Dynamic Defense Against Byzantine Poisoning Attacks in Federated Learning

Nuria Rodríguez-Barroso[a,*], Eugenio Martínez-Cámara[a], M. Victoria Luzón[b], Francisco Herrera[a]

[a] *Department of Computer Science and Artificial Intelligence, AndalusianResearch Institute in Data Science and Computational Intelligence (DaSCI), University of Granada, 18071 Granada, Spain*

[b] *Department of Software Engineering, Andalusian Research Institute inData Science and Computational Intelligence (DaSCI), University of Granada, 18071 Granada, Spain*

Abstract

Federated learning, as a distributed learning that conducts the training on the local devices without accessing to the training data, is vulnerable to Byzatine poisoning adversarial attacks. We argue that the federated learning model has to avoid those kind of adversarial attacks through filtering out the adversarial clients by means of the federated aggregation operator. We propose a dynamic federated aggregation operator that dynamically discards those adversarial clients and allows to prevent the corruption of the global learning model. We assess it as a defense against adversarial attacks deploying a deep learning classification model in a federated learning setting on the Fed-EMNIST Digits, Fashion MNIST and CIFAR-10 image datasets. The results show that the dynamic selection of the clients to aggregate enhances the performance of the global learning model and discards the adversarial and poor (with low quality models) clients.

Keywords: Federated Learning, Deep Learning, adversarial attacks, byzantine attacks, dynamic aggregation operator

1. Introduction

The standard machine learning approach is built upon an algorithm that learns from a centralized data source. Distributed machine learning proposes the distribution of the data and elements of a learning model among several nodes as a solution for the unceasing growing of learning model complexity and the size of training data [1, 2]. However, the distributed machine learning solution is neither valid for the data privacy challenge, nor for an scenario with a large number of clients and a non homogeneous data distribution [3, 4].

Federated learning (FL) is a machine learning approach in which the algorithms learn from sequestered data [3, 5]. The FL model is mainly composed of two components: a global server that owns the global learning model and a set of clients storing the local learning models and the local training datasets. Likewise, FL consists in: (1) training the local learning models in each data source, (2) distilling the parameters of the local learning models into a central server, (3) aggregating the parameters of the local models in the global learning model and (4) updating the local learning models with the aggregated federated global learning model after the aggregation. This specific setting supports its main feature, which is the prevention of data leakage and the protection of data privacy, since the data do not abandon its local storage and they are not shared with any other client or third party. Since FL is a user privacy-preserving approach designed to decentralized scenarios, an Artificial Intelligence of Things (AIoT) setting is a natural way to use it, for both the distributed nature and the privacy needed in IoT (Internet of Things) devices [6].

Machine learning is vulnerable to malicious manipulations on the input data or the learning model to cause incorrect classification [7]. This vulnerability becomes harder to address in FL due to most of the defensive approaches are based data inspection techniques. Among the different kind of adversarial attacks in the literature [8], in this paper we focus on byzantine poisoning attacks [9], which are based on the arbitrary manipulation of the training data (data poisoning attack [10, 11]) or the client model updates (model poisoning attacks [12]) with the aim of hindering the performance of the FL model.

We argue in this paper that the FL model has to be able to dynamically avoid adversarial clients to preserve the learning model from byzantines poisoning attacks, which is usually performed on the server by the federated aggregation operator. In the literature there are a number of federated aggregation operators, but they do not prevent the federated model from this kind of attacks [13, 14, 15], or they do it following some assumptions about the nature of the adversarial clients [16] or prove to be insufficiently effective [17].

We propose the Dynamic Defense Against Byzantine Attacks (DDaBA), which is a dynamic aggregation operator that dynamically selects the clients to be aggregated and discards those ones considered as adversarial, and it features agnostic about the number and nature of the adversarial clients. This dynamic defense is built upon an Induced Ordered Weighted Averaging (IOWA) operator [18], which aggregates the clients on a weighted basis according to an induced-ordered function and a linguistic quantifier. We use as induced-ordered function the performance of the local learning models on a validation

[*] Corresponding author

Email addresses: rbnuria@ugr.es (Nuria Rodríguez-Barroso), emcamara@decsai.ugr.es (Eugenio Martínez-Cámara), luzon@ugr.es (M. Victoria Luzón), herrera@decsai.ugr.es (Francisco Herrera)

set stored in the server. The linguistic quantifier addresses the weighting of the clients, which usually depends on the knowledge of the problem and predefined parameters. We design an agnostic linguistic quantifier on the nature of the problem, which is based on: (1) considering the distribution of data resulting from measuring the performance variation between local learning models on the validation set, (2) assuming that the resulting distribution follows an exponential distribution, and (3) using the properties of that distribution to set the parameters of the linguistic quantifier in order to discard the adversarial clients that correspond to outliers in the exponential distribution according to the Tukey criteria.

We evaluate the DDaBA as a defense in a FL model for image classification. For that purpose, we leverage the benchmark image classification datasets Fed-EMNIST[1] Digits [19], Fashion MNIST[2] [20] and CIFAR-10,[3] and we distribute the data over the clients following a non independent and identically distributed (non-IID) distribution. We compare the DDaBA with the classical federated aggregation operator FedAvg [13] with no defense and the state-of-the-art defenses against three different byzantine attacks: label-flipping [21], out-of-distribution [22] and random weights [23] attacks. We show that the DDaBA is able to identify the adversarial and poor clients, filter them out and enhance the performance of the global learning model.

We analyze the behavior of the DDaBA in an scenario with a extreme proportion of adversarial clients, and we see that the performance of the federated global model is hindered. Although this is a very unlikely scenario, we also introduce the static version of DDaBA, Static Defense Against Byzantine Attacks (SDaBA), which predefine the parameters of the linguistic quantifier of the IOWA operator for discarding the susceptible adversarial clients. The SDaBA, as well as the DDaBA, outperforms all the baselines in the three adversarial attacks developed for the evaluation.

The rest of the work is organized as follows: the following section summarizes the background related to FL, adversarial attacks in FL and defenses against them. Section 3 is focused on the description of the dynamic FL model for identifying adversarial clients. We detail the experimental set-up in Section 4 and evaluate and analyze the results of the FL models in Section 5. Finally, conclusions are described in Section 6.

2. Background

We expound in this section some relevant concepts and related works. We introduce FL in Section 2.1, we describe the main types of adversarial attacks in FL in Section 2.2, and we detail the proposed defenses against byzantine attacks in Section 2.3.

2.1. Federated Learning

FL is a learning approach pushed by the need of overcoming the limitations of distributed learning for preserving data privacy and for processing large number of clients following a non homogeneous data distribution [24]. FL proposes a new training approach of learning algorithms that consists in the iterative training of the model in the devices that own the data, the aggregation of those models in the federated model, and the updating of the local models with the federated model. Hence, FL prevents from data leakage and preserves data privacy, since the data do not leave the electronic device.

Formally, FL is a distributed machine learning paradigm consisting of a set of clients $\{C_1, \ldots, C_n\}$ with their respective local training data $\{D_1, \ldots, D_n\}$. Each of these clients C_i has a local learning model named as L_i represented by the parameters $\{L_1, \ldots, L_n\}$. FL aims at learning a global learning model represented by G, using the scattered data across clients through an iterative learning process known as *round of learning*. For that purpose, in each round of learning t, each client trains its local learning model over their local training data D_i^t, which updates the local parameters L_i^t to $\hat{L}_i^t$. Subsequently, the global parameters G^t are computed aggregating the trained local parameters $\{\hat{L}_1^t, \ldots, \hat{L}_n^t\}$ using an specific federated aggregation operator Δ, and the local learning models are updated with the aggregated parameters:

$$\begin{aligned} G^t &= \Delta(\hat{L}_1^t, \hat{L}_2^t, \ldots, \hat{L}_n^t) \\ L_i^{t+1} &\leftarrow G^t, \quad \forall i \in \{1, \ldots, n\} \end{aligned} \tag{1}$$

The updates among the clients and the server are repeated as much as needed for the learning process. Thus, the final value of G will sum up the knowledge sequestered in the clients.

2.2. Related works about adversarial attacks

Machine learning is highly susceptible to adversarial attacks [25], and the vast majority of the defensive approaches are based on three approaches [8]: (1) game theory [26], (2) data sanitation [27] and (3) resilient and robust learning models, which assume that a fraction of the training data may be manipulated and consider it as outliers [28]. Due to the federated aggregation operator is agnostic in relation with adversarial clients information, the first approach can not be applied in FL. Likewise, since the training data in FL is inaccessible by the server, the second approach is also not feasible in FL. Therefore, the most promising defense approach is developing resilient and robust federated aggregation operators with the ability to safeguard the model from the effect of attacks.

According to [29], there are two types of adversarial attacks in FL: (1) *Inference attacks* [30], which aim at inferring information from the training data; and (2) *poisoning attacks* [31], which pursue to compromise the global learning model. Concerning inference attacks, there are different types of them depending on the information being inferred. The most important ones are the property and membership inference attacks, which respectively seek to infer certain properties of the data and the membership of specific samples in the training set. Because of

[1] `https://www.nist.gov/node/1298471/emnist-dataset`
[2] `https://github.com/zalandoresearch/fashion-mnist`
[3] `https://www.cs.toronto.edu/~kriz/cifar.html`

their nature, the defenses proposed in the literature are based on applications derived from or inspired by the Differential Privacy [32]. Regarding poisoning attacks, we identify two taxonomies:

1 Depending on which part of the FL model is attacked, we differentiate between *model-poisoning* [33] and *data-poisoning attacks* [34]. In practice, both are almost equivalent, since a poisoning of the data results in a poisoned model. However, data-poisoning attacks and some of the model-poisoning attacks fail to be effective since the attack dissipates in the aggregation of many clients. Hence, these attacks are combined with *model-replacement* [17] techniques, which boosts the adversarial model (or models) in order to replace the global model.

2 Depending on the purpose of the attack, we distinguish between *untargeted or byzantine attacks* [35], which seek to affect the model's performance, and *targeted or backdoor attacks* [17], which aim at injecting a secondary or backdoor task into the global model by stealth.

2.3. Defenses against adversarial attacks

The literature provides multiple solutions to both byzantine and backdoor attacks in classical machine learning. The vast majority of these defenses are based on data inspection methods, such as removing outliers from the training data in centralized learning [36] or, in a distributed setting, removing outliers from participant's training data or models [37, 38]. In both cases, the available defenses require data inspection, which is not possible in FL. Therefore, defenses against adversarial attacks in FL must be designed ad hoc.

Regarding the state-of-the-art defenses designed to be applied in federated settings, they are based on the modification of the aggregation operator, because the attack is usually carried out by the clients. The most important defenses against byzantine attacks are based on a more robust aggregation of the updates and they are called *byzantine-robust aggregation rules*. We highlight the following ones:

- Coordinate-wise aggregations [39], which replaces the mean of the classical aggregation operator FedAvg [13] with more robust statistics to outliers or anomalous data. The main ones are the trimmed-mean and the median.

- Krum (and MultiKrum) [40], which is based on using geometric properties to determine the most central model updates vectors. This defense requires a k hyper-parameter that determines the number of clients remaining in the aggregation.

- Bulyan [41] which is the state of the art. It is built as a combination of Krum and trimmed-mean. Accordingly, the model updates vectors are sorted according to their geometrical centrality and are aggregated through a trimmed-mean with a m parameter, which discards a total of $2m$ clients.

Additionally, differential privacy [32] methods are an important safeguard for the information shared during the communication between the server and the clients. Therefore, the defensive challenges of the FL should focus on client attacks.

The main weakness of the defenses proposed in the literature is that they are highly dependent on parameters, which beforehand are difficult to set without information about the number or nature of the adversary clients. Thus, we propose in this paper a defense mechanism against poisoning attacks, which dynamically selects the clients that are not adversarial and filters out the adversarial or the poor ones (clients with low quality models) without the requirement to set any parameters.

3. Dynamic Defense Against Poisoning Attacks

FL is featured by its restriction to access to the training data, which is sequestered in the clients. Accordingly, poisoning attacks, both data and (local) model poisoning [10, 11], grounded in the malicious manipulation of the training data or the local model updates, can corrupt the FL model, which cannot inspect the training to defend itself against this kind of adversarial attacks.

We propose a defense against byzantine poisoning attacks built upon a federated aggregation operator based on a Induced Ordered Weighted Averaging (IOWA) [18] that dynamically selects the clients to be aggregated, and filters out the adversarial ones. We call it Dynamic Defense Against Byzantine Attacks (DDaBA).

The IOWA operators, and more generally the Ordered Weighted Averaging (OWA) ones [42], are functions for weighting the contribution of a set of clients in a aggregation process, as it is the aggregation of the parameters of the local learning models in FL. We mathematically introduce OWA and IOWA operators in Appendix Appendix A, and according to the definition the IOWA operator is composed of (1) an order-inducing function to set the weighting assignation order, and (2) a linguistic quantifier to calculate the weight contribution value. We define the induced-order function used in DDaBA in Section 3.1, and the linguistic quantifier that dynamically adapts the weighting value calculation during the FL training in Section 3.2. Finally, we sum up DDaBA in Section 3.3.

3.1. Accuracy-based induced ordering function for clients model updates

The aim of byzantine poisoning adversarial attacks is hindering the performance of a FL model through altering the training data or directly the model updates. Since FL is grounded in the aggregation of the L_i, those maliciously altered ones would perform lower than the non-altered ones. Hence, the validation of the L_i before the aggregation may help to identify the suspicious adversarial clients.

We propose the Local Accuracy Function, f_{LA}, to measure the performance of each L_i before its aggregation. The f_{LA} function is based on the availability of a validation set shared among the clients. The viability of this validation set is justified by its reduced size compared to the size required for training, and the possibility of making it up through expert or prior knowledge. We define the f_{LA} function in Definition 3.1.

Definition 3.1 (Local Accuracy Function (f_{LA}))**.** *it measures the performance of a local learning model L_i using a fixed validation dataset named as VD. For that, it computes the accuracy of L_i over VD:*

$$f_{LA}(L_i) = accuracy(L_i, VD) \tag{2}$$

where $accuracy(L_i, VD)$ refers to the standard accuracy evaluation measure of the local learning model L_i in the dataset VD.

Once the clients model updates are sorted according to this sorting function, we expect that the benign client's models will converge to a common solution, while the adversarial client's models will not, but they will converge to a worse solution for the original problem. Therefore, if we define the random variable resulting from the differences in accuracy among all clients with the client that scored the highest accuracy as follows:

$$\mathbb{X}_i^{f_{LA}} = \max_i\{f_{LA}(L_i)\} - f_{LA}(L_i). \tag{3}$$

We assume that this random variable $\mathbb{X}$ will approximate an Exponential Distribution, since there will be many values close to zero (and always positive), and very few far from zero.

3.2. Dynamic linguistic quantifier for weighting the contribution of clients

The non-IID data distribution of most of the FL settings make impossible to know beforehand the nature of the clients, and hence it is impossible to know the amount of adversarial clients. Therefore, the selection of the FL clients by its weighted contribution has to be dynamically calculated for adapting to the nature of the clients.

The dynamic selection of the DDaBA model is based on a IOWA linguistic quantifier that some of its parameters values depend on the resulting exponential distribution after ordering the clients model updates $\mathbb{X}_i^{f_{LA}}$. Before the definition of the linguistic quantifier of DDaBA, we first define the IOWA linguistic quantifier in Definition 3.2.

Definition 3.2 (Linguistic quantifier)**.** *It is a function $Q : [0,1] \to [0,1]$ verifying $Q(0) = 0$, $Q(1) = 1$ and $Q(x) \geq Q(y)$ for $x > y$. Equation 4 defines how the function Q computes the weighting values where w_i represents the weighting associated to the position i of a vector of dimension n, and Equation 5 defines the behaviour of the function Q.*

$$w_i^{(a,b)} = Q_{a,b}\left(\frac{i}{n}\right) - Q_{a,b}\left(\frac{i-1}{n}\right) \tag{4}$$

$$Q_{a,b}(x) = \begin{cases} 0 & 0 \leq x \leq a \\ \dfrac{x-a}{b-a} & a \leq x \leq b \\ 1 & b \leq x \leq 1 \end{cases} \tag{5}$$

where $a, b \in [0,1]$ satisfying $0 \leq a \leq b \leq 1$, and they set the intervals for calculating the contribution weight of each L_i. For the sake of clarification, those x values in the same interval will have the same weighting value.

$$Q_{a,b,c,y_b}(x) = \begin{cases} 0 & 0 \leq x \leq a \\ \dfrac{x-a}{b-a} \cdot y_b & a \leq x \leq b \\ \dfrac{x-b}{c-b} \cdot (1-y_b) + y_b & b \leq x \leq c \\ 1 & c \leq x \leq 1 \end{cases} \tag{6}$$

We redefine the function $Q_{a,b}$ for providing it a dynamic behaviour and a higher weighting of top clients, which depends on the random variable $\mathbb{X}_i^{f_{LA}}$. Accordingly, we propose Q_{a,b,c,y_b} that is defined in Equations 6, and incorporates two new parameters to the model (c and y_b), in addition to the two existing ones. The definition of each of the parameters is as follows:

1 Parameter a. This parameter represents the proportion of clients to which null weighing is assigned. Since we do not want to filter out those clients which stand out "at the top", i.e. those that obtain the best accuracy, we set the value to 0.

2 Parameter b. It sets the portion of clients we consider as top clients and we want to weight higher. The choice of this parameter is done dynamically, so that the top clients correspond to the first decile of the distribution of $\mathbb{X}_i^{f_{LA}}$. Formally, b is the portion of clients that verify

$$\mathbb{X}_i^{f_{LA}} \leq \frac{\ln(10/9)}{\lambda}, \tag{7}$$

where $\lambda = \frac{1}{\mu_{\mathbb{X}_i^{f_{LA}}}}$ and $\mu_{\mathbb{X}_i^{f_{LA}}}$ the mean of $\mathbb{X}_i^{f_{LA}}$.

3 The dynamic behavior of the parameter c. This parameter represents the portion of clients that we do not discard. For example, a value of $c = 0.8$ means that the 20% of the clients will be discarded. With the aim of dynamically adapt it in each aggregation, we identify the problem of filtering out adversarial clients as a problem of outlier detection in $\mathbb{X}_i^{f_{LA}}$. We thus employ the Tukey criteria [43, 44] for anomalies in exponential probability distribution functions and set $c = 1 - \hat{c}$ where $\hat{c}$ is the portion of clients that verify

$$\mathbb{X}_i^{f_{LA}} \geq Q_3 + 1.5IQR = \frac{\ln(4)}{\lambda} + 1.5\frac{\ln(3)}{\lambda}, \tag{8}$$

where $\lambda = \frac{1}{\mu_{\mathbb{X}_i^{f_{LA}}}}$ and $\mu_{\mathbb{X}_i^{f_{LA}}}$ the mean of $\mathbb{X}_i^{f_{LA}}$.

4 Parameter y_b. It provides the weighting of the top clients together with b. In particular, it represents the portion of the total weight assigned to these clients. In order to weight the top clients with double the importance of the rest of the clients participating in the aggregation, we set

$$y_b = \frac{2|Top|}{2|Top| - |Rest|}, \tag{9}$$

where $|Top| = b \times n$ and $|Rest| = (c-b) \times n$.

Analogously to Equation 4, we obtain the weighting of each client from the Q_{a,b,c,y_b} function according to Equation .

$$w_i^{(a,b,c,y_b)} = Q_{a,b,c,y_b}\left(\frac{i}{n}\right) - Q_{a,b,c,y_b}\left(\frac{i-1}{n}\right) \tag{10}$$

3.3. Defense based on the federated aggregation

Finally, using the equations defined above and the definitions of FL (Equation 1), we define DDaBA as a defense consisting of the following aggregation operator:

$$DDaBA(\{\hat{L}_1^t, \hat{L}_2^t, \ldots, \hat{L}_n^t\}, VD) = \sum_{i=1}^{n} w_i^{(a,b,c,y_b)} \hat{L}_i^t \tag{11}$$

where $w_i^{(a,b,c,y_b)}$ is defined in Equation 10 and $\hat{L}_i^t$ the local model update of the client i for $i \in \{1, \ldots, n\}$. Algorithm 1 depicts the DDaBA pseudo-code.

Algorithm 1 DDaBA

```
Input: local updates $\{\hat{L}_1^t, \hat{L}_2^t, \ldots, \hat{L}_n^t\}$ and $VD$
Initialize $G^t$
for $i = 0$ to $n$ do
    $f_{LA}(L_i) = \text{accuracy}(L_i, VD)$
end for
for $i = 0$ to $n$ do
    $\mathbb{X}_i^{f_{LA}} = \max_i\{f_{LA}(L_i)\} - f_{LA}(L_i)$
end for
$a = 0$
$b = |\mathbb{X}_i^{f_{LA}} \leq \frac{\ln(10/9)}{\lambda}|$
$c = |\mathbb{X}_i^{f_{LA}} \geq \frac{\ln(4)}{\lambda} + 1.5\frac{\ln(3)}{\lambda}|$
$y_b = \frac{2|b \times n|}{2|b \times n| - |(c-b) \times n|}$
for $i = 0$ to $n$ do
    $w_i = w_i^{(a,b,c,y_b)}$ according to Equation 10.
end for
$G_t = \sum_{i=0}^{n} w_i \hat{L}_i^t$
Return $G_t$
```

4. Experimental set-up

The evaluation of DDaBA is performed by means of the accuracy of the resulting FL model in three datasets arranged for FL, and we describe them in Section 4.1. Also, we deployed an image classification deep learning model in the FL setting. Since the main aim of this work is to propose a dynamic defense against byzantine attacks, we use an standard CNN-based image classification model composed of two CNN layers followed by its corresponding max-pooling layers, a dense layer and the output layer with a softmax activation function for the Fed-EMNIST and Fashion MNIST datasets and a pre-tained model based on EfficientNet [45] for the CIFAR-10 dataset. Finally, the federated aggregation operators used as baselines are introduced in Section 4.2 and the covered attacks in Section 4.3.

4.1. Evaluation datasets

Since the DDaBA needs a validation set for dynamically discarding adversarial clients, we create it from the test subsets of the three datasets, by assigning 20% of the sample in the test dataset to the validation set. The three datasets used in the evaluation are described as what follows:

1 The Fed-EMNIST (Federated Extended Modified NIST) dataset, which was presented in 2017 in [19] as an extension of the MNIST dataset [46]. The EMNIST Digits class contains a balanced subset of the digits dataset containing 28,000 samples of each digit. The dataset consists of 280,000 samples, which 240,000 are training samples and 40,000 test samples. We use its federated version by identifying each client with an original writer.

2 The Fashion MNIST [20] aims to be a more challenging replacement for the original MNSIT dataset. It contains a balanced subset of the 10 different classes containing 7,000 samples of each class. Hence, the dataset consists of 70,000 samples, which 60,000 are training samples and 10,000 test samples. We set the number of clients to 500.

3 The CIFAR-10 dataset is a labeled subset of the 80 million tiny images dataset [47]. It consists of 60000 32x32 color images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images, which correspond to 1000 images of each class. We set the number of clients to 100.

In summary, the datasets, after appropriate modifications to prepare the validation sets, follow the data distributions shown in Table 1.

Table 1: Size of the training, validation and test sets of Fed-EMNIST, Fashion MNIST and CIFAR-10 datasets.

	Training	**Validation**	**Test**
Fed-EMNIST	240,000	8,000	32,000
Fashion MNIST	60,000	2,000	8,000
CIFAR-10	60,000	2,000	8,000

With the aim of adapting both Fashion MNIST and CIFAR-10 datasets to a federated environment, the training data is distributed among the clients following a non-IID distribution. Accordingly, we randomly assign instances of a reduced number of labels to each client simulating a scenario in which each client contains partial information.

4.2. Baselines based on federated aggregation operators

We compare the DDaBA defense with the classical federated aggregation operator FedAvg [48] and the following state-of-the-art defenses against byzantine poisoning attacks:

- *Median* [49]. It is one of the byzantine-robust aggregation rules which is based on replacing the mean with the median in the aggregation method, which is more robust against extreme values.

- *Trimmed-mean* [50]. It represents another byzantine-robust aggregation rule. It relies on using a more robust version of the mean that consists in eliminating a fixed percentage (k) of extreme values both below and above the data distribution.

- *Krum and Multikrum* [40]. It sorts the clients according to the geometric distances of their model updates distributions and chooses the one closest to the majority as the aggregated model. Multikrum incorporates an d parameter, which specifies the number of clients to be aggregated (the first d after being sorted) resulting in the aggregated model.

- *Bulyan* [41]. It represents the state-of-the-art combining Krum and the thrimmed-mean. Hence, it sorts the clients according to their geometric distances and, according to an f parameter, filters out the $2f$ clients of the tails of the sorted distribution of clients and aggregates the rest of them.

The main weakness of Multikrum and Bulyan is that they strongly depend on a parameter given by the user. Both are optimal if the number of adversarial clients is known, which is usually not the case.

4.3. Byzantine Data and Model Poisoning Attacks

There are multitude of byzantine adversarial attacks both data and model poisoning. Due to the high number of clients participating in the aggregation and the low proportion of clients that will be adversarial in a reasonable configuration, poisoning attacks are very ineffective as their effect dissipates in the aggregation. For that reason, poisoning attacks are combined with model-replacement [17] techniques, which weight the contribution of adversarial clients in the aggregation according to a boosting parameter that is distributed among the adversarial clients.

The adversarial attacks covered in this work are the following:

- *Label-flipping attack* [21]. It is a data poisoning attack consisting of randomly flipping the labels of the adversarial attacks. This way, the adversarial clients learn incorrect information that send to the server.

- *Out-of-distribution attack* [22]. It is another data poisoning attack consisting of introducing into the adversarial clients' training dataset some samples out of the training distribution. In practice, the most frequent approaches are to introduce samples from another dataset with the same features (e.g. EMNIST and Fashion MNIST) or to introduce randomly generated samples. We adopt the second approach in the experimentation.

- *Random weights* [23]. It is a model poisoning attack based on randomly generate the model updates of each adversarial client.

We experiment with four different settings of adversarial clients for each of the previously described attacks:

- *1-out-of-30 attack scenario*. Consisting of 1 adversarial clients of a total of 30 clients participating in each aggregation, which represents $1/30$ of adversarial clients.

- *5-out-of-30 attack scenario*. Consisting of 5 adversarial clients of a total of 30 clients participating in each aggregation, which represents $1/6$ of adversarial clients.

- *10-out-of-50 attack scenario*. Consisting of 5 adversarial clients of a total of 50 clients participating in each aggregation, which represents $1/5$ of adversarial clients.

In each of the scenarios described, the boosting factor is divided by the number of adversarial clients in order to carry out the model-replacement.

4.4. Implementation details

We provide the code of DDaBA federated aggregtion operator[4] in order to ensure the reproducibility of the experiments. Due to the large number of existing FL frameworks [51] and with the aim of showing that DDaBA is independent of the framework, we have selected two of them:

- The Sherpa.ai FL [51] framework.
- The Flower [52] framework.

For each framework, we include Jupyter notebooks to visualise how the aggregation operator works and to facilitate its understanding.

5. Experimental results

We evaluate the performance of DDaBA as a defense to the byzantine attacks described in Section 2.2 in two ways: (1) In Section 5.1, we compare the behavior of DDaBA in terms of the performance of the resulting FL model with the baselines described in Section 4.2 and, (2) In Section 5.2 we analyze DDaBD in a scenario with a high number of adversarial clients, and we propose a modification of it for this particular scenario.

5.1. Analysis of the results

Tables 2, 3 and 4 show the results obtained in label-flipping, out-of-distribution and random weights attacks. Regarding the strength of the attacks, we find that all three are sufficiently powerful to pose a challenge to defenses. In fact, notice that the attack is slightly more effective when there are fewer adversarial clients since the boosting factor is divided among fewer

[4] `https://github.com/ari-dasci/S-DDaBA`

Table 2: Mean results for the label-flipping byzantine attack in terms of accuracy. We also show, in the first row, the expected accuracy with *FedAvg* but without any attack. The best result for each of the scenarios is highlighted in bold.

	Federated EMNIST			Fashion MNIST			CIFAR-10		
	1-out-of-30	**5-out-of-30**	**10-out-of-50**	**1-out-of-30**	**5-out-of-30**	**10-out-of-50**	**1-out-of-30**	**5-out-of-30**	**10-out-of-50**
No attack	0,9657	0,9657	0,9629	0,8719	0,8719	0,8697	0,8357	0,8357	0,8231
FedAvg	0,1591	0,4212	0,4007	0,1917	0,3665	0,4322	0,1184	0,1436	0,2448
Trim.-mean	0,9428	0,8739	0,8370	0,8672	0,8325	0,861	0,8239	0,7346	0,8220
Median	0,9313	0,9161	0,9097	0,8671	0,8473	0,8585	0,8287	0,8090	0,8289
Krum	0,8917	0,8706	0,8634	0,7264	0,7197	0,7473	0,7479	0,7610	0,7698
MultiKrum (5)	0,9132	0,9270	0,9189	0,8403	0,8433	0,8255	0,8164	0,8232	0,8114
MultiKrum (20)	0,9563	0,9571	0,9504	0,8727	0,8724	0,8680	0,8439	0,8479	0,8518
Bulyan (f=1)	0,9523	0,7813	0,5809	0,8689	0,7830	0,7875	0,8265	0,6595	0,6454
Bulyan (f=5)	0,9365	0,9421	0,9516	0,8617	0,8652	0,8726	0,8492	0,8451	0,8540
DDaBA	**0,9657**	**0,9663**	**0,9643**	**0,8817**	**0,8783**	**0,8807**	**0,8633**	**0,8503**	**0,8557**

Table 3: Mean results for the label-flipping byzantine attack in terms of accuracy. We also show, in the first row, the expected accuracy with *FedAvg* but without any attack. The best result for each of the scenarios is highlighted in bold.

	Federated EMNIST			Fashion MNIST			CIFAR-10		
	1-out-of-30	**5-out-of-30**	**10-out-of-50**	**1-out-of-30**	**5-out-of-30**	**10-out-of-50**	**1-out-of-30**	**5-out-of-30**	**10-out-of-50**
No attack	0,9657	0,9657	0,9629	0,8719	0,8719	0,8697	0,8357	0,8357	0,8231
FedAvg	0,4093	0,4404	0,4350	0,2041	0,3667	0,4657	0,1468	0,1922	0,3419
Trim.-mean	0,9456	0,8602	0,8531	0,8652	0,8348	0,8310	0,8202	0,7441	0,7400
Median	0,9345	0,9200	0,9144	0,8662	0,8465	0,8454	0,8223	0,8019	0,8073
Krum	0,8693	0,8668	0,8621	0,7361	0,7062	0,7281	0,7202	0,7310	0,7408
MultiKrum (5)	0,9169	0,9330	0,9198	0,8493	0,8430	0,8345	0,8305	0,8191	0,8023
MultiKrum (20)	0,9545	0,9544	0,9506	0,8747	0,8719	0,8733	0,8607	0,8519	0,8521
Bulyan (f=1)	0,9507	0,7872	0,5812	0,8704	0,7601	0,6930	0,8319	0,6862	0,5551
Bulyan (f=5)	0,9353	0,9383	0,9502	0,8713	0,8654	0,8757	0,8440	0,8498	0,8481
DDaBA	**0,9652**	**0,9620**	**0,9654**	**0,8761**	**0,8841**	**0,8783**	**0,8626**	**0,8599**	**0,8632**

Table 4: Mean results for the label-flipping byzantine attack in terms of accuracy. We also show, in the first row, the expected accuracy with *FedAvg* but without any attack. The best result for each of the scenarios is highlighted in bold.

	Federated EMNIST			Fashion MNIST			CIFAR-10		
	1-out-of-30	**5-out-of-30**	**10-out-of-50**	**1-out-of-30**	**5-out-of-30**	**10-out-of-50**	**1-out-of-30**	**5-out-of-30**	**10-out-of-50**
No attack	0,9657	0,9657	0,9629	0,8719	0,8719	0,8697	0,8357	0,8357	0,8231
FedAvg	0,0997	0,0994	0,1001	0,1006	0,1016	0,0997	0,0998	0,0994	0,1005
Trim.-mean	0,9537	0,1039	0,0990	0,8751	0,1004	0,0999	0,8608	0,0992	0,0998
Median	0,9367	0,9354	0,9342	0,8654	0,8618	0,8554	0,8499	0,8664	0,8646
Krum	0,8314	0,8652	0,8541	0,7156	0,7459	0,7342	0,7184	0,7164	0,7994
MultiKrum (5)	0,9325	0,9228	0,9191	0,8348	0,8343	0,8278	0,8164	0,8115	0,8167
MultiKrum (20)	0,9565	0,9577	0,9510	0,8764	0,8751	0,8676	0,8488	0,8488	0,8531
Bulyan (f=1)	0,9598	0,0997	0,0998	0,0990	0,1001	0,0990	0,8529	0,0996	0,0993
Bulyan (f=5)	0,9379	0,9377	0,9514	0,8746	0,8690	0,8746	0,8502	0,8411	0,8519
DDaBA	**0,9653**	**0,9645**	**0,9622**	**0,8801**	**0,8778**	**0,8777**	**0,8656**	**0,8624**	**0,8626**

clients. The out-of-distribution attack is slightly less damaging while the random weights attack achieves the lowest performance without defense, ranking as the most challenging. The results obtained both in the different types of attacks and in the considered datasets confirm common conclusions, so we discuss all the results as a whole.

When evaluating the performance of the baselines we hereby confirm that MultiKrum and Bulyan do indeed represent the state of the art. However, they are highly dependent of the d and f parameters since they set the number of clients to keep or discard, respectively, in the aggregation. For example, in the 10-out-of-50 scenario and Bulyan with $f = 1$ we verify this weakness, since only $2f = 2$ clients would be discarded from the aggregation, which is not enough to defend the model in the presence of 10 adversarial clients. A possible solution would be to set this value always to high, but this is also a limitation because in the case of having fewer adversarial clients than $2f$ the quality of the model decreases (e.g., 1-out-of-30 using Bulyan with $f = 5$). Finally, MultiKrum and Bulyan promise optimal performance in the case of knowing the number of adversarial clients, which is not the case. This enhances the need for a defense that dynamically estimates how many clients to filter in the aggregation.

In contrast, the outperformance of DDaBA is confirmed in all the attack settings considered enhancing its success regardless of the type of attack and the proportion of adversarial clients. Moreover, DDaBA achieves better results than the no attack situation in the vast majority of the scenarios. This is because the dynamic filtering of clients not only discards those that are adversarial but also those that perform too poorly to contribute to improving the global learning model.

5.2. Extreme attack scenarios - A static version of DDaBA

It has been proven that discarding clients based on whether or not they are outliers in a distribution formed from performance on a common validation set overcomes the defenses of the state of the art. However, this approach based on data distributions has a weakness stemmed from the fact that the distribution we use to search outliers is configured with the same data that we subsequently evaluate. Therefore, with a very high presence of adversarial clients, the resulting distribution will be highly skewed by this data, resulting in no outlier. Although we recognize this weakness, we point out that it is not a major one, since it is highly unlikely for the percentage of adversarial clients in an FL scenario to be so high as to cause the defense to fail.

To overcome this weakness, we propose a static version of DDaBA called Static Defense Against Byzantine Attacks (SDaBA), which incorporates the only difference that the proportion of clients to be discarded from the aggregation is computed using a fixed parameter α. In particular, instead of eliminating those clients that represent outliers in the distribution $\mathbb{X}_i^{f_{LA}}$, we eliminate those clients whose distance to the best accuracy is greater than α times the maximum of the distances. In other words, using $\mathbb{X}_i^{f_{LA}}$, we set $c = 1 - \beta$ where β is the portion of clients verifying that

$$\mathbb{X}_i^{f_{LA}} \geq \alpha \mathbb{X}_n^{f_{LA}} \quad \forall i \in \{1, \ldots, n\} \tag{12}$$

in Equations 6 and 10. Analogously, we set $b = 0.2$ in order to consider as top clients the top 20% clients.

With the aim of evaluating SDaBA we set $\alpha = 1/4$ and the 10-out-of-30 attack scenario consisting of 10 adversarial clients of a total of 30 clients participating in each aggregation, which represents 1/3 of adversarial clients, which is an unusual high proportion of them. Table 5 shows the results of DDaBA and SDaBA in comparison with the baselines in this extreme attack scenario in Federated EMNIST.

Table 5: Mean results for the extreme scenario (10-out-of-30) in Federated EMNIST in terms of accuracy. We also show, in the first row, the expected accuracy with *FedAvg* but without any attack. The best result for each of the scenarios is highlighted in bold.

	Label-flipping	**Out-of-dist.**	**Random weights**
No attack	0,9657	0,9657	0,9657
FedAvg	0,3561	0,4394	0,0994
Trimmed-mean	0,6256	0,5778	0,1002
Median	0,8595	0,8347	0,9355
Krum	0,8801	0,8678	0,8633
MultiKrum (5)	0,9336	0,9366	0,9349
MultiKrum (20)	0,9623	0,9617	0,8595
MultiKrum (25)	0,9623	0,9617	0,8595
Bulyan (f=1)	0,4755	0,5005	0,1000
Bulyan (f=5)	0,9485	0,9475	0,9455
DDaBA	0,4235	0,4819	0,0997
SDaBA (1/4)	**0,9654**	**0,9653**	**0,9629**

The results show how this extreme scenario highly affects to DDaBA, but also Bulyan (f=1). With respect to the baselines, in this case it is MultiKrum with $d = 20$ that achieves the best results by setting the d parameter to its optimal value. Finally, we highlight the appropriate performance of SDaBA, outperforming the rest of the defenses and solving the problem of extreme scenarios.

6. Conclusion and future work

We addressed the problem of defending against byzantine attacks in FL, which is a real challenge since the existing defenses are not enough. Using the exponential distribution resulting of the differences between the best model and the rest of them in terms of accuracy over a central validation set, we consider that those clients that represent outliers in that distribution are likely to be adversarial ones. Hence, we propose DDaBA, a defense against byzantine attacks which dynamically filters out the adversarial and poor clients.

We evaluated the DDaBA in three different byzantine attacks, in three datasets and using three different settings. In addition, we proposed a static version of the defense approach in order to use it in scenarios with an extremely high proportion of adversarial clients. Both the experiments corroborate the following conclusions:

- DDaBA is a highly effective defense against byzantine attacks in real attack scenarios.
- It properly filters out adversarial and poor clients improving the performance of the global model in scenarios with adversarial clients, even outperforming the performance in the original task.
- The static version SDaBA is an effective solution for extreme attack scenarios.

To conclude, we have proven that DDaBA is a high quality defense against byzantine attacks, and it can act as a proper federated aggregation operator, since it defends the global model against the effect of the attacks while improving the learning of the global model.

Appendix A. Ordered weighted model averaging

Group decision making is the AI task focused on finding out a consensus decision from a set of experts by summing up their individual evaluations. Yager proposed in [42] the Ordered Weighted Averaging (OWA) operators with the aim of modelling the fuzzy opinion majority [53] in group decision making. Yager and Filev generalised the OWA operator definition in [18], where they defined the OWA operator with an order-induced vector for ordering the argument variable. They called this generalisation of OWA operators with a specific semantic in the aggregation process as Induced Ordered Weighted Averaging (IOWA). The OWA and IOWA operators are weighted aggregation functions that are mathematically defined as what follows:

Definition Appendix A.1 (OWA Operator [42])**.** *An OWA operator of dimension n is a function $\Phi : \mathbb{R}^n \to \mathbb{R}$ that has an associated set of weights or weighting vector $W = (w_1, \dots, w_n)$ so that $w_i \in [0,1]$ and $\sum_{i=1}^n w_i = 1$, and it is defined to aggregate a list of real values $\{c_1, \dots, c_n\}$ according to the Equation A.1:*

$$\Phi(c_1, \dots, c_n) = \sum_{i=1}^{n} w_i c_{\sigma(i)} \tag{A.1}$$

being $\sigma : \{1, \dots, n\} \to \{1, \dots, n\}$ a permutation function such that $c_{\sigma(i)} \geq c_{\sigma(i+1)}$, $\forall i = \{1, \dots, n-1\}$.

Definition Appendix A.2 (IOWA Operator [18])**.** *An IOWA operator of dimension n is a mapping $\Psi : (\mathbb{R} \times \mathbb{R})^n \to \mathbb{R}$ which has an associated set of weights $W = (w_1, \dots, w_n)$ so that $w_i \in [0,1]$ and $\sum_{i=1}^n w_i = 1$, and it is defined to aggregate the second arguments of a 2-tuple list $\{\langle u_1, c_1\rangle, \dots, \langle u_n, c_n\rangle\}$ according to the following expression:*

$$\Psi(\langle u_1, c_1\rangle, \dots, \langle u_n, c_n\rangle) = \sum_{i=1}^{n} w_i c_{\sigma(i)} \tag{A.2}$$

being $\sigma : \{1, \dots, n\} \to \{1, \dots, n\}$ a permutation function such that $u_{\sigma(i)} \geq u_{\sigma(i+1)}$, $\forall i = \{1, \dots, n-1\}$. The vector of values $U = (u_1, \dots, u_n)$ is called the order-inducing vector and $(c_1, \dots, c_n)$ the values of the argument variable.

The OWA and IOWA operators are functions for weighting the contribution of experts for the global decision in the case of group decision making, and the contribution of a set of clients in an aggregation process in a general scenario. However, they need an additional function to calculate the values of the parameters, which in the context of group decision making means the grade of membership to a fuzzy concept. The weight value calculation function is known as linguistic quantifier [54], which is defined as a function $Q : [0,1] \to [0,1]$ such as $Q(0) = 0$, $Q(1) = 1$ and $Q(x) \geq Q(y)$ for $x > y$. Equation A.3 defines how the function Q computes the weight values and Equation A.4 defines the behaviour of the function Q.

$$w_i^{(a,b)} = Q_{a,b}\left(\frac{i}{n}\right) - Q_{a,b}\left(\frac{i-1}{n}\right) \tag{A.3}$$

$$Q_{a,b}(x) = \begin{cases} 0 & 0 \leq x \leq a \\ \dfrac{x-a}{b-a} & a \leq x \leq b \\ 1 & b \leq x \leq 1 \end{cases} \tag{A.4}$$

where $a, b \in [0,1]$ satisfying $0 \leq a \leq b \leq 1$.

The function Q in Equation A.4 can be redefined in order to model different linguistic quantifiers. Since the definition of the notion quantifier guided aggregation [42, 54], other definitions of the function Q has been proposed to model different linguistic quantifiers like "most" or "at least" [53].

Acknowledgements

This research work is supported by the R&D&I grants PID-2020-119478GB-I00, PID2020-116118GA-I00 and EQC2018-005084-P funded by MCIN/AEI/10.13039/501100011033 and by "ERDF A way of making Europe". Nuria Rodríguez Barroso were supported by the grant FPU18/04475 funded by MCIN-/AEI/10.13039/501100011033 and by "ERDF A way of making Europe". Eugenio Martínez Cámara were supported by the grant IJC2018-036092-I funded by MCIN/AEI/10.13039/5011-00011033.

References

[1] J. Dean, G. Corrado, R. Monga, C. Kai, M. Devin, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, Q. Le, A. Ng, Large scale distributed deep networks, Advances in Neural Information Processing Systems 25 (2012) 1223–1231.

[2] C. Ma, J. Konečný, M. Jaggi, V. Smith, M. Jordan, P. Richtárik, M. Takáč, Distributed optimization with arbitrary local solvers, Optimization Methods and Software 32 (2017) 813–848.

[3] P. Kairouz, H. B. McMahan, Advances and open problems in federated learning, Foundations and Trends® in Machine Learning 14 (2021).

[4] M. Chen, B. Mao, T. Ma, Fedsa: A staleness-aware asynchronous federated learning algorithm with non-iid data, Future Generation Computer Systems 120 (2021) 1–12.

[5] V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, G. Srivastava, A survey on security and privacy of federated learning, Future Generation Computer Systems 115 (2021) 619–640.

[6] X. Zhang, M. Hu, J. Xia, T. Wei, M. Chen, S. Hu, Efficient federated learning for cloud-based AIoT applications, in: IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2020.

[7] N. Dalvi, P. Domingos, Mausam, S. Sanghai, D. Verma, Adversarial classification, Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2004) 99–108.

[8] L. Huang, A. D. Joseph, B. Nelson, B. I. Rubinstein, J. D. Tygar, Adversarial machine learning, Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence (2011) 43–58.

[9] B. Biggio, I. Corona, B. Nelson, B. I. P. Rubinstein, D. Maiorca, G. Fumera, G. Giacinto, F. Roli, Security Evaluation of Support Vector Machines in Adversarial Environments, 2014, pp. 105–153.

[10] T. Gu, K. Liu, B. Dolan-Gavitt, S. Garg, Badnets: Evaluating backdooring attacks on deep neural networks, IEEE Access 7 (2019) 47230–47244.

[11] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nita-Rotaru, B. Li, Manipulating machine learning: Poisoning attacks and countermeasures for regression learning, IEEE Symposium on Security and Privacy (SP) (2018) 19–35.

[12] A. N. Bhagoji, S. Chakraborty, P. Mittal, S. Calo, Model poisoning attacks in federated learning, in: In Workshop on Security in Machine Learning (SecML), collocated with the 32nd Conference on Neural Information Processing Systems (NeurIPS'18), 2018.

[13] B. McMahan, E. Moore, D. Ramage, S. Hampson, B. A. y Arcas, Communication-Efficient Learning of Deep Networks from Decentralized Data 54 (2017) 1273–1282.

[14] J. Konečný, H. McMahan, D. Ramage, P. Richtárik, Federated optimization: Distributed machine learning for on-device intelligence, CoRR abs/1511.03575 (2016).

[15] Y. Wang, CO-OP: Cooperative Machine Learning from Mobile Devices, Master's thesis, University of Alberta, 2017.

[16] C. Fung, C. J. M. Yoon, I. Beschastnikh, Mitigating sybils in federated learning poisoning, CoRR abs/1808.04866 (2018).

[17] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, V. Shmatikov, How to backdoor federated learning, Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics 108 (2020) 2938–2948.

[18] R. Yager, D. Filev, Induced ordered weighted averaging operators, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 29 (1999) 141–150.

[19] G. Cohen, S. Afshar, J. Tapson, A. van Schaik, Emnist: Extending mnist to handwritten letters, International Joint Conference on Neural Networks (IJCNN) (2017) 2921–2926.

[20] H. Xiao, K. Rasul, R. Vollgraf, Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, CoRR abs/1708.07747 (2017).

[21] V. Tolpegin, S. Truex, M. E. Gursoy, L. Liu, Data poisoning attacks against federated learning systems, CoRR abs/2007.08432 (2020).

[22] S. Fort, J. Ren, B. Lakshminarayanan, Exploring the limits of out-of-distribution detection, CoRR abs/2106.03004 (2021).

[23] J. Lin, M. Du, J. Liu, Free-riders in federated learning: Attacks and defenses, CoRR abs/1911.12560 (2019).

[24] J. Konečný, H. McMahan, F. X. Yu, P. Richtárik, A. Suresh, D. Bacon, Federated learning: Strategies for improving communication efficiency, CoRR abs/1610.05492 (2016).

[25] P. Laskov, R. Lippmann, Machine learning in adversarial environments, Machine Learning 81 (2010) 115–119.

[26] N. Dalvi, P. Domingos, Mausam, S. Sanghai, D. Verma, Adversarial classification, Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2004) 99–108.

[27] B. Nelson, M. Barreno, F. Jack Chi, A. D. Joseph, B. I. P. Rubinstein, U. Saini, C. Sutton, J. D. Tygar, K. Xia, Misleading Learners: Co-opting Your Spam Filter, 2009, pp. 17–51.

[28] C. Croux, P. Filzmoser, M. Oliveira, Algorithms for projection–pursuit robust principal component analysis, Chemometrics and Intelligent Laboratory Systems 87 (2007) 218–225.

[29] L. Lyu, H. Yu, X. Ma, L. Sun, J. Zhao, Q. Yang, P. S. Yu, Privacy and robustness in federated learning: Attacks and defenses, CoRR abs/2012.06337 (2020).

[30] M. Nasr, R. Shokri, A. Houmansadr, Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning, IEEE Symposium on Security and Privacy (SP) (2019) 739–753.

[31] D. Cao, S. Chang, Z. Lin, G. Liu, D. Sun, Understanding distributed poisoning attack in federated learning, IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS) (2019) 233–239.

[32] C. Dwork, A. Roth, The algorithmic foundations of differential privacy, Foundations and Trends® in Theoretical Computer Science 9 (2014) 211–407.

[33] X. Zhou, M. Xu, Y. Wu, N. Zheng, Deep model poisoning attack on federated learning, Future Internet 13 (2021).

[34] G. Sun, Y. Cong, J. Dong, Q. Wang, J. Liu, Data poisoning attacks on federated machine learning, CoRR abs/2004.10020 (2020).

[35] L. Lamport, R. Shostak, M. Pease, The Byzantine Generals Problem, Association for Computing Machinery, 2019, p. 203–226.

[36] J. Steinhardt, P. W. Koh, P. Liang, Certified defenses for data poisoning attacks, Proceedings of the 31st International Conference on Neural Information Processing Systems (2017) 3520–3532.

[37] M. Shayan, C. Fung, C. J. M. Yoon, I. Beschastnikh, Biscotti: A ledger for private and secure peer-to-peer machine learning, CoRR abs/1811.09904 (2018).

[38] S. Shen, S. Tople, P. Saxena, Auror: defending against poisoning attacks in collaborative deep learning systems, Proceedings of the 32nd Annual Conference on Computer Security Applications (2016) 508–519.

[39] D. Yin, Y. Chen, R. Kannan, P. Bartlett, Byzantine-robust distributed learning: Towards optimal statistical rates, Proceedings of the 35th International Conference on Machine Learning 80 (2018) 5650–5659.

[40] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, J. Stainer, Machine learning with adversaries: Byzantine tolerant gradient descent, Advances in Neural Information Processing Systems 30 (2017) 119–129.

[41] E. M. El Mhamdi, R. Guerraoui, S. Rouault, The hidden vulnerability of distributed learning in Byzantium, Proceedings of the 35th International Conference on Machine Learning 80 (2018) 3521–3530.

[42] R. Yager, On ordered weighted averaging aggregation operators in multicriteria decisionmaking, IEEE Transactions on Systems, Man, and Cybernetics 18 (1988) 183–190.

[43] J. W. Tukey, Exploratory Data Analysis, Addison-Wesley, 1977.

[44] K. Wada, Outliers in official statistics, Japanese Journal of Statistics and Data Science 3 (2020).

[45] M. Tan, Q. Le, EfficientNet: Rethinking model scaling for convolutional neural networks, in: Proceedings of the 36th International journal on Machine Learning, volume 97, 2019, pp. 6105–6114.

[46] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (1998) 2278–2324.

[47] A. Torralba, R. Fergus, W. T. Freeman, 80 million tiny images: A large data set for nonparametric object and scene recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (2008) 1958–1970.

[48] B. McMahan, E. Moore, D. Ramage, S. Hampson, B. A. y Arcas, Communication-Efficient Learning of Deep Networks from Decentralized Data, volume 54, 2017, pp. 1273–1282.

[49] Y. Chen, L. Su, J. Xu, Distributed statistical machine learning in adversarial settings: Byzantine gradient descent, volume 1, 2017.

[50] D. Yin, Y. Chen, K. Ramchandran, P. L. Bartlett, Byzantine-robust distributed learning: Towards optimal statistical rates, CoRR abs/1803.01498 (2018).

[51] N. Rodríguez-Barroso, G. Stipcich, D. Jiménez-López, J. A. Ruiz-Millán, E. Martínez-Cámara, G. González-Seco, M. V. Luzón, M. Ángel Veganzones, F. Herrera, Federated learning and differential privacy: Software tools analysis, the Sherpa.ai FL framework and methodological guidelines for preserving data privacy, Information Fusion 64 (2020) 270 – 292.

[52] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, T. Parcollet, N. D. Lane, Flower: A friendly federated learning research framework, arXiv preprint arXiv:2007.14390 (2020).

[53] G. Pasi, R. Yager, Modeling the concept of majority opinion in group decision making, Information Science 176 (2006) 390–414.

[54] R. Yager, Quantifier guided aggregation using owa operators, International Journal of Intelligent Systems 11 (1996) 49–73.