Adversarial Robustness through Bias Variance Decomposition: A New Perspective for Federated Learning

Yao Zhou*
Instacart & University of Illinois at Urbana Champaign
yaozhou3@illinois.edu

Jun Wu*
University of Illinois at Urbana Champaign
junwu3@illinois.edu

Haixun Wang
Instacart
haixun@gmail.com

Jingrui He
University of Illinois at Urbana Champaign
jingrui@illinois.edu

ABSTRACT

Federated learning learns a neural network model by aggregating the knowledge from a group of distributed clients under the privacy-preserving constraint. In this work, we show that this paradigm might inherit the adversarial vulnerability of the centralized neural network, i.e., it has deteriorated performance on adversarial examples when the model is deployed. This is even more alarming when federated learning paradigm is designed to approximate the updating behavior of a centralized neural network. To solve this problem, we propose an adversarially robust federated learning framework, named Fed_BVA, with improved server and client update mechanisms. This is motivated by our observation that the generalization error in federated learning can be naturally decomposed into the bias and variance triggered by multiple clients' predictions. Thus, we propose to generate the adversarial examples via maximizing the bias and variance during server update, and learn the adversarially robust model updates with those examples during client update. As a result, an adversarially robust neural network can be aggregated from these improved local clients' model updates. The experiments are conducted on multiple benchmark data sets using several prevalent neural network models, and the empirical results show that our framework is robust against white-box and black-box adversarial corruptions under both IID and non-IID settings.

CCS CONCEPTS

• **Information systems** → **Federated databases**; • **Computing methodologies** → **Adversarial learning**.

KEYWORDS

federated learning, bias-variance analysis, adversarial robustness

ACM Reference Format:
Yao Zhou, Jun Wu, Haixun Wang, and Jingrui He. 2022. Adversarial Robustness through Bias Variance Decomposition: A New Perspective for Federated Learning. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM '22), October 17–21, 2022, Atlanta, GA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3511808.3557232

*Both authors contributed equally to this research.

CIKM '22, October 17–21, 2022, Atlanta, GA, USA

ACM ISBN 978-1-4503-9236-5/22/10…$15.00
https://doi.org/10.1145/3511808.3557232

1 INTRODUCTION

The explosive amount of decentralized user data collected from the ever-growing usage of smart devices, e.g., smartphones, wearable devices, home sensors, etc., has led to a surge of interest in the field of decentralized learning. To protect the privacy-sensitive data of local clients, federated learning [26, 42] has been proposed. It only allows a group of clients to train local models using their own data on each individual device, and then collectively merges the model updates on a central server using secure aggregation [1]. Due to its high privacy-preserving property, federated learning has attracted much attention in recent years along with the prevalence of efficient light-weight deep models [16] and low-cost network communications [20].

Most existing federated learning approaches [18, 29, 38] focus on improving the strategies of local model training (e.g., local SGD [34]) and global aggregation (e.g., FedAvg [26]). The intuition behind them is to approximate the updating behavior of the centralized model trained on all clients' data. However, little effort (if any) has been devoted to systematically analyzing the adversarial robustness of federated learning paradigm. This becomes even more alarming when centralized machine learning models have been shown to be vulnerable to adversarial attacks when those models are deployed in the testing phase [13, 28]. It is studied [30, 32, 49] that a heuristic solution is to leverage the adversarial training techniques [25, 39] for clients' local training. However, it might suffer from expensive computation for local clients associated with limited storage and computational resources.

Our work studies the adversarial robustness of federated learning paradigm by investigating the generalization error incurred in the server's aggregation process from the perspective of bias-variance decomposition [9, 37]. Specifically, we show that this generalization error on the central server can be decomposed as the combination of **bias** (triggered by the main prediction of these clients) and **variance** (triggered by the variations among clients' predictions). This motivates us to propose a novel adversarially robust federated learning framework Fed_BVA. The key idea is to perform the local robust training on clients by supplying them with bias-variance perturbed examples generated from a tiny auxiliary training set on the central server. It has the following advantages. First, it encourages the

clients to consistently produce the optimal prediction for perturbed examples, thereby leading to a better generalization performance. Second, local adversarial training on the perturbed examples learns a robust local model, and thus an adversarially robust global model could be aggregated from clients' local updates. The experiments are conducted on neural networks with cross-entropy loss, however, other loss functions are also applicable as long as their gradients w.r.t. bias and variance are tractable to be computed.

It is worth noting that our problem setting fundamentally differs from the existing Byzantine-robust federated learning [3, 8, 24, 29]. To be more specific, those works proposed to improve the robustness of federated learning against Byzantine failures induced by corrupted clients' updates during model training, by performing client-level robust training or designing server-level aggregation variants with hyper-parameter tuning. In contrast, we focus on the adversarial robustness of federated learning against adversarial examples, when the model is deployed in the testing phase. Generally, the problem studied in this paper assumes that the learning process is clean (no malicious behaviors or Byzantine faults are observed in clients). But we also empirically show in Subsection 6.4.2 that when this assumption is violated, our `Fed_BVA` can be robust against both adversarial perturbation and Byzantine failures by incorporating Byzantine-robust aggregation variants [3, 6, 44].

Compared with previous work, our major contributions include:

- We provide the exact solution of bias-variance analysis w.r.t. the generalization error for neural networks in the federated learning setting. As a comparison, performing the adversarial training on conventional federated learning methods can only focus on the bias of the central model but ignore the variance.
- We demonstrate that the conventional federated learning framework is vulnerable to strong attacks with increasing communication rounds even if the adversarial training using the locally generated adversarial examples is performed on each client.
- Without violating the clients' privacy, we show that providing a tiny amount of bias-variance perturbed data from the central server to the clients through asymmetrical communication could dramatically improve the robustness of the training model under various adversarial settings.

The rest of this paper is organized as follows. The related work is reviewed in Section 2. In Section 3, the preliminary is provided. Then, in Section 4, a generic framework for adversarially robust federated learning is proposed, followed by the instantiated algorithm `Fed_BVA` with bias-variance attacks in Section 5. The experiments are presented in Section 6. We conclude this paper in Section 7.

2 RELATED WORK

2.1 Federated Learning

Federated learning [5, 20, 22, 26] aggregates the knowledge from a large number of local devices (e.g., smartphone [14]) with limited storage and computational resources. This aggregated knowledge can thus be leveraged to train the modern machine learning models (e.g., deep neural networks) [10, 18, 38]. However, federated learning algorithms over neural networks are approximating the update dynamics of centralized neural networks, thus resulting in inheriting the adversarial vulnerability of centralized neural networks [11, 13, 35, 40, 45–48]. More specifically, the deployed neural network models in federated learning might incorrectly classify the adversarial examples generated by the existing evasion attacks (e.g., FGSM [13], PGD [25]).

There are two lines of related works for improving the robustness of federated learning. One is to develop robust model aggregation schemes [3, 6, 8, 24, 29, 44] against Byzantine failures induced by corrupted client's updates during model training. Those works focus on performing client-level robust training or designing server-level aggregation variants with hyper-parameter tuning. The other one is to improve the model robustness against the adversarial attacks after model deployment [32, 49]. This idea fundamentally differs from the Byzantine-robust approaches in the following. (1) Different from Byzantine-robust model training, it is assumed that no malicious behaviors or Byzantine faults are observed in clients in training. (2) It improves the model robustness by refining the decision boundary around the adversarial examples, whereas Byzantine-robust models focus on mitigating the impact of potential clients' faults. In this paper, we will mainly focus on the second scenario by analyzing the adversarial robustness of federated learning from the perspective of bias-variance analysis. Compared to previous works [32, 49], the bias-variance analysis on local clients allows capturing the global model dynamics for improving the adversarial robustness of the aggregated global model in the federated learning setting.

2.2 Bias-Variance Decomposition

Bias-variance decomposition [12, 19] was originally introduced to analyze the generalization error of a learning algorithm. Then, a generalized bias-variance decomposition [9, 37] was studied in the classification setting which enabled flexible loss functions (e.g., squared loss, zero-one loss). Specifically, it is shown that under mild conditions, the generalization error of a learning algorithm can be decomposed into bias, variance and irreducible noise. In this case, bias measures how the trained models over different training data sets deviate from the optimal model, and variance measures the differences among those trained models. Moreover, it is previously observed [12] that the bias decreases and the variance increases monotonically with respect to the model complexity. It thus suggests that a better trade-off of bias and variance could lead to improving the generalization performance of a learning algorithm. Nevertheless, in recent years, it is empirically shown [33] that increasing the model complexity of deep neural networks tends to produce better generalization performance, which is in contradiction to previous bias-variance analysis. Following this idea, bias-variance trade-off was experimentally revisited on modern neural network models [2, 43]. It is found that for deep neural networks, the variance is more likely to increase first and then decrease with respect to the model complexity. We would like to point out that compared to standard supervised learning, federated learning can be better characterized by the bias-variance decomposition. That is because the trained models in local clients can be naturally applied to define the bias and variance in the generalization error of a learning algorithm. To the best of our knowledge, this is the first work studying the federated learning problem from the perspective of bias-variance analysis.

3 PRELIMINARIES

In this section, we formally present the problem definition and the bias-variance trade-off in the classification setting.

3.1 Federated learning

In federated learning [26, 42], there is a central server and K different clients, each with access to a private training set $\mathcal{D}_k = \{(x_i^k, t_i^k)\}_{i=1}^{n_k}$. Here, x_i^k, t_i^k, and n_k are the features, label, and number of training examples in the k^{th} client where $k = 1, \cdots, K$. Each data $\mathcal{D}_k$ is exclusively owned by client k and will not be shared with the central server or other clients.

The goal of standard federated learning is to learn the prediction function by aggregating the knowledge of user data from a set of local clients without leaking user privacy. A typical framework [26, 42] of federated learning can be summarized as follows: (1) *Client Update:* Each client updates local parameters w_k by minimizing the empirical loss over its own training set; (2) *Forward Communication:* Each client uploads its parameter update to the central server; (3) *Server Update:* It synchronously aggregates the received parameters; (4) *Backward Communication:* The global parameters are sent back to the clients.

3.2 Problem Definition and Motivation

It can be seen that the neural network in federated learning actually approximates the updating behavior of a centralized model. For example, the update behavior of FedAvg [26] is closely equivalent to the stochastic gradient descent (SGD) on the centralized data when each client only takes one step of gradient descent. That explains why it might potentially inherit the adversarial vulnerability of centralized neural network models [13, 35]. Therefore, in this paper, we study the adversarial robustness of neural networks in the federated learning setting. Formally, given a set of private training data $\{\mathcal{D}_k\}_{k=1}^K$ on K different clients, a learning algorithm $f(\cdot)$ and loss function $L(\cdot,\cdot)$, ***adversarially robust federated learning*** aims to output a trained model on the central server that is robust against adversarial perturbations on the test set $\mathcal{D}_{test}$.

It is previously [30, 32, 49] revealed that one can simply apply the adversarial training techniques [25] to clients' local training. Nevertheless, it is unclear how the local adversarial training on clients affects the adversarial robustness of the aggregated global model on the central server. To answer this question, we study the intrinsic relationship between the set of local clients' models and the aggregated server's model from the perspective of bias-variance trade-off. It is found that the generalization error of the server's model is induced by both bias and variance of local clients' models. Therefore, instead of separately focusing on the adversarial robustness of individual clients [32, 49], we propose to analyze the adversarially robust federated learning in a unified framework. The crucial idea is to generate some global adversarial examples shared with clients, by leveraging a tiny auxiliary training set $\mathcal{D}_s = \{(x_j^s, t_j^s)\}_{j=1}^{n_s}$ with n_s ($n_s \ll \sum_{k=1}^K n_k$) examples on the central server. This will not break the privacy constraints, as the local data in every client is not shared. In real scenarios, the auxiliary data can be some representative or synthetic template examples. For example, during the COVID-19 pandemic, hospitals (local clients) would like to consider a few publicly accessible template data with typical symptoms[1] for model training of the diagnostic system. Notice that federated semi-supervised learning [17] also considered a similar problem setting with labeled examples on the server and unlabeled examples on the clients. Instead, in our paper, we propose to improve the adversarial robustness of federated learned systems over local clients by taking those publicly accessible data $\mathcal{D}_s$ into consideration.

We would like to point out that our problem definition has the following properties. *(1) Asymmetrical communication:* The asymmetrical communication between each client and server cloud is allowed: the server provides both global model parameters and limited shared data to the clients; while each client uploads its local model parameters back to the server. This implies that compared to standard federated learning, the communication cost might increase due to those shared data, but it enables the improvement of adversarial robustness in federated learning (see Subsection 6.5.1 for more empirical analysis)[2]. *(2) Data distribution:* All training examples on the clients and the server are assumed to follow the same data distribution. However, the experiments show that our proposed algorithm also achieves outstanding performance under the non-IID setting (see Subsection 6.2), which could be commonly seen among personalized clients in real scenarios. *(3) Shared learning algorithm:* All the clients are assumed to use the identical model $f(\cdot)$, including architectures as well as hyper-parameters (e.g., learning rate, local epochs, local batch size, etc.).

3.3 Bias-Variance Trade-off

In this paper, we investigate the adversarially robust federated learning by studying the generalization error incurred in its aggregation process. We discover that the key to analyzing this error is from the perspective of the bias-variance trade-off. Following [9], we define the optimal prediction, main prediction as well as the bias, variance, and noise for any real-valued loss function $L(\cdot,\cdot)$ as follows:

Definition 3.1. (**Optimal Prediction and Main Prediction** [9]) Given a loss function $L(\cdot,\cdot)$ and a learning algorithm $f(\cdot)$, optimal prediction y_* and main prediction y_m for input x are defined as:

$$y_*(x) = \arg\min_y \mathbb{E}_t[L(y,t)] \quad y_m(x) = \arg\min_{y'} \mathbb{E}_{\mathcal{D}}[L(f_{\mathcal{D}}(x), y')]$$

where the label[3] t and data set $\mathcal{D}$ are viewed as the random variables to denote the class label and training set, and $f_{\mathcal{D}}$ denotes the model trained on $\mathcal{D}$.

Definition 3.2. (**Bias, Variance and Noise**) Given a loss function $L(\cdot,\cdot)$ and a learning algorithm $f(\cdot)$, the bias, variance and noise can be defined as follows.

$$B(x) = L(y_m, y_*),\ V(x) = \mathbb{E}_{\mathcal{D}}[L(f_{\mathcal{D}}(x), y_m)],\ N(x) = \mathbb{E}_t[L(y_*, t)]$$

Furthermore, there exists $\lambda, \lambda_0 \in \mathbb{R}$ such that the expected prediction loss $\mathbb{E}_{\mathcal{D},t}[L(f_{\mathcal{D}}(x), t)]$ for an example x can be decomposed into bias, variance and noise as follows:

$$\mathbb{E}_{\mathcal{D},t}[L(f_{\mathcal{D}}(x), t)] = B(x) + \lambda V(x) + \lambda_0 N(x) \tag{1}$$

In short, bias is the loss incurred by the main prediction w.r.t. the optimal prediction, and variance is the average loss incurred by all individual predictions w.r.t. the main prediction. Noise is conventionally assumed to be irreducible and independent of $f(\cdot)$. Our definitions of optimal prediction, main prediction, bias, variance and noise slightly differ from previous ones [9, 37]. For example, conventional optimal prediction was defined as $y_*(x) =$

[1] https://www.cdc.gov/coronavirus/2019-ncov/symptoms-testing/symptoms.html

[2] We would like to leave the trade-off of communication cost and adversarial robustness as our future work. For example, asymmetrical communication can be device-dependent in real scenarios, by acquiring the transmitted data based on devices' storage and computational capacities.

[3] In general, t is a non-deterministic function [9, 37] of x when the irreducible noise is considered. Namely, if x is sampled repeatedly, different values of t will be observed.

$\arg\min_y \mathbb{E}_t[L(t,y)]$, and it is equivalent to our definition when loss function is symmetric over its arguments, i.e., $L(y_1,y_2)=L(y_2,y_1)$.

4 THE PROPOSED FRAMEWORK

In this section, we present the adversarially robust federated learning framework. It follows the same paradigm of traditional federated learning (see Subsection 3.1) but with substantial modifications on the server update and client update as follows.

4.1 Server Update with Data Poisoning

The server has two crucial components. The first one is model aggregation, which synchronously compresses and aggregates the received local model parameters. Another component is designed to produce adversarially perturbed examples which are induced by a poisoning attack algorithm for the usage of adversarial training.

4.1.1 Model Aggregation. The overall goal of federated learning is to learn a prediction function by using knowledge from all clients such that it could generalize well over the test data set. When the central server receives the parameter updates from local clients, it aggregates the locally updated parameters to obtain a shared global model. It is notable that most existing federated learning approaches [26, 27, 38] focus on developing advanced model aggregation schemes. One of the popular aggregation methods is FedAvg [26], which aims to average the element-wise parameters of local models, i.e., $w_G = \text{Aggregate}(w_1,\cdots,w_K) = \sum_{k=1}^{K}\frac{n_k}{n}w_k$ where w_k is the model parameters in the k^{th} client and $n=\sum_{k=1}^{K} n_k$. In this paper, we focus on improving the adversarial robustness of federated learning by encouraging the local model parameters to be updated with adversarial examples (see next subsection). Then the adversarial robustness of local models can be naturally propagated into the server's global model after model aggregation. Therefore, our framework is flexible to be incorporated with existing model aggregation methods, e.g., FedAvg [26], FedMA [38], AFL [27], etc.

4.1.2 Adversarial Examples. It is shown [13] that the adversarial robustness of deep neural networks can be improved by updating the model parameters over adversarial examples. That is because the generalization error of neural networks on adversarial examples can be minimized during model training. However, it will encounter a few issues if we apply adversarial training on federated learning directly. Specifically, one intuitive solution for generating adversarial examples is to separately maximize the generalization error in each local client (see Subsection 4.3 for more discussion). Its drawbacks are two folds: First, it will significantly increase the computational burden and memory usage on local clients. Second, the locally generated adversarial examples make the augmented data distributions of local clients much more biased [41], which challenges the standard server-level aggregation mechanisms [27].

Instead, we study the adversarial robustness of federated learning from the perspective of bias-variance decomposition. The following theorem shows that in the classification setting, the generalization error of a learning algorithm can be decomposed into bias, variance, and irreducible noise. Note that this decomposition holds for any real-valued loss function in the binary setting [9] with a bias & variance trade-off coefficient that has a closed-form expression.

Theorem 4.1. *In binary case, the decomposition in Eq. (1) is valid for any real-valued loss function that satisfy* $\forall_y L(y,y)=0$ *and* $\forall_{y_1\neq y_2} L(y_1,y_2)\neq 0$ *with* $\lambda=1$ *if* $y_m=y_*$ *or* $\lambda=-\frac{L(y_m,y_*)}{L(y_*,y_m)}$ *otherwise.*

The proof of Theorem 4.1 is similar to [9], we omit it here for space. Note that noise is irreducible and we usually drop this term in real applications [9, 12, 37, 43]. Commonly used loss functions for this decomposition are square loss, zero-one loss, and cross-entropy loss with one-hot labels. For the multi-class setting, the closed-form solution of coefficient λ for cross-entropy loss is given as follows.

$$\lambda=\begin{cases}1 & , \ y_m=y_*\\ -\frac{L(y_m,y_*)}{L(y_*,y_m)} & , \ y=y_m \text{ or } y=y* \text{ (given } y_m\neq y_*)\\ 0 & , \ y\neq y_m\neq y_*\end{cases} \quad (2)$$

Remark. Intuitively, when $y_m=y_*$, no bias is induced and the generalization error mainly comes from variance; when bias exists $y_m\neq y_*$, a negative variance will help to reduce error when the prediction is identical to the main prediction or optimal prediction; otherwise, the variance is set to zero. In this paper, neural networks that use cross-entropy loss and mean squared loss with softmax prediction outputs are studied. Thus, we inherit their definition of bias & variance directly, but treat the trade-off coefficient λ as a hyper-parameter to tune because no closed-form expression of λ is available in a general multi-class learning scenario.

Following [9, 37], we assume a noise-free scenario where the class label t is a deterministic function of x (i.e., if x is sampled repeatedly, the same values of its class t will be observed). The bias and variance can be empirically estimated over the clients' models. This motivates us to generate the adversarial examples by attacking the bias and variance induced by clients' models as:

$$\max_{\hat{x}\in\Omega(x)} B(\hat{x};w_1,\cdots,w_K)+\lambda V(\hat{x};w_1,\cdots,w_K) \quad \forall(x,t)\in\mathcal{D}_s \quad (3)$$

where $B(\hat{x};w_1,\cdots,w_K)$ and $V(\hat{x};w_1,\cdots,w_K)$ could be empirically estimated from a finite number of clients' parameters $w_1,\cdots,w_K$ trained on local training sets $\{\mathcal{D}_1,\cdots,\mathcal{D}_K\}$. Here λ is a hyper-parameter to measure the trade-off of bias and variance, and $\Omega(x)$ is the perturbation constraint. $\hat{x}$ is the perturbed examples w.r.t. the clean example x associated with class label t. Specifically, the perturbation constraint $\hat{x}\in\Omega(x)$ forces the adversarial example $\hat{x}$ to be visually indistinguishable w.r.t. x. In our paper, we use l_∞-bounded adversaries [13], i.e., $\Omega(x):=\{\hat{x}\big|\|\hat{x}-x\|_\infty\leq\epsilon\}$ for a perturbation magnitude ϵ.

Note that $\mathcal{D}_s$ (on the server) is the candidate subset of all available training examples that would lead to their perturbed counterparts. This is a more feasible setting as compared to generating adversarial examples on clients' devices because the server usually has powerful computational capacity in real scenarios that allows the usage of flexible poisoning attack algorithms. In this case, both poisoned examples and server model parameters would be sent back to each client (*Backward Communication*), while only clients' local parameters would be uploaded to the server (*Forward Communication*), i.e., the *asymmetrical communication.*

4.2 Robust Client Update

The robust training of one client's prediction model (i.e., w_k) can be formulated as the following minimization problem.

$$\min_{w_k}\left(\sum_{i=1}^{n_k} L(f_{\mathcal{D}_k}(x_i^k;w_k),t_i^k)+\sum_{j=1}^{n_s} L(f_{\mathcal{D}_k}(\hat{x}_j^s;w_k),t_j^s)\right) \quad (4)$$

where $\hat{x}_j^s \in \Omega(x_j^s)$ is the perturbed example that is asymmetrically transmitted from the server.

Intuitively, the bias measures the systematic loss of a learning algorithm, and the variance measures the prediction consistency of the learner over different training sets. Therefore, our robust federated learning framework has the following advantages: (i) it encourages the clients to consistently produce the optimal prediction for perturbed examples, thereby leading to a better generalization performance; (ii) local adversarial training on perturbed examples allows to learn a robust local model, and thus a robust global model could be aggregated from clients.

4.3 Discussion

Theoretically, we could have another alternative robust federated learning strategy where the perturbed training examples of each client k is generated on local devices from $\mathcal{D}_k$ instead of transmitted from the server. It is similar to [25, 36] where it iteratively synthesizes the adversarial counterparts of clean examples and updates the model parameters over perturbed training examples. Thus, each local robust model is trained individually. Nevertheless, poisoning attacks on the device will largely increase the computational cost and memory usage. Meanwhile, it only considers the client-specific loss and is still vulnerable against adversarial examples with increasing communication rounds. Both phenomena are observed in our experiments (see Fig. 2(b) and Fig. 2(c)).

5 INSTANTIATED ALGORITHM

In this section, we present the instantiated algorithm Fed_BVA.

5.1 Bias-Variance Attack

We first consider the maximization problem in Eq. (3) using bias-variance based adversarial attacks. It finds the adversarial example $\hat{x}$ (from the original example x) that produces large bias and variance values w.r.t. clients' local models. To this end, we propose two gradient-based algorithms to generate adversarial examples.

Bias-variance based Fast Gradient Sign Method (BV-FGSM): Following FGSM [13], it linearizes the maximization problem in Eq. (3) with one-step attack as follows.

$$\hat{x}_{\text{BV-FGSM}} = x + \epsilon \cdot \text{sign}\left(\nabla_x \left(B(x; w_1, \cdots, w_K) + \lambda V(x; w_1, \cdots, w_K)\right)\right) \tag{5}$$

where $\text{sign}(\cdot)$ denotes the sign function.

Bias-variance based Projected Gradient Descent (BV-PGD): PGD [25] can be considered as a multi-step variant of FGSM and might generate powerful adversarial examples. This motivated us to derive a BV-based PGD attack:

$$\begin{aligned}&\hat{x}^{l+1}_{\text{BV-PGD}}\\ &=\text{Proj}_{\Omega(x)}\left(\hat{x}^l + \epsilon\, \text{sign}\left(\nabla_{\hat{x}^l}\left(B(\hat{x}^l; w_1, \cdots, w_K) + \lambda V(\hat{x}^l; w_1, \cdots, w_K)\right)\right)\right)\end{aligned} \tag{6}$$

where $\hat{x}^l$ is the adversarial example at the l^{th} step with the initialization $\hat{x}^0 = x$ and $\text{Proj}_{\Omega(x)}(\cdot)$ projects each step onto $\Omega(x)$.

The proposed framework could b generalized to any gradient-based adversarial attack algorithms where the gradients of bias $B(\cdot)$ and variance $V(\cdot)$ w.r.t. x are tractable when estimated from finite training sets. Compared with the existing attack methods [4, 13], our loss function the adversary aims to optimize is a linear combination of bias and variance, whereas existing work mainly focused on attacking the overall classification error involving bias only.

Algorithm 1 Fed_BVA

Input: K (number of clients, with local data sets $\{\mathcal{D}_k\}_{k=1}^K$); f (learning model); L (loss function); E (number of local epochs); F (fraction of clients selected on each round); B (batch size of local client); η (learning rate); $\mathcal{D}_s$ (shared data set on server); ϵ (perturbation magnitude).
Initialization: Initialize w_G^0 and $\hat{\mathcal{D}}_s = \mathcal{D}_s$
for each round $r = 1, 2, \cdots$ **do**
  $m = \max(F \cdot K, 1)$
  $S_r \leftarrow$ randomly sampled m clients
  for each client $k \in S_r$ in parallel **do**
    Initialize k^{th} client's model with w_G^{r-1}
    $\mathcal{B} \leftarrow$ split $\mathcal{D}_k \cup \hat{\mathcal{D}}_s$ into batches of size B
    for each local epoch $i = 1, 2, \cdots, E$ **do**
      for local batch $(x, t) \in \mathcal{B}$ **do**
        $w_k^r \leftarrow w_k^r - \eta \nabla_w L\left(f_{\mathcal{D}_k}(x; w_k^r), t\right)$
      end for
    end for
    Calculate $f_{\mathcal{D}_k}(x; w_k^r)$, $\nabla_x f_{\mathcal{D}_k}(x; w_k^r)$ for $\forall x \in \mathcal{D}_s$
  end for
  for $(x, t) \in \mathcal{D}_s$ **do**
    Estimate the gradients $\nabla_x B(x)$ and $\nabla_x V(x)$
    Update $\hat{x} \in \hat{\mathcal{D}}_s$ using Eq. (5) or (6)
  end for
  $w_G^r \leftarrow$ **Aggregate**$(w_k^r | k \in S_r)$
end for
return w_G

The following theorem states that bias $B(\cdot)$ and variance $V(\cdot)$ as well as their gradients over input x could be estimated using the clients' models.

Theorem 5.1. *Assume that $L(\cdot, \cdot)$ is the cross-entropy loss function, then, the empirical estimated main prediction y_m for an input example (x, t) has the following closed-form expression: $y_m(x) = \frac{1}{K}\sum_{k=1}^K f_{\mathcal{D}_k}(x; w_k)$. Furthermore, the empirical bias and variance over an input x are estimated as $B_{CE}(x) = \frac{1}{K}\sum_{k=1}^K L(f_{\mathcal{D}_k}(x; w_k), t)$ and $V_{CE}(x) = H(y_m)$ and their corresponding gradients over x are:*

$$\nabla_x B_{CE}(x) = \frac{1}{K}\sum_{k=1}^K \nabla_x L(f_{\mathcal{D}_k}(x; w_k), t)$$

$$\nabla_x V_{CE}(x) = -\frac{1}{K}\sum_{k=1}^K \sum_{j=1}^C (\log y_m^{(j)} + 1) \cdot \nabla_x f_{\mathcal{D}_k}^{(j)}(x; w_k)$$

Proof. Following the definition of main prediction, it aims to optimize $\arg\min_{y'} -\frac{1}{n}\sum_{k=1}^n \sum_{j=1}^C \left(f_{\mathcal{D}_k}^{(j)}(x) \cdot \log y'^{(j)}\right)$ with constraints $\sum_{j=1}^C y'^{(j)} = 1$ and $\sum_{j=1}^C f_{\mathcal{D}_k}^{(j)}(x) = 1$ for all $k = 1, \cdots, n$. It can then be solved by the Lagrange multiplier. □

In the above theorem, $H(y_m) = -\sum_{j=1}^C y_m^{(j)} \log y_m^{(j)}$ is the entropy of the main prediction y_m and C is the number of classes. In addition, we also consider the case where $L(\cdot, \cdot)$ is the MSE loss function. Its main prediction y_m for an input example (x, t) has a closed-form expression which is exactly the same as the CE loss, its empirical bias and unbiased variance can only be estimated in

the following formulas:

$$B_{MSE}(x) = \left\| \frac{1}{K} \sum_{k=1}^{K} f_{\mathcal{D}_k}(x; w_k) - t \right\|_2^2$$

$$V_{MSE}(x) = \frac{1}{K-1} \sum_{k=1}^{K} \left\| f_{\mathcal{D}_k}(x; w_k) - \frac{1}{K} \sum_{k=1}^{K} f_{\mathcal{D}_k}(x; w_k) \right\|_2^2$$

and their gradients over input x are:

$$\nabla_x B_{MSE}(x) = \left(\frac{1}{K} \sum_{k=1}^{K} f_{\mathcal{D}_k}(x; w_k) - t \right) \cdot \left(\frac{1}{K} \sum_{k=1}^{K} \nabla_x f_{\mathcal{D}_k}(x; w_k) \right)$$

$$\nabla_x V_{MSE}(x) = \frac{1}{K-1} \sum_{k=1}^{K} \left(f_{\mathcal{D}_k}(x; w_k) - \frac{1}{K} \sum_{k=1}^{K} f_{\mathcal{D}_k}(x; w_k) \right) \cdot \left(\nabla_x f_{\mathcal{D}_k}(x; w_k) - \frac{1}{K} \sum_{k=1}^{K} \nabla_x f_{\mathcal{D}_k}(x; w_k) \right)$$

We observe that the empirical estimate of $\nabla_x B_{MSE}(x)$ ha higher computational complexity than $\nabla_x B_{CE}(x)$ because it involves the gradient calculation of prediction vector $f_{\mathcal{D}_k}(x; w_k)$ over the input tensor x. Besides, it is easy to show that the empirical estimate of $\nabla_x V_{MSE}(x)$ is also computationally expensive. A comparison between using CE and MSE losses is presented in Subsection 6.3.1.

5.2 Fed_BVA

We present a novel robust federated learning algorithm with our proposed bias-variance attacks, named Fed_BVA. Following the framework defined in Eq. (3) and Eq. (4), key components of our algorithm are (1) bias-variance attacks for generating adversarial examples on the server, and (2) adversarial training using poisoned server examples together with clean local examples on each client. Therefore, we optimize these two objectives by producing the adversarial examples $\hat{\mathcal{D}}_s$ and updating the local model parameters w iteratively.

The proposed algorithm is summarized in Alg. 1. Given the server's $\mathcal{D}_s$ and clients' training data $\{\mathcal{D}_k\}_{k=1}^K$ as input, the output is a robust global model on the server. In this case, the clean server data $\mathcal{D}_s$ will be shared with all the clients. First, it initializes the server's model parameter w_G and perturbed data $\hat{\mathcal{D}}_s$, and then assigns to the randomly selected clients (Steps 4-5). Next, each client optimizes its own local model (Steps 6-15) with the received global parameters w_G as well as its own clean data $\mathcal{D}_k$, and uploads the updated parameters as well as the gradients of the local model on each shared server example back to the server. At last, the server generates the perturbed data $\hat{\mathcal{D}}_s$ (Step 16-19) using the proposed bias-variance attacks with aggregations (model parameters average, bias gradients average, and variance gradients average) in a similar manner as FedAvg [26]. These aggregations can be privacy secured if additive homomorphic encryption [1] is applied.

6 EXPERIMENTS

6.1 Settings

6.1.1 Data Sets. We evaluate our proposed algorithm on four data sets: MNIST[4], Fashion-MNIST[5], CIFAR-10[6] and CIFAR-100[6]. Following [26], we consider two methods to partition the data over clients: IID and non-IID. For IID setting, the data is shuffled and uniformly partitioned into each client. For non-IID setting, data is divided into $2F \cdot K$ shards based on sorted labels, then assigns each client with 2 shards. Thereby, each client will have data with at most two classes.

	MNIST	Fashion-MNIST	CIFAR-10	CIFAR-100 (coarse)
# comms.	100	100	100	100
# clients (K)	100	100	20	20
fraction (F)	0.1	0.1	0.2	0.2
# epochs (E)	50	50	5	5
local batch (B)	64	64	128	128
# shared (n_s)	64	64	30	60
# categories	10	10	10	20

Table 1: Learning setting of Fed_BVA

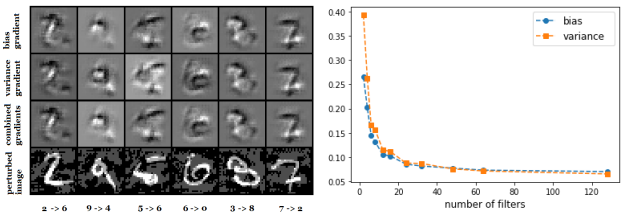
6.1.2 Baselines. The baseline models include: (1). Centralized: the training with one centralized model, which is identical to the federated learning case that only has one client ($K = 1$) with a fraction ($F = 1$). (2). FedAvg: the classical federated averaging model [26]. (3). FedAvg_AT: The simplified version of our proposed method where the local clients perform adversarial training with the asymmetrical transmitted perturbed data generated on top of FedAvg's aggregation. (4) - (6). Fed_Bias, Fed_Variance, Fed_BVA: Our proposed methods where the asymmetrical transmitted perturbed data is generated using the gradients of bias-only attack, variance-only attack, and bias-variance attack, respectively. (7). EAT[7]: Ensemble adversarial training [36], where each client performs local adversarial training, and their model updates are aggregated on the server using FedAvg. (8). EAT+Fed_BVA: A combination of baselines (6) and (7). Note that the baselines (7) and (8) have high computational requirements on client devices, and are usually not preferred in real scenarios. For fair comparisons, all baselines are modified to the asymmetrical communication setting (FedAvg and EAT have clean $\mathcal{D}_s$ received), and all their initializations are set to be the same.

6.1.3 Model Configuration. For Fed_BVA framework, we use a 4-layer CNN model for MNIST and Fashion-MNIST, and VGG9 architecture for CIFAR-10 and CIFAR-100. The training is performed using the SGD optimizer with a fixed learning rate of 0.01 and momentum of value 0.9. The trade-off coefficient between bias and variance is set to $\lambda = 0.01$ for all experiments. All hyper-parameters of federated learning are presented in Table 1. We empirically demonstrate that these hyper-parameter settings are preferable in terms of both training accuracy and robustness (see the details of Fig. 4 - Fig. 6 in Subsection 6.5). To demonstrate the robustness of our Fed_BVA framework, we evaluate the deployed server model on the test set $\mathcal{D}_{test}$ against adversarial attacks FGSM [13], PGD [25] with 10 and 20 steps (i.e., PGD-10, PGD-20). Following [36, 39], the maximum perturbations allowed are $\epsilon = 0.3$ on MNIST and Fashion-MNIST, and $\epsilon = \frac{16}{255}$ on CIFAR-10 and CIFAR-100 for both threat and defense models.

6.2 Main Results

To analyze the properties of our proposed Fed_BVA framework, we present two visualization plots on MNIST using a trained CNN model where the bias and variance are both calculated on the training examples. In Fig. 1(a), we visualize the extracted gradients using

[4] http://yann.lecun.com/exdb/mnist
[5] https://github.com/zalandoresearch/fashion-mnist
[6] https://www.cs.toronto.edu/~kriz/cifar.html
[7] Note that EAT shares the same idea as [49] for robust federated learning.



(a) Visualization (b) Bias-variance curve

Figure 1: Bias-variance analysis on MNIST with (a) visualizations of bias, variance, bias+variance, and perturbed images, and (b) Bias-variance curve w.r.t. the CNN model complexity

Method	IID			non-IID		
	Clean	FGSM	PGD-20	Clean	FGSM	PGD-20
Centralized	**0.991**$_{\pm 0.000}$	0.689$_{\pm 0.000}$	0.182$_{\pm 0.000}$	n/a	n/a	n/a
FedAvg	0.989$_{\pm 0.001}$	0.669$_{\pm 0.009}$	0.267$_{\pm 0.014}$	**0.980**$_{\pm 0.002}$	0.491$_{\pm 0.067}$	0.158$_{\pm 0.074}$
FedAvg_AT	0.988$_{\pm 0.000}$	0.802$_{\pm 0.001}$	0.512$_{\pm 0.042}$	0.974$_{\pm 0.005}$	0.649$_{\pm 0.066}$	0.363$_{\pm 0.066}$
Fed_Bias	0.986$_{\pm 0.000}$	0.812$_{\pm 0.009}$	0.583$_{\pm 0.036}$	0.971$_{\pm 0.004}$	0.679$_{\pm 0.040}$	0.394$_{\pm 0.103}$
Fed_Variance	0.985$_{\pm 0.001}$	0.803$_{\pm 0.007}$	0.572$_{\pm 0.019}$	0.973$_{\pm 0.005}$	0.684$_{\pm 0.004}$	0.395$_{\pm 0.049}$
Fed_BVA	0.986$_{\pm 0.001}$	0.818$_{\pm 0.003}$	0.613$_{\pm 0.020}$	0.969$_{\pm 0.002}$	0.705$_{\pm 0.009}$	0.469$_{\pm 0.031}$
EAT	0.981$_{\pm 0.000}$	**0.902**$_{\pm 0.001}$	0.811$_{\pm 0.004}$	0.972$_{\pm 0.002}$	0.789$_{\pm 0.016}$	0.415$_{\pm 0.035}$
EAT+Fed_BVA	0.980$_{\pm 0.001}$	0.901$_{\pm 0.006}$	**0.821**$_{\pm 0.013}$	0.965$_{\pm 0.005}$	**0.811**$_{\pm 0.020}$	**0.670**$_{\pm 0.014}$

Table 2: Accuracy of MNIST under white-box attacks in IID and non-IID settings

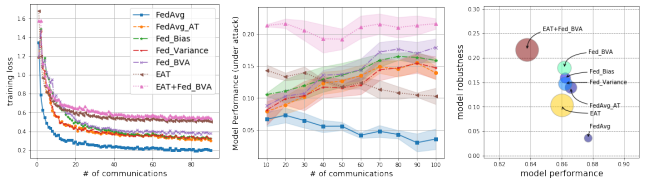
Method	IID			non-IID		
	Clean	FGSM	PGD-20	Clean	FGSM	PGD-20
Centralized	**0.882**$_{\pm 0.000}$	0.229$_{\pm 0.000}$	0.009$_{\pm 0.000}$	n/a	n/a	n/a
FedAvg	0.877$_{\pm 0.001}$	0.300$_{\pm 0.021}$	0.036$_{\pm 0.016}$	**0.804**$_{\pm 0.013}$	0.193$_{\pm 0.036}$	0.017$_{\pm 0.003}$
FedAvg_AT	0.866$_{\pm 0.001}$	0.490$_{\pm 0.021}$	0.139$_{\pm 0.011}$	0.730$_{\pm 0.023}$	0.445$_{\pm 0.065}$	0.087$_{\pm 0.042}$
Fed_Bias	0.862$_{\pm 0.001}$	0.505$_{\pm 0.015}$	0.159$_{\pm 0.003}$	0.709$_{\pm 0.025}$	0.460$_{\pm 0.038}$	0.115$_{\pm 0.054}$
Fed_Variance	0.862$_{\pm 0.002}$	0.496$_{\pm 0.012}$	0.157$_{\pm 0.017}$	0.719$_{\pm 0.036}$	0.499$_{\pm 0.081}$	**0.120**$_{\pm 0.038}$
Fed_BVA	0.862$_{\pm 0.003}$	0.528$_{\pm 0.016}$	0.180$_{\pm 0.027}$	0.710$_{\pm 0.045}$	0.495$_{\pm 0.030}$	0.093$_{\pm 0.028}$
EAT	0.860$_{\pm 0.005}$	**0.773**$_{\pm 0.029}$	0.103$_{\pm 0.013}$	0.791$_{\pm 0.012}$	0.597$_{\pm 0.033}$	0.027$_{\pm 0.023}$
EAT+Fed_BVA	0.838$_{\pm 0.009}$	0.715$_{\pm 0.011}$	**0.226**$_{\pm 0.006}$	0.735$_{\pm 0.020}$	**0.632**$_{\pm 0.015}$	0.106$_{\pm 0.039}$

Table 3: Accuracy of Fashion-MNIST under white-box attacks in IID and non-IID settings

Method	CIFAR-10			CIFAR-100		
	Clean	FGSM	PGD-20	Clean	FGSM	PGD-20
Centralized	**0.903**$_{\pm 0.003}$	0.288$_{\pm 0.001}$	0.074$_{\pm 0.005}$	**0.741**$_{\pm 0.003}$	0.166$_{\pm 0.012}$	0.032$_{\pm 0.003}$
FedAvg	0.890$_{\pm 0.002}$	0.225$_{\pm 0.022}$	0.062$_{\pm 0.008}$	0.730$_{\pm 0.003}$	0.161$_{\pm 0.009}$	0.035$_{\pm 0.006}$
FedAvg_AT	0.890$_{\pm 0.003}$	0.280$_{\pm 0.021}$	0.099$_{\pm 0.014}$	0.707$_{\pm 0.003}$	0.162$_{\pm 0.006}$	0.048$_{\pm 0.003}$
Fed_Bias	0.890$_{\pm 0.004}$	0.280$_{\pm 0.018}$	0.103$_{\pm 0.012}$	0.702$_{\pm 0.002}$	0.163$_{\pm 0.005}$	0.061$_{\pm 0.003}$
Fed_Variance	0.889$_{\pm 0.001}$	0.267$_{\pm 0.014}$	0.092$_{\pm 0.009}$	0.710$_{\pm 0.007}$	0.161$_{\pm 0.005}$	0.045$_{\pm 0.016}$
Fed_BVA	0.889$_{\pm 0.003}$	0.286$_{\pm 0.013}$	0.104$_{\pm 0.012}$	0.709$_{\pm 0.003}$	0.163$_{\pm 0.007}$	0.062$_{\pm 0.005}$
EAT	0.833$_{\pm 0.003}$	0.596$_{\pm 0.003}$	0.561$_{\pm 0.002}$	0.661$_{\pm 0.001}$	0.267$_{\pm 0.002}$	0.188$_{\pm 0.001}$
EAT+Fed_BVA	0.833$_{\pm 0.003}$	**0.598**$_{\pm 0.002}$	**0.564**$_{\pm 0.003}$	0.657$_{\pm 0.002}$	**0.272**$_{\pm 0.003}$	**0.211**$_{\pm 0.002}$

Table 4: Accuracy of CIFAR-10 and CIFAR-100 under white-box attacks

adversarial attack from bias, variance, and bias-variance. Notice that the gradients of bias and variance are similar but with subtle differences in local pixel areas. However, according to Theorem 5.1, the gradient calculation of these two are quite different: bias requires the target label as input, but variance only needs the model output and main prediction. From another perspective, we also investigate the bias-variance magnitude relationship with varying model complexity. As shown in Fig. 1(b), with increasing model complexity (more convolutional filters in CNN), both bias and variance decrease. This result is different from the double-descent curve or bell-shape variance curve claimed in [2, 43]. The reasons are



(a) Convergence (b) Performance (c) Efficiency

Figure 2: Model analysis on Fashion-MNIST (PGD-20 attack)

twofold: First, their bias-variance definitions are from the MSE regression decomposition perspective, whereas our decomposition utilizes the concept of the main prediction, and the generalization error is decomposed from the classification perspective; Second, their implementations only evaluate the bias and variance using training batches on one central model and thus is different from the definition which requires the variance to be estimated from multiple sub-models (in our scenario, client models).

The convergence plot of all baselines is presented in Fig. 2(a). We observe that FedAvg has the best convergence, and all robust training will have a slightly higher loss upon convergence. This matches the observations in [25] which states that training performance may be sacrificed in order to provide robustness for small capacity networks. For the model performance shown in Fig. 2(b), we observe that the aggregation of federated learning is vulnerable to adversarial attacks since both FedAvg and EAT have decreased performance with an increasing number of server-client communications. Other baselines that utilized asymmetrical communications have increasing robustness with more communication rounds although only a small number of perturbed examples ($n_s = 64$) are transmitted. We also observe that when communication rounds reach 40, Fed_BVA starts to outperform EAT while the latter is even more resource-demanding than Fed_BVA (shown in Fig. 2(c), where the pie plot size represents the running time). Overall, bias-variance based adversarial training via asymmetric communication is both effective and efficient for robust federated learning.

For the comprehensive experiments in Table 2 and Table 3, it is easy to verify that our proposed model outperforms all other baselines regardless of the source of the perturbed examples (i.e., locally generated like EAT+Fed_BVA or asymmetrically transmitted from the server like Fed_BVA). In this case, BV-FGSM (see Eq. (5)) is used to generate the adversarial examples $\hat{\mathcal{D}}_s$ during model training for robust federated learning. Compared to standard robust federated learning FedAvg_AT, the performance of Fed_BVA against adversarial attacks still increases 4% – 13% and 2% – 9% on IID and non-IID settings respectively, although Fed_BVA is theoretically suitable for the cases that clients have IID samples. In Table 4, we observe a similar trend where Fed_BVA outperforms FedAvg_AT on CIFAR-10 and CIFAR-100 (with 0.2% – 10% increases) when defending different types of adversarial examples. Compared to the strong local adversarial training baseline EAT, we also observe a maximum 13% accuracy increase when applying its bias-variance oriented baseline EAT+Fed_BVA. Overall, the takeaway is that without local adversarial training, using a bias-variance based robust learning framework could outperform other baselines for defending FGSM and PGD attacks. When local adversarial training is allowed (e.g., the client device has powerful computation ability), using bias-variance robust learning with local adversarial training will mostly have the best robustness.

Loss	Clean	Fed_BVA		
		BiasOnly	VarianceOnly	BVA
CE	$0.588_{(38.13s)}$	$\mathbf{0.763}_{(47.58s)}$	$\mathbf{0.759}_{(63.46s)}$	$\mathbf{0.776}_{(63.67s)}$
MSE	$\mathbf{0.601}_{(39.67s)}$	$0.711_{(65.03s)}$	$0.711_{(162.40s)}$	$0.712_{(179.60s)}$

Table 5: Accuracy of training with different loss functions and running time (second/epoch)

Method	IID			non-IID		
	FGSM	PGD-10	PGD-20	FGSM	PGD-10	PGD-20
FedAvg	0.588	0.620	0.205	0.147	0.525	0.089
$\text{Fed\_BVA}_{(\text{BV-FGSM})}$	**0.776**	0.793	0.570	**0.670**	0.695	0.472
$\text{Fed\_BVA}_{(\text{BV-PGD})}$	0.757	**0.840**	**0.632**	0.659	**0.784**	**0.575**

Table 6: Comparison of training with BV-PGD and BV-FGSM

Strategy	IID			non-IID		
	FGSM	PGD-10	PGD-20	FGSM	PGD-10	PGD-20
S1	**0.745**	**0.743**	**0.450**	**0.529**	0.677	**0.433**
S2	0.743	0.731	0.436	0.513	**0.680**	0.432
S3	0.730	0.704	0.400	0.495	0.657	0.380

Table 7: Robust training with different strategies on MNIST

6.3 Ablation Studies

6.3.1 MSE loss v.s. CE loss. Both cross-entropy (CE) and mean squared error (MSE) loss functions could be used for training a neural network model. In our paper, the loss function of neural networks determines the derivation of bias and variance terms used for producing the adversarial examples. We experimentally compare the CE and MSE loss functions in our framework. Table 5 reports the adversarial robustness of our federated framework w.r.t. FGSM attack ($\epsilon = 0.3$) on MNIST with IID setting. It is observed that (1) our framework with MSE has a significantly larger running time; (2) the robustness of our framework with MSE becomes slightly weaker, which might be induced by the weakness of MSE in training neural networks under the classification setting.

6.3.2 BV-PGD v.s. BV-FGSM. Our bias-variance attack could be naturally generalized to any gradient-based adversarial attack algorithms when the gradients of bias $B(\cdot)$ and variance $V(\cdot)$ w.r.t. x are tractable to be estimated from clients' models learned on finite training sets. Here, we empirically compare the adversarial robustness of the proposed framework trained with bias-variance guided PGD (BV-PGD) adversarial examples and bias-variance guided FGSM (BV-FGSM) adversarial examples. Table 6 provides our results on w.r.t. FGSM and PGD attacks ($\epsilon = 0.3$) on MNIST with IID and non-IID settings. Compared to FedAvg, our framework Fed_BVA with either BV-FGSM or BV-PGD could largely improve the model robustness against adversarial noise. Furthermore, BV-PGD could potentially improve white-box robustness on multi-step attacks, but it is often more computationally demanding. As a comparison, BV-FGSM is more robust against single-step attacks.

6.3.3 Alternative Training Strategies for Fed_BVA. Our `Fed_BVA` framework maximizes the overall generalization error induced by bias and variance from different clients for the adversarial examples generation. Under this setting, the generated adversarial examples on the server are shared with all the clients for local adversarial training. In particular, we found that when using the CE loss, the estimated gradients of bias and variance can be considered as the average of clients' local gradients over input x (see Subsection 5.1). This motivates us to consider several alternative training strategies by generating client-specific adversarial examples on the server.

CIFAR-10	**Source** (FGSM attack)				**Source** (PGD-20 attack)			
Target	R	V	X	M	R	V	X	M
FedAvg	0.707	0.688	0.689	0.793	0.611	0.623	0.597	0.787
FedAvg_AT	0.742	0.710	0.720	0.808	0.695	0.670	0.661	0.808
Fed_Bias	0.740	0.703	0.715	0.799	0.690	0.667	0.654	0.799
Fed_Variance	0.738	0.704	0.719	0.810	0.677	0.656	0.648	0.809
Fed_BVA	0.744	0.706	0.722	0.809	0.693	0.669	0.664	0.809
EAT	0.821	0.806	0.815	0.823	0.819	0.808	**0.813**	0.822
EAT+Fed_BVA	**0.828**	**0.808**	**0.817**	**0.828**	**0.825**	**0.809**	0.812	**0.829**

Table 8: CIFAR-10 accuracy towards black-box adversarial examples on transferability between models (R: ResNet18; V: VGG11; X: Xception; M: MobileNetV2)

- **S1:** We generate the adversarial examples $\hat{\mathcal{D}}_s$ to maximize the bias and variance from all clients' predictions. In this case, the generated adversarial examples on the server will be shared with the local clients. This is the strategy used in our `Fed_BVA` algorithm. It guarantees the minimization of generalization error from the perspective of bias-variance decomposition, thus leading to an adversarially robust federated learning model.
- **S2:** The bias-variance decomposition with CE loss indicates that we generate the client-specific adversarial examples as

$$\nabla_x B_k(x; w_k) = \nabla_x L(f_{\mathcal{D}_k}(x; w_k), t)$$

$$\nabla_x V_k(x; w_k) = \sum_{j=1}^{C} (\log y_m^{(j)} + 1) \nabla_x f_{\mathcal{D}_k}^{(j)}(x; w_k)$$

  where $x \in \mathcal{D}_s$ and $k \in \{1, ..., K\}$. If we using FGSM for attacking, the adversarial example on the k^{th} client is:

$$\hat{x}^k_{\text{BV-FGSM}} := x + \epsilon \cdot \text{sign}\left(\nabla_x \left(B_k(x; w_k) + \lambda V_k(x; w_k)\right)\right)$$

  Note that the gradient estimate of client-specific variance also relies on the main prediction y_m. But in this case, it allows every client to have different adversarial examples $\hat{\mathcal{D}}_s$. Intuitively, this training strategy further decomposes the bias and variance into individual client-specific terms.
- **S3:** Another training strategy is to use every local client model to generate the adversarial examples on the server individually as follows.

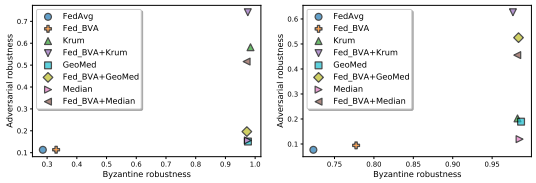
$$\nabla_x B_k(x; w_k) = \nabla_x L(f_{\mathcal{D}_k}(x; w_k), t) \quad \forall x \in \mathcal{D}_s \quad k \in \{1, ..., K\}$$

  Similarly, we can have:

$$\hat{x}^k_{\text{BV-FGSM}} := x + \epsilon \cdot \text{sign}\left(\nabla_x B_k(x; w_k)\right)$$

  It is a special case of **S2** with $\lambda = 0$. In this case, every client will only use its own model parameters to generate the client-specific adversarial examples on the server.

We conduct the ablation study to compare different training strategies in our `Fed_BVA` framework. In this case, we use $K = 10$ clients with fraction $F = 1$ and local epoch $E = 5$. Other hyper-parameters and model architecture settings are the same as in our previous experiments. Table 7 provides the performance of adversarial robustness using our `Fed_BVA` framework on MNIST with both IID and non-IID settings. It is observed that `Fed_BVA` with **S1** has the best robustness in most cases compared to other heuristic training strategies **S2** and **S3**. This indicates that bias and variance provide better direction to generate the adversarial examples for robust training. In contrast, detecting the adversarial examples with individual directions in **S2** for each client might be sub-optimal.



(a) FGSM & Sign-flipping (b) PGD-20 & Additive noise

Figure 3: Adversarial and Byzantine robustness on MNIST with (a) FGSM attack on the test set and sign-flipping attack on local model updates, and (b) PGD-20 attack on the test set and additive noise attack on local model updates

6.4 Against Other Attacks

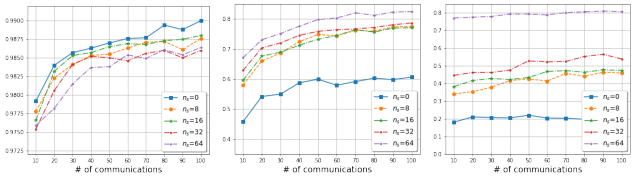
6.4.1 Black-box Attacks. Using black-box attack, we test the transferability of single-step attack (i.e., FGSM [13]) and multi-step attack (i.e., PGD [21]) on various federated learning baseline models. For CIFAR-10 data set, we pretrained ResNet18 [15], VGG11 [33], Xception [7], and MobileNetV2 [31] as the source threat models for generating the single-step and multi-step adversarial examples. The black-box transfer attacking results are shown in Table 8. We observe that without any adversarial training, FedAvg will suffer a maximum of 28% accuracy lost. For comparison, the robust federated learning model with global transmitted perturb samples (i.e., FedAvg_AT, Fed_Bias, Fed_Variance, Fed_BVA) will have increase robustness with a maximum of 23% accuracy drop on best baselines. For the computation-demanding local robust training methods (i.e., EAT and EAT+Fed_BVA), the maximum accuracy drop is only 3%, respectively. Thus, it is straightforward to see that CIFAR-10 is more vulnerable to multi-step black-box adversarial attacks, but the local adversarial training methods could improve its robustness.

6.4.2 Byzantine Attacks. The proposed Fed_BVA is flexible to incorporate with Byzantine-robust aggregation variants, in order to improve both the adversarial robustness against the corrupted test data set and the Byzantine robustness against the corrupted local model updates. Here we use sign-flipping attack [23] and additive noise attack [23] to manipulate the local model updates of 40% clients. Then we adopt the Byzantine-robust aggregation mechanisms in our Algorithm 1. In this case, we use three popular mechanisms, i.e., Krum [3], Median [44] and GeoMed [6], to aggregate the local model updates. We report the results in Figure 3. It is observed that Fed_BVA is robust against adversarial and Byzantine attacks, when using Krum as our aggregation strategy.

6.5 Parameter Analysis

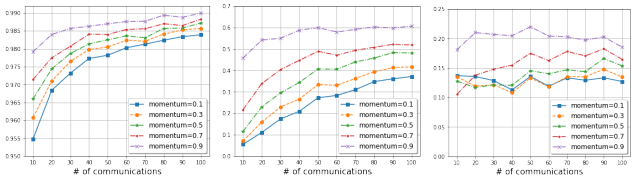
6.5.1 Number of shared perturbed samples n_s. From Fig. 4, we see that the accuracy of FedAvg (i.e., $n_s = 0$) has the best accuracy as we expected. For Fed_BVA with varying size of asymmetrical transmitted perturbed samples (i.e., $n_s = 8, 16, 32, 64$), its performance drops slightly with increasing n_s (on average drop of 0.05% per plot). As a comparison, the robustness on the test set increases dramatically with increasing n_s (the improvement ranges from 18% to 22% under FGSM attack and ranges from 15% to 60% under PGD-20 attack). However, choosing large n_s would have high model robustness but also suffer from the high communication cost. In our experiments, we choose $n_s = 64$ for MNIST for the ideal trade-off point.

6.5.2 Momentum. We also care about the choice of options in the SGD optimizer. Fig. 5(a) shows that the accuracy of clean training



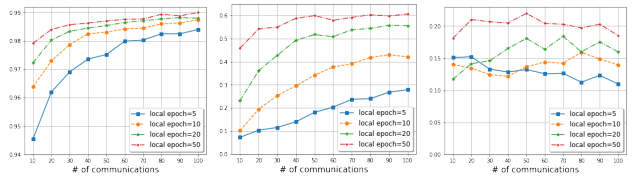
(a) Clean training (b) Under FGSM attack (c) Under PGD-20 attack

Figure 4: Accuracy of Fed_BVA w.r.t. number of shared data n_s



(a) Clean training (b) Under FGSM attack (c) Under PGD-20 attack

Figure 5: Accuracy of Fed_BVA w.r.t. momentum



(a) Clean training (b) Under FGSM attack (c) Under PGD-20 attack

Figure 6: Accuracy of Fed_BVA w.r.t. local epoch

monotonically increases when the momentum is varying from 0.1 to 0.9. Interestingly, we observe from Fig. 5(b) and Fig. 5(c) that the federated learning model is less vulnerable when momentum is large no matter whether the adversarial attack is FGSM or PGD-20 on $\mathcal{D}_{test}$. So we choose the momentum as 0.9 for our experiments.

6.5.3 Local epochs E. Another important factor of federated learning is the number of local epochs. In Fig. 6, we see that more local epochs in each client lead to a more accurate aggregated server model in prediction. Its robustness against both FGSM and PGD-20 attacks on the test set $\mathcal{D}_{test}$ also has the best performance when the local epochs are large on the device. Hence, in our experiments, if the on-device computational cost is not very high (large data example size, deep models with many layers), we choose $E = 50$. Otherwise, we will reduce E to a smaller number accordingly.

7 CONCLUSION

In this paper, we proposed a novel robust federated learning framework, in which the loss incurred during the server's aggregation is dissected into a bias part and a variance part. Our approach improves the model robustness through adversarial training by supplying a few bias-variance perturbed samples to the clients via asymmetrical communications. Extensive experiments have been conducted where we evaluated its performance from various aspects on several benchmark data sets.

ACKNOWLEDGMENTS

This work is supported by National Science Foundation under Award No. IIS-1947203, IIS-2117902, IIS-2137468, and Agriculture and Food Research Initiative (AFRI) grant no. 2020-67021-32799/project accession no.1024178 from the USDA National Institute of Food and Agriculture. The views and conclusions are those of the authors and should not be interpreted as representing the official policies of the funding agencies or the government.

REFERENCES

[1] Abbas Acar, Hidayet Aksu, A Selcuk Uluagac, and Mauro Conti. 2018. A survey on homomorphic encryption schemes: Theory and implementation. *Comput. Surveys* 51, 4 (2018), 1–35.

[2] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. 2019. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences* 116, 32 (2019), 15849–15854.

[3] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. 2017. Machine learning with adversaries: Byzantine tolerant gradient descent. *Advances in Neural Information Processing Systems* 30 (2017).

[4] Nicholas Carlini and David Wagner. 2017. Towards evaluating the robustness of neural networks. In *2017 IEEE symposium on security and privacy*. IEEE, 39–57.

[5] Liwei Che, Zewei Long, Jiaqi Wang, Yaqing Wang, Houping Xiao, and Fenglong Ma. 2021. FedTriNet: A Pseudo Labeling Method with Three Players for Federated Semi-supervised Learning. In *2021 IEEE International Conference on Big Data*. IEEE, 715–724.

[6] Yudong Chen, Lili Su, and Jiaming Xu. 2017. Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 1, 2 (2017), 1–25.

[7] François Chollet. 2017. Xception: Deep Learning with Depthwise Separable Convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1251–1258.

[8] Deepesh Data and Suhas Diggavi. 2021. Byzantine-resilient high-dimensional SGD with local iterations on heterogeneous data. In *International Conference on Machine Learning*. 2478–2488.

[9] Pedro Domingos. 2000. A unified bias-variance decomposition and its applications. In *International Conference on Machine Learning*. 231–238.

[10] Dongqi Fu, Liri Fang, Ross Maciejewski, Vetle I. Torvik, and Jingrui He. 2022. Meta-Learned Metrics over Multi-Evolution Temporal Graphs. In *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*. ACM, 367–377.

[11] Dongqi Fu, Zhe Xu, Bo Li, Hanghang Tong, and Jingrui He. 2020. A View-Adversarial Framework for Multi-View Network Embedding. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*. ACM, 2025–2028.

[12] Stuart Geman, Elie Bienenstock, and René Doursat. 1992. Neural networks and the bias/variance dilemma. *Neural computation* 4, 1 (1992), 1–58.

[13] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*.

[14] Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. 2018. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604* (2018).

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[16] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. 2017. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *CoRR* abs/1704.04861 (2017).

[17] Wonyong Jeong, Jaehong Yoon, Eunho Yang, and Sung Ju Hwang. 2021. Federated Semi-Supervised Learning with Inter-Client Consistency & Disjoint Learning. In *International Conference on Learning Representations*.

[18] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. 2020. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*. 5132–5143.

[19] Ron Kohavi, David H Wolpert, et al. 1996. Bias plus variance decomposition for zero-one loss functions. In *International Conference on Machine Learning*. 275–83.

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

[21] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2017. Adversarial machine learning at scale. In *International Conference on Learning Representations*.

[22] Rui Li, Fenglong Ma, Wenjun Jiang, and Jing Gao. 2019. Online Federated Multitask Learning. In *2019 IEEE International Conference on Big Data*. IEEE, 215–220.

[23] Suyi Li, Yong Cheng, Wei Wang, Yang Liu, and Tianjian Chen. 2020. Learning to detect malicious clients for robust federated learning. *arXiv preprint arXiv:2002.00211* (2020).

[24] Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. 2021. Ditto: Fair and robust federated learning through personalization. In *International Conference on Machine Learning*. 6357–6368.

[25] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards Deep Learning Models Resistant to Adversarial Attacks. In *International Conference on Learning Representations*.

[26] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*. PMLR, 1273–1282.

[27] Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. 2019. Agnostic federated learning. In *International Conference on Machine Learning*. PMLR, 4615–4625.

[28] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. 2016. DeepFool: A simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2574–2582.

[29] Venkata Krishna Pillutla, Sham M. Kakade, and Zaïd Harchaoui. 2019. Robust Aggregation for Federated Learning. *CoRR* abs/1912.13445 (2019).

[30] Amirhossein Reisizadeh, Farzan Farnia, Ramtin Pedarsani, and Ali Jadbabaie. 2020. Robust federated learning: The case of affine distribution shifts. *Advances in Neural Information Processing Systems* 33 (2020), 21554–21565.

[31] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 2018. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4510–4520.

[32] Devansh Shah, Parijat Dube, Supriyo Chakraborty, and Ashish Verma. 2021. Adversarial training in communication constrained federated learning. *arXiv preprint arXiv:2103.01319* (2021).

[33] Karen Simonyan and Andrew Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *International Conference on Learning Representations*.

[34] Sebastian U. Stich. 2019. Local SGD Converges Fast and Communicates Little. In *International Conference on Learning Representations*.

[35] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna Estrach, Dumitru Erhan, Ian Goodfellow, and Robert Fergus. 2014. Intriguing properties of neural networks. In *International Conference on Learning Representations*.

[36] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick Drew McDaniel. 2018. Ensemble adversarial training: Attacks and defenses. In *International Conference on Learning Representations*.

[37] Giorgio Valentini and Thomas G Dietterich. 2004. Bias-variance analysis of support vector machines for the development of SVM-based ensemble methods. *Journal of Machine Learning Research* (2004), 725–775.

[38] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. 2020. Federated Learning with Matched Averaging. In *International Conference on Learning Representations*.

[39] Yisen Wang, Xingjun Ma, James Bailey, Jinfeng Yi, Bowen Zhou, and Quanquan Gu. 2019. On the Convergence and Robustness of Adversarial Training. In *International Conference on Machine Learning*.

[40] Jun Wu and Jingrui He. 2021. Indirect Invisible Poisoning Attacks on Domain Adaptation. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 1852–1862.

[41] Cihang Xie and Alan Yuille. 2020. Intriguing Properties of Adversarial Training at Scale. In *International Conference on Learning Representations*.

[42] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)* 10, 2 (2019), 1–19.

[43] Zitong Yang, Yaodong Yu, Chong You, Jacob Steinhardt, and Yi Ma. 2020. Rethinking bias-variance trade-off for generalization of neural networks. In *International Conference on Machine Learning*.

[44] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. 2018. Byzantine-robust distributed learning: Towards optimal statistical rates. In *International Conference on Machine Learning*. PMLR, 5650–5659.

[45] Lecheng Zheng, Yu Cheng, Hongxia Yang, Nan Cao, and Jingrui He. 2021. Deep co-attention network for multi-view subspace learning. In *Proceedings of the Web Conference 2021*. 1528–1539.

[46] Dawei Zhou, Lecheng Zheng, Jiawei Han, and Jingrui He. 2020. A Data-Driven Graph Generative Model for Temporal Interaction Networks. In *The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, 401–411.

[47] Dawei Zhou, Lecheng Zheng, Jiejun Xu, and Jingrui He. 2019. Misc-GAN: A multi-scale generative model for graphs. *Frontiers in big Data* 2 (2019), 3.

[48] Yao Zhou, Jianpeng Xu, Jun Wu, Zeinab Taghavi Nasrabadi, Evren Körpeoglu, Kannan Achan, and Jingrui He. 2021. PURE: Positive-Unlabeled Recommendation with Generative Adversarial Network. In *The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, 2409–2419.

[49] Giulio Zizzo, Ambrish Rawat, Mathieu Sinn, and Beat Buesser. 2020. FAT: Federated adversarial training. *arXiv preprint arXiv:2012.01791* (2020).