2 to 1, 1, and 10,000, respectively.

Robustness comparison with previous AGRs. Table IV shows, for each of the attacks, the attack impact on DnC; in parentheses, we show the impact of the attack on the most of existing AGRs, e.g., for Fang attack on CIFAR10 + Alexnet, Bulyan is the most robust AGR, hence, for CIFAR10 + Alexnet, we show the impact of Fang attack on Bulyan.

Below, *we analyze the AGRs based on the increase in accuracy of the global model under the strongest of the attacks*, i.e., based on the minimum A^*_θ (Section V-B) for the AGR. For an AGR, the minimum A^*_θ is obtained by subtracting the impact of the strongest attack, I_θ, from 'No attack' accuracy, A_θ. For instance, for CIFAR10 + Alexnet, our adaptive attack is the strongest attack against DnC and the corresponding minimum A^*_θ is 61.5% (as A_θ is 67.6% and the maximum I_θ is 6.1%). While our AGR-tailored attack is the strongest attack against the best of the existing AGRs, thus the minimum A^*_θ is 30.8% (as A_θ is 67.6% and the maximum I_θ is 32.5%). Hence, ***for CIFAR10 + Alexnet, DnC increases A^*_θ from 30.8% to 61.5% ($\sim$100% increase)***. For CIFAR10 + VGG11, DnC increases the minimum A^*_θ from 43.0% to 69.2% ($\sim$150% increase). For Purchase, DnC increase the minimum A^*_θ from 88.6% to 90.2%.

DnC increases the minimum A^*_θ for MNIST from 90.7% (93.2% − 2.5%) to 94.3% (96.2% − 1.9%). Although, the absolute increase due to DnC is small for MNIST, it is significant due to the simplicity of the tasks.

Robustness comparisons under cross-device FL setting. Now we compare robustness of previous AGRs and our DnC

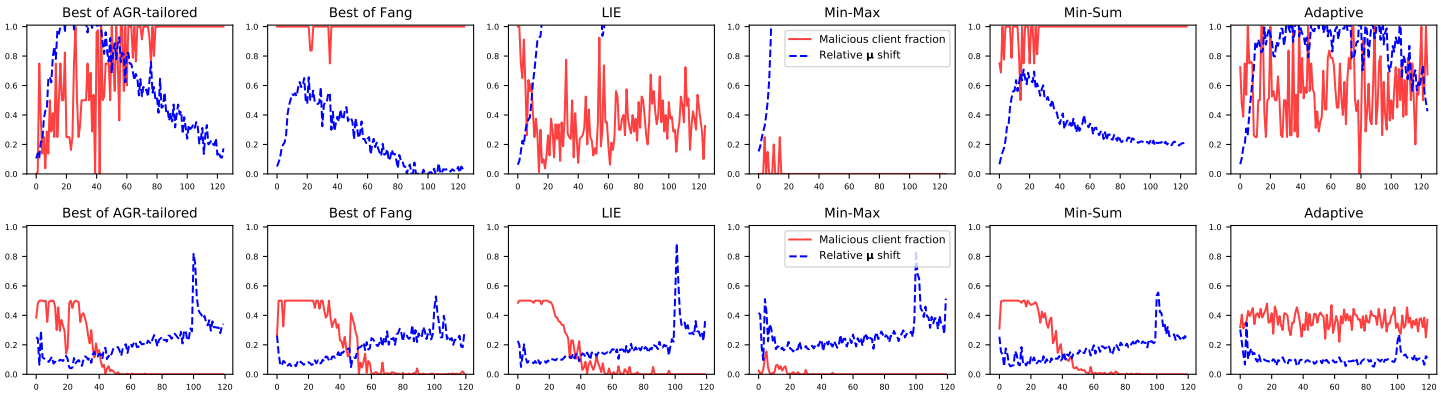


Figure 5: DnC selects high fractions of malicious gradients (red plots) iff the distances between $\boldsymbol{\mu}_B$ and $\boldsymbol{\mu}_M$, the means of benign and malicious gradients, are low (blue plots), i.e., poisoning impact of the malicious gradients is low. Upper row is for MNIST and lower row is for CIFAR10 + Alexnet. We use the strongest full knowledge `agr-updates` adversary.

Table IV: Our robust DnC AGR defends against all the existing model poisoning attacks for independently and identically distributed datasets. We consider the adversaries with complete knowledge of gradients of benign clients with 20% malicious clients. For each attack, we report its attack impact on DnC and on the existing defense with the highest global model accuracy A^*_θ, computed as $(A_\theta - I_\theta)$ from Table II.

Dataset + Model	No attack (A_θ)	Fang	LIE	Best of our AGR-tailored attacks	Our AGR-agnostic attacks		Adaptive attack
					Min-Max	Min-Sum	
CIFAR10 + Alexnet	67.6	**3.2** (7.0)	**3.0** (5.9)	**4.3** (36.8)	**3.5** (27.8)	**2.0** (16.9)	**6.1**
CIFAR10 + VGG11	75.5	**3.3** (8.5)	**1.7** (6.8)	**3.4** (32.5)	**2.5** (21.9)	**2.2** (10.4)	**6.3**
Purchase + FC	92.0	0.8 (**0.2**)	**0.5** (0.5)	**0.9** (3.4)	**0.6** (1.4)	**0.8** (0.7)	**1.8**
MNIST + FC	96.2	**0.1** (0.3)	**0.2** (0.5)	**1.8** (2.5)	**0.2** (1.2)	**1.2** (2.2)	**1.9**

Table V: Results of empirical robustness analysis of DnC for *cross-device FL* setting. We consider the adversaries with complete knowledge of gradients of benign clients with 20% malicious clients, and report A^*_θ as described in Table II.

Dataset + Model	No attack (A_θ)	Fang	LIE	Best of our AGR-tailored attacks	Our AGR-agnostic attacks		Adaptive attack
					Min-Max	Min-Sum	
CIFAR10 + Alexnet	64.6	**0.6** (1.6)	**0.3** (3.8)	**0.2** (14.0)	**0.3** (3.8)	**0.0** (2.6)	**3.4**
CIFAR10 + VGG11	72.1	**0.8** (1.4)	**0.3** (2.3)	**2.0** (8.7)	**0.4** (1.8)	**0.4** (1.5)	**4.1**

Table VI: For non-iid FEMNIST dataset, DnC cannot mitigate our attacks in the worst case settings when the adversary knows gradients of the benign devices. But, mitigates all the attacks in the more practical settings when the gradients of benign devices are unknown. We report I_θ on DnC of all adversaries in Table I with 20% malicious clients. 'No attack' accuracy A_θ for FEMNIST with DnC is 86.6%.

AGR	Gradients of benign devices are known				Gradients of benign devices are unknown			
	Best of AGR-tailored (`agr-updates`)	AGR-agnostic (`updates-only`)		Adaptive attack	Best of AGR-tailored (`agr-only`)	AGR-agnostic (`agnostic`)		Adaptive attack
		Min-Max	Min-Sum			Min-Max	Min-Sum	
DnC	48.1	13.8	79.3	78.6	12.7	9.3	11.7	10.2
DnC + resampling	79.3	80.5	45.9	77.6	77.5	79.1	43.4	70.6

when cross-device FL is used. We use CIFAR10 dataset with Alexnet and VGG11 architectures; Table V shows the results in similar fashion as Table IV. As before, we analyze robustness of an AGR based on the minimum A^*_θ for the AGR. We note that the impact of attacks on DnC reduces in cross-device FL, as for the other AGRs. **For CIFAR10 + Alexnet, with no attack accuracy, A_θ, of 64.6%, DnC increases A^*_θ from 50.6% to 61.2%**: for best of existing AGRs, our AGR-tailored attack is the strongest attack with I_θ of 14.0%, i.e., A^*_θ $(A_\theta - I_\theta)$ of 50.6%. While for DnC, our adaptive attack is the strongest with I_θ of 3.4%, i.e., A^*_θ of 61.2%. Similarly, for CIFAR10 + VGG11, DnC increases A^*_θ from 63.4% to 68.0%. The increase in A^*_θ due to DnC in cross-device FL is lower, because the impact of attacks on previous AGRs is lower, which leaves smaller room for improvements.

Why DnC is superior? ***The strong robustness of DnC stems from the effective filtering guarantees of Lemma 1***, which we empirically confirm in Figure 5: Here, in each epoch of FL + DnC training, we compute the fraction of malicious clients DnC selects and the norm of the difference between means of benign and malicious gradients relative to the norm of mean of the benign gradients, i.e. $\boldsymbol{\mu}_{\text{shift}} = \frac{\|\boldsymbol{\mu}_B - \boldsymbol{\mu}_M\|_2^2}{\|\boldsymbol{\mu}_B\|_2^2}$. We then average these entities over a few epochs for presentation clarity.

We observe in Figure 5 that for MNIST (upper row), Fang and Min-Sum attacks evade DnC's detection, but have very small $\boldsymbol{\mu}_{\text{shift}}$ which leads to high accuracy of global model A^*_θ. DnC mitigates LIE even when LIE evades DnC's detection to some extent and introduces large $\boldsymbol{\mu}_{\text{shift}}$. We suspect that, this is because LIE uses ineffective perturbation ∇^b_{std} for MNIST (Figure 2). Our AGR-tailored and adaptive attacks

evade DnC's detection to some extent by maintaining low μ_{shift}. Hence, MNIST, due to its simplicity, withstands their poisoning impact.

For CIFAR10 + Alexnet, we observe that DnC effectively filters malicious gradients of all but our adaptive attack. However, the adaptive attack manages to evade DnC's detection only due to low μ_{shift}, which is insufficient to poison DnC based FL.

2) Robustness of DnC for non-iid data: Table VI shows the evaluation of DnC for FEMNIST, an imbalanced and non-iid datasets. We set niters, c, and b in Algorithm 2 to 1, $(n-1)/n$, and 10,000, respectively We note that, ***DnC cannot defend at least one of our attacks by the strongest adversaries with complete knowledge of the gradients of benign clients***, i.e., `agr-updates` and `updates-only` adversaries. Min-sum has attack impact of 79.3%, i.e., it reduces the accuracy from 86.6% in the benign setting to 7.3%. Here, We omit evaluation of DnC against Fang and LIE attacks, as they are strictly weaker than all of our attacks against FEMNIST.

Furthermore, resampling [19], a mechanism proposed to reduce non-iid nature of the input gradients, exacerbates DnC's robustness (also of existing AGRs as shown in Table VII).

Even the benign gradients of FEMNIST with highly non-iid nature, do not point in a single direction, and therefore, DnC cannot reliably detect malicious gradients. This allows adversaries to easily circumvent DnC's detection and mount strong attacks. ***However, DnC mitigates all of the model poisoning on FL by more practical adversaries who do not know the gradients on benign devices***, i.e., `agr-only` and `agnostic` adversaries. The maximum attack impact is only 12.7%, i.e., the maximum accuracy of the global model due to DnC is 73.9%. The most robust of existing AGRs is Krum and the corresponding maximum accuracy is 66.4%.

We note that, defending the real-world non-iid FL settings from the worst case model poisoning attacks is a well-known challenging task [17], [19], [18] and a limitation of DnC in its current form. We leave investigating further to improve DnC to make it robust to the non-iid settings to future work.

VIII. Conclusions

We presented a general framework to mount systematic model poisoning attacks on FL. We demonstrated that our framework results in attacks that outperform state-of-the-art poisoning attacks against *all* Byzantine-robust FL algorithms and *by large margins*. We gave concrete reasons for the strength of our attacks, which future Byzantine-robust FL algorithms should address. We also presented a robust aggregation algorithm, called *divide-and-conquer*, that outperforms *all* existing robust aggregation algorithms in defeating poisoning attacks on FL.

References

[1] Acquire Valued Shoppers Challenge at Kaggle. https://www.kaggle.com/c/acquire-valued-shoppers-challenge/data, 2019. [Online; accessed 19-June-2020].

[2] Dan Alistarh, Zeyuan Allen-Zhu, and Jerry Li. Byzantine stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pages 4613–4623, 2018.

[3] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. *arXiv preprint arXiv:1807.00459*, 2018.

[4] Moran Baruch, Baruch Gilad, and Yoav Goldberg. A little is enough: Circumventing defenses for distributed learning. *Advances in Neural Information Processing Systems*, 2019.

[5] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. *arXiv preprint arXiv:1811.12470*, 2018.

[6] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. In *International Conference on Machine Learning*, pages 634–643, 2019.

[7] B. Biggio, B. Nelson, and P. Laskov. Poisoning attacks against support vector machines. *In Proceedings of 29th International Conference on Machine Learning*, 2012.

[8] Peva Blanchard, Rachid Guerraoui, Julien Stainer, et al. Machine learning with adversaries: Byzantine tolerant gradient descent. In *Advances in Neural Information Processing Systems*, pages 119–129, 2017.

[9] Sebastian Caldas, Peter Wu, Tian Li, Jakub Konečnỳ, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

[10] Hongyan Chang, Virat Shejwalkar, Reza Shokri, and Amir Houmansadr. Cronus: Robust and heterogeneous collaborative learning with black-box knowledge transfer. *arXiv preprint arXiv:1912.11279*, 2019.

[11] Moses Charikar, Jacob Steinhardt, and Gregory Valiant. Learning from untrusted data. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 47–60. ACM, 2017.

[12] Lingjiao Chen, Hongyi Wang, Zachary Charles, and Dimitris Papailiopoulos. Draco: Byzantine-resilient distributed training via redundant gradients. In *International Conference on Machine Learning*, pages 903–912, 2018.

[13] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. Emnist: Extending mnist to handwritten letters. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 2921–2926. IEEE, 2017.

[14] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Andrew Senior, Paul Tucker, Ke Yang, Quoc V Le, and Andrew Y Ng. Large scale distributed deep networks. *Advances in neural information processing systems*, 2012.

[15] Ilias Diakonikolas, Gautam Kamath, Daniel Kane, Jerry Li, Jacob Steinhardt, and Alistair Stewart. Sever: A robust meta-algorithm for stochastic optimization. In *International Conference on Machine Learning*, pages 1596–1606, 2019.

[16] Ilias Diakonikolas, Gautam Kamath, Daniel M. Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. Being robust (in high dimensions) can be practical. *In Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 2017.

[17] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Local model poisoning attacks to byzantine-robust federated learning. In *29th USENIX Security Symposium (USENIX Security 20)*, Boston, MA, aug 2020. USENIX Association.

[18] Avishek Ghosh, Justin Hong, Dong Yin, and Kannan Ramchandran. Robust federated learning in a heterogeneous environment. *arXiv preprint arXiv:1906.06629*, 2019.

[19] Lie He, Sai Praneeth Karimireddy, and Martin Jaggi. Byzantine-robust learning on heterogeneous datasets via resampling. *arXiv preprint arXiv:2006.09365*, 2020.

[20] Matthew Jagielski, Aline Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li. Manipulating machine learning: Poisoning attacks and countermeasures against regression learning. *39th IEEE Symposium on Security and Privacy*, 2018.

[21] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

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

[23] Jakub Konečný, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2017.

[24] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009.

[25] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 2012.

[26] Kevin Lai, Anup Rao, and Santosh Vempala. Agnostic estimation of mean and covariance. *In 2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, 2016.

[27] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[28] Jerry Zheng Li. *Principled approaches to robust machine learning and beyond*. PhD thesis, Massachusetts Institute of Technology, 2018.

[29] Saeed Mahloujifar, Mohammad Mahmoody, and Ameer Mohammed. Universal multi-party poisoning attacks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 4274–4283, 2019.

[30] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. *Proceedings of the 20 th International Conference on Artificial Intelligence and Statistics*, 2017.

[31] El Mahdi El Mhamdi, Rachid Guerraoui, and Sébastien Rouault. The hidden vulnerability of distributed learning in byzantium. In *International Conference on Machine Learning*, pages 3518–3527, 2018.

[32] Luis Muñoz-González, Battista Biggio, Ambra Demontis, Andrea Paudice, Vasin Wongrassamee, Emil C Lupu, and Fabio Roli. Towards poisoning of deep learning algorithms with back-gradient optimization. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 27–38. ACM, 2017.

[33] Luis Muñoz-González, Kenneth T Co, and Emil C Lupu. Byzantine-robust federated machine learning through adaptive model averaging. *arXiv preprint arXiv:1909.05125*, 2019.

[34] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[35] Ziteng Sun, Peter Kairouz, Ananda Theertha Suresh, and H Brendan McMahan. Can you really backdoor federated learning? *arXiv preprint arXiv:1911.07963*, 2019.

[36] Brandon Tran, Jerry Li, and Aleksander Madry. Spectral signatures in backdoor attacks. In *Advances in Neural Information Processing Systems*, pages 8000–8010, 2018.

[37] Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta. Generalized byzantine-tolerant sgd. *arXiv preprint arXiv:1802.10116*, 2018.

[38] Cong Xie, Sanmi Koyejo, and Indranil Gupta. Fall of empires: Breaking byzantine-tolerant sgd by inner product manipulation. *arXiv preprint arXiv:1903.03936*, 2019.

[39] Dong Yin, Yudong Chen, Kannan Ramchandran, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. *In Proceedings of the 35th International Conference on Machine Learning*, 2018.

APPENDIX

A. Missing experiments

Figure 6a shows the results for the attack impacts of LIE, Fang, and our model poisoning attacks on FEMNIST and MNIST datasets, when the percentage of malicious clients is varied from 5% to 24%. Detailed comparison is in Section VI-D2.

Table VII shows the results for the empirical robustness of three state-of-the-art robust AGRs when coupled with recently proposed resampling mechanism [19]. We note that similar to our DnC, resampling reduces the robustness existing AGRs.

Figure 7 shows the impact of the three perturbation vectors introduced in Section IV-A on emulated (light bars) and actual FL settings. Detailed discussion in Section VI-C2.

B. Proof of Lemma 1

The outliers detection guarantees of our robust AGR DnC depend on Lemma 1, which is a standard result in spectral-methods based outliers detection [36], [28], [15], [11]. For completeness, we provide its proof motivated from [28], [36].

As in Lemma 1, consider two distributions B, M with means $\boldsymbol{\mu}_B, \boldsymbol{\mu}_M$ and covariances $\boldsymbol{\Sigma}_B, \boldsymbol{\Sigma}_M \preceq \sigma^2 I$. Let $U = (1-\epsilon)B + \epsilon M$ be a mixture of samples from B and M. The following holds for any unit vector u due to the Chebyshev's inequality:

$$\Pr_{X\sim B}[|\langle X - \mu_B, u\rangle| > t] \leq \frac{\sigma^2}{t^2} \tag{9}$$

$$\Pr_{X\sim M}[|\langle X - \mu_M, u\rangle| > t] \leq \frac{\sigma^2}{t^2} \tag{10}$$

Let $\Delta = \mu_B - \mu_M$ and v be the top right singular eigenvector of U. Ideally, the unit vector u should be a scaled version of Δ, which will maximally separates the samples from B and M. But as argued in [36], one can show any u with sufficient correlation with Δ suffices, which gives us Lemma

Lemma 2. *Let $\alpha > 0$ such that $|\langle u, \Delta\rangle| > \frac{\alpha\sigma}{\sqrt{\epsilon}}$. Then there exists $t > 0$ such that*

$$\Pr_{X\sim B}[|\langle X - \mu_B, u\rangle| > t] < \epsilon/\alpha^2$$
$$\Pr_{X\sim M}[|\langle X - \mu_M, u\rangle| > t] < \frac{\epsilon}{(\alpha-1)^2}$$

Proof: As $U = (1-\epsilon)B + \epsilon M$, we have

$$\langle X - \mu_U, u\rangle = \langle X - \mu_B, u\rangle + \epsilon\langle \Delta, u\rangle$$
$$\langle X - \mu_U, u\rangle = \langle X - \mu_M, u\rangle + (1-\epsilon)\langle \Delta, u\rangle$$

now assuming $t = \epsilon|\langle \Delta, v\rangle| + \frac{\sigma}{\sqrt{\epsilon}}$, we have

$$\Pr_{X\sim B}[|\langle X - \mu_U, v\rangle| > t]$$
$$\leq \Pr_{X\sim B}\left[|\langle X - \mu_B, v\rangle| > \frac{\sigma}{\sqrt{\epsilon}}\right] \leq \frac{\epsilon}{\alpha^2}$$

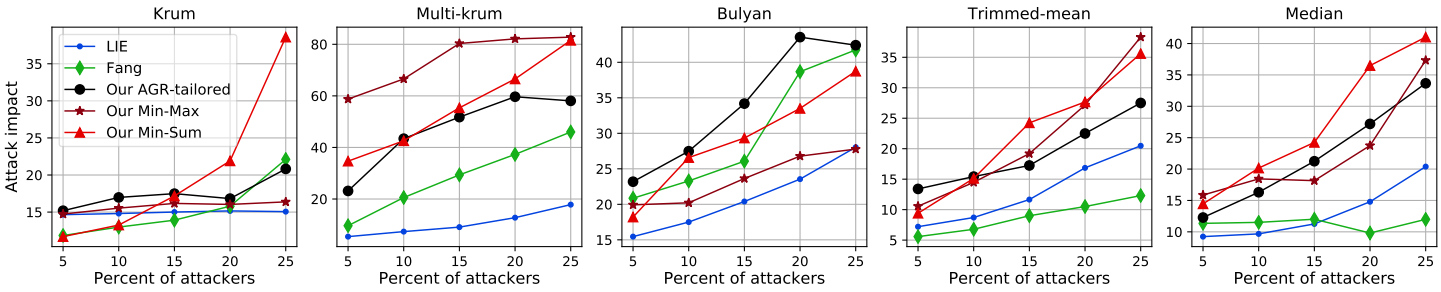
and by (9), we have

$$\Pr_{X\sim M}[|\langle X - \mu_U, v\rangle| < t]$$
$$\leq \Pr_{X\sim M}\left[|\langle X - \mu_M, v\rangle| > \langle \Delta, v\rangle - \frac{\alpha\sigma}{\sqrt{\epsilon}}\right]$$
$$\overset{(a)}{\leq} \Pr_{X\sim M}\left[|\langle X - \mu_M, v\rangle| > \frac{(\alpha-1)\sigma}{\sqrt{\epsilon}}\right]$$
$$\overset{(b)}{\leq} \frac{\epsilon}{(\alpha-1)^2}$$

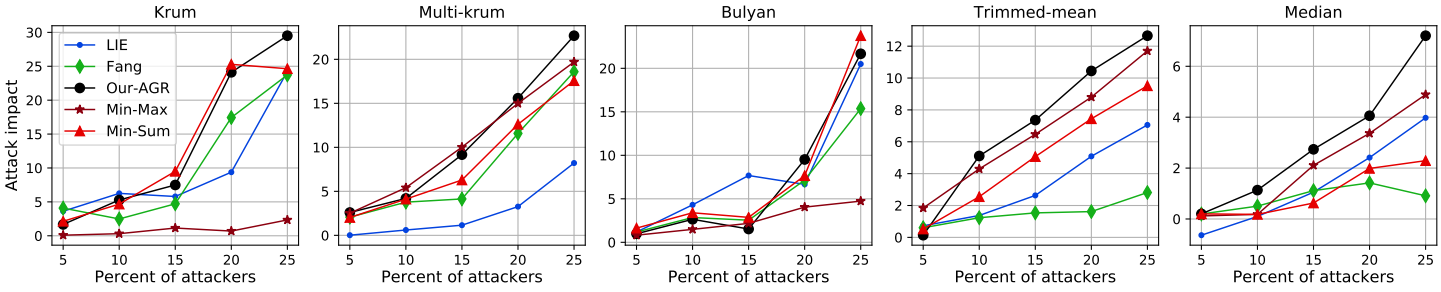
where (a) follows from the assumption and (b) from (10).

■

Lemma 3. *Under the assumptions of Lemma 1, we have $\langle \Delta, v\rangle^2 \geq \frac{2\sigma^2}{\epsilon}$.*



(a) FEMNIST with convolution neural network



(b) MNIST with CNN architecture

Figure 6: Effect of increasing percentage of malicious clients on the impacts of model poisoning attacks on FL.

Table VII: Resampling [19] significantly reduces the robustness of existing defenses against our attacks for all the threat models from Table I. We use 20% malicious clients and evaluate for all adversaries in Table I. This is expected, as resampling increases the number of malicious updates processed, and therefore, their poisoning impacts.

AGR	No attack (A_θ)	Updates of benign devices are known			Updates of benign devices are unknown		
		AGR-tailored (`agr-updates`)	AGR-agnostic (`updates-only`)		AGR-tailored (`agr-only`)	AGR-agnostic (`agnostic`)	
			Min-Max	Min-Sum		Min-Max	Min-Sum
Krum	69.3	30.0	0.1	9.8	2.9	1.1	8.0
Krum + resampling		**47.8**	**64.0**	43.6	**45.2**	**63.6**	58.0
Bulyan	86.1	41.0	20.1	40.0	40.5	18.7	**30.4**
Bulyan + resampling		**76.8**	**81.7**	53.9	**49.6**	10.0	8.1
Trmean	86.7	24.3	29.7	26.8	20.1	24.7	25.2
Trmean + resampling		**70.3**	**81.2**	46.0	**46.4**	**80.6**	68.5

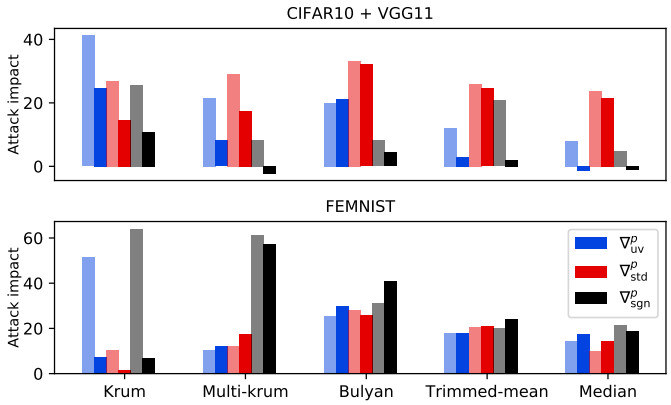


Figure 7: Selecting an effective perturbation for CIFAR with VGG11 and FEMNIST with CNN. Please check Section VI-C2 and Figure 2 for details.

Proof: We have

$$\mathbb{E}_{X\sim B}\Big[(X-\mu_U)(X-\mu_U)^T\Big] = \Sigma_B + \epsilon^2\Delta\Delta^T$$
$$\mathbb{E}_{X\sim M}\Big[(X-\mu_U)(X-\mu_U)^T\Big] = \Sigma_B + (1-\epsilon)^2\Delta\Delta^T \quad (11)$$

If Σ_U is the covariance of U, we have

$$\begin{aligned}\Sigma_U &= (1-\epsilon)\Sigma_B + \epsilon\Sigma_M + \epsilon(1-\epsilon)\Delta\Delta^T \\ \implies \Sigma_U &\succeq \epsilon(1-\epsilon)\Delta\Delta^T \\ \implies \|\Sigma_U\|_2 &\geq \epsilon(1-\epsilon)\|\Delta\|_2^2\end{aligned}$$

Next, we have

$$\begin{aligned}\epsilon(1-\epsilon)\|\Delta\|_2^2 &\leq v^T\Sigma_U v \\ &= (1-\epsilon)v^T\Sigma_B v + \epsilon v^T\Sigma_M v + \epsilon(1-\epsilon)\langle\Delta, v\rangle^2 \\ &\leq \sigma^2 + \epsilon(1-\epsilon)\langle\Delta, v\rangle^2\end{aligned}$$

Recall the assumption $\sigma^2 \leq \frac{\epsilon}{6}\|\Delta\|_2^2$, which gives us

$$\langle\Delta, v\rangle^2 \geq \Big(1 - \frac{1}{6(1-\epsilon)}\Big)\|\Delta\|_2^2 \overset{(a)}{\geq} \frac{2}{3}\|\Delta\|_2^2 \geq \frac{2\sigma^2}{\epsilon}$$

where (a) follows from the assumption that $\epsilon < 1/2$; taking square roots of both the sides gives the final result. ■

Finally combining the results of Lemmas 2 and 3 gives us the result of Lemma 1.