Efficient Federated Learning Under Non-IID Conditions With Attackers

Huan Zou
BUPT
Beijing, China
zouhuan@bupt.edu.cn

Yuchao Zhang*
BUPT
Beijing, China
yczhang@bupt.edu.cn

Xirong Que
BUPT
Beijing, China
rongqx@bupt.edu.cn

Yilei Liang
University of Cambridge
London, UK
yl841@cst.cam.ac.uk

Jon Crowcroft
University of Cambridge
London, UK
Jon.Crowcroft@cl.cam.ac.uk

ABSTRACT

Federated learning (FL) has recently attracted much attention due to its advantages for data privacy. But every coin has two sides: protecting users' data (not requiring users to send their data) also makes FL more vulnerable to some types of attacks, such as targeted attacks and untargeted attacks. Many robust FL algorithms have therefore been proposed, in order to ensure training accuracy under such attacks. Some of the existing solutions assume that data conforms to the independent and identically distribution (i.i.d), so as to simplify the problem. But, limiting the data distribution to i.i.d hinders the practical application of FL, and FL under non-i.i.d conditions is more general. However, designing efficient robust algorithm for FL under non-i.i.d faces two additional challenges: identifying malicious clients and guaranteeing model accuracy. To tackle these challenges, we propose a new FL workflow named *Cominer* which consists of a Label Cluster process and a Vertical Comparison (VC) process. LC solves the problem of declining accuracy by supporting non-iid data diversity by classifying all clients into multiple clusters, then VC identifies and eliminates malicious clients within each cluster. We verify the improvement in accuracy achieved by *Cominer* in a series of experiments, and show that under Non-IID conditions, *Cominer* not only improves the accuracy of the federated model over previous algorithms by up to 24.85%, but also enjoys high resilience to different kinds of attacks while maintaining accuracy over 80%.

CCS CONCEPTS

• **Security and privacy** → Distributed systems security.

KEYWORDS

federated learning, robustness, Non-i.i.d

ACM Reference Format:
Huan Zou, Yuchao Zhang, Xirong Que, Yilei Liang, and Jon Crowcroft. 2022. Efficient Federated Learning Under Non-IID Conditions With Attackers. In *1st ACM Workshop on Data Privacy and Federated Learning Technologies for Mobile Edge Network (FedEdge '22), October 17, 2022, Sydney, NSW, Australia.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3556557.3557951

1 INTRODUCTION

With increasing attention being paid to data privacy, data owners are becoming less willing to share their data, which greatly limits the applications of traditional decentralised machine learning. Federated Learning (FL) [16] is a new distributed machine learning framework proposed by Google, which allows multiple clients to jointly train a federated model without uploading their local data to a central server. The protection of privacy makes FL receive widespread attention from industry and academia.

While FL can avoid direct leakage of data privacy, it faces serious potential attacks due to the weak control over the clients. For example, the well-known poison attack which includes targeted attack [1, 20, 22, 24] and untargeted attack [3][7][8], aims to destroy the federated models by modifying the training data or updating the wrong gradients. The widely-used aggregation method, FedAvg [16], was proved

*Corresponding authors

The work was supported in part by the National Natural Science Foundation of China(NSFC) under Grant 62172054, the Key Project of Beijing Natural Science Foundation under M21030, the NSFC under Grant 62072047, and the National Key R&D Program of China under Grant 2019YFB1802603.

FedEdge '22, October 17, 2022, Sydney, NSW, Australia

ACM ISBN 978-1-4503-9521-2/22/10…$15.00
https://doi.org/10.1145/3556557.3557951

to be susceptible to attacks [12]. Consequently, robustness aggregation [3, 5, 7, 26] are proposed to defence attacks.

While most of these robust algorithms require the data to be independent and identically distributed (i.i.d), which simplifies the identification of attacks. However, the i.i.d assumption is not easy to be satisfied: data between clients are often non-independent and identically distribution (non-i.i.d). This brings two challenges to FL:

Non-i.i.d increases the difficulty of identifying attackers. Under i.i.d conditions, the gradients uploaded by normal clients have the consistent convergence direction, while that of malicious gradients are inconsistent [27]. Therefore, the server can easily identify those poisoned gradients. However, in the case of non-i.i.d, the normal gradients are inconsistent too, which increases the difficulty to find abnormal gradients.

Non-i.i.d decreases the accuracy of federated models. First, clients will generate inconsistent gradients according to their local training data. Such inconsistent will compromise the performance of the federated model even though there is no attack. Second, to prevent attacks, robust algorithms[3, 5, 7, 26] are introduced into FL, but excessively pursue the elimination of attacks may exclude lots of normal clients. When the data is non-i.i.d, the local data of each client may be unique, which makes each client irreplaceable in FL. Arbitrary exclusion of clients will seriously affect the accuracy, efficiency, availability, and fairness [14] of the federated model.

To tackle these challenges, we propose a new FL workflow named Cominer. First, to solve the low accuracy problem caused by non-i.i.d, we propose Label Cluster (LC), a clustering scheme. LC can divide clients with similar data labels into the same cluster to save more normal clients when robust algorithms are used in FL. Second, to achieve robustness, we proposed Vertical Comparison (VC) to exclude clients with abnormal vertical distance, where the vertical distance is the gradients' distance between adjacent rounds of each client. The combination of LC and VC can ensure the robustness and effectiveness of FL to the greatest extent. We name the whole FL framework Cominer, COMpare IN clustER, which works like a miner to detect attackers.

2 RELATED WORK

Federated learning has attracted much attention in recent years. This section gives an overview of prior work in robustness and optimization in federated learning.

2.1 Robustness in federated learning

Robustness research aims to mitigate poison attacks[2][14]. Such as Krum [3] and Bulyan [7] are two classical algorithms that compare the similarity between the gradients of clients and then keep the most similar gradients to update the federated model. However, directly excluding the majority of participating clients is an effective but arbitrary protection method, so Geometric Median [5] and Trimmed Mean [26] respectively choose to compute the geometric median of all gradients and the mean of the gradients after excluding outliers in each dimension of the gradient as the aggregation result. Besides, BREA [19] alleviates this problem by a secure aggregation function. Fed-Influence [25] eliminates malicious clients by evaluating the impact of each gradient on the accuracy of the global model. However, all of these approaches are based on the assumption of the i.i.d.

The data heterogeneous setting of clients makes it difficult to resist attacks by comparing the similarity of gradients. Therefore, RSA [12] was proposed to eliminate the influence of malicious gradients by adding a regularization term to the loss function to avoid direct comparison between gradients. FoolsGold [9] eliminates byzantine clients by making use of the feature that gradients provided by the byzantine clients are more similar than that of the normal clients. Li et al. [13] are the first one that employs spectral anomaly detection against malicious clients in the heterogeneous FL setting. Scattler et al. [18] prove that CFL [17], which is based on a clustering algorithm, can exclude malicious clients. While these strategies can improve robustness in some non-i.i.d cases, they may not be effective in some conditions like each client only have one class of data.

2.2 FL Optimization.

Heterogeneity of data would possibly affect federated model accuracy even with no attacks [15]. This concern leads to research on FL optimization. Oort [10] selects high statistical utility and system efficiency clients to participate in federated training. While a single federated model may not perform well on all clients, some researchers propose to train multiple federated models to solve the problems caused by non-i.i.d [21]. CFL [17] trains two models in parallel and Briggs et al. [4] introduce a hierarchical clustering step to separate clusters of clients. Li et al. [14] show that the constraints of fairness and robustness can directly compete when training a single global model and propose Ditto, a multi-task learning framework, using personalized federated learning to train a separate model for each client. However, these algorithms ignore the existence of attackers.

Our work strives to achieve both efficient and robust FL.

3 COMINER FRAMEWORK

Federated learning is vulnerable to attacks. We believe that in order to achieve the purpose of the attack, the gradients uploaded by attacker in adjacent rounds may be more regular than that of normal clients. At the same time, normal clients should be saved as much as possible to maintain the data diversity and improve the federated model's accuracy. The

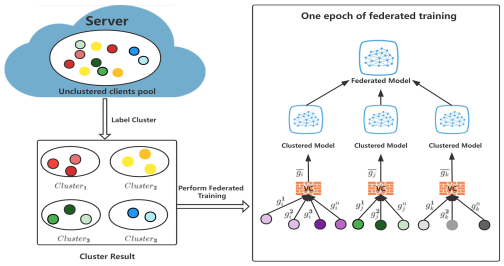

Figure 1: The framework of Cominer.

Algorithm 1: Clustering Process

Input: V: label vectors, θ: threshold, $\overline{P}$: unclustered clients pool, $client_t$
Output: b: one cluster

for v_i *in* V **do**
  $s_{i,n} = \min(\frac{v_n \cdot v_i}{|v_n|}, \frac{v_n \cdot v_i}{|v_i|})$, $s = s \cup s_{i,n}$;
while $\max(s) > \theta$ **do**
  s = {};
  select the $client_j$ with the largest $s_{j,n}$;
  $b = b \cup client_j, \overline{P} = \overline{P} - client_j, V = V - v_j$;
  for v_i *in* V **do**
    $s_{i,j} = \min(\frac{v_j \cdot v_i}{|v_j|}, \frac{v_j \cdot v_i}{|v_i|})$, $s = s \cup s_{i,j}$;
return b;

federated system optimization workflow Cominer consists of two parts which correspond to the above two conclusions.

LC divides all clients into different clusters according to their data label similarity. Then, Vertical Comparison (VC) acts like a firewall to identify possible malicious clients among the clients belonging to the same cluster (the cluster classified by LC). In each round of federated training, local model, clustered model and federated model appear sequentially. The local model is the foundation, and the clustered model connects the local model and the federated model. Fig.1 shows the overall workflow of Cominer.

3.1 Label Cluster

3.1.1 Design principle of LC. First, the heterogeneity causes a decrease in the model's accuracy [27]. After implementing LC, clients in the same cluster have similar label distributions. At this point, when a robust algorithm is used as the aggregation method in each cluster, even if most clients in the cluster are excluded, the remaining clients can well represent the data distribution of the clients that were incorrectly excluded. Moreover, different cluster represents different data distributions, retaining as many clients as possible from different clusters enables the federated model to obtain more valuable gradients in each round of federated training.

Table 1: Example of client's data label vector

	l_0	l_1	l_2	l_3	l_4	l_5	l_6	l_7	l_8	l_9
$client_0$	0	1	0	1	1	0	0	1	0	1

3.1.2 How to calculate the similarity. For the convenience of subsequent explanations, we use the MNIST [11] dataset as an example. Table 1 shows the label vectors of a client, where l_i represents the label, and i corresponds to the 10 kinds of handwritten digits in the MNIST dataset. The value of l_i represents whether the client owns this type of data. We define $S(v_i, v_j)$ to calculate the similarity $s_{i,j} (0 \leq s_{i,j} \leq 1)$ between v_i and v_j.

$$S(v_i, v_j) = \min(\frac{v_i \cdot v_j}{|v_i|}, \frac{v_i \cdot v_j}{|v_j|}) \tag{1}$$

θ is set to the similarity threshold, for $client_i$ and $client_j$ within the same cluster, $s_{i,j}$ should be greater than θ.

3.1.3 The workflow of LC. The central server maintains an unclustered clients pool P and collects all clients' label vector to V. When starting a round of clustering, the server will randomly select $client_n$ from P, and then finds other clients with similar label vector to $client_n$.

Algorithm 1 shows a round of LC process. First, the server will copy the latest P to $\overline{P}$ and use Eq. (1) to calculate the label similarity between v_n and other clients and then remove clients whose label similarity to $client_n$ is less than θ from $\overline{P}$. However, it can only be guaranteed that the similarity between the client in $\overline{P}$ and $client_n$ meets the threshold limit, but the similarities between the clients left in $\overline{P}$ is no guarantee. Therefore, the server takes out $client_j$ which is the most similar to $client_n$ from $\overline{P}$ and add $client_j$ to b, b is used to save the clients that belong to the same cluster as $client_n$. Then repeat the clustering process with v_j replacing v_n until no clients remain in $\overline{P}$ and b is the cluster result. Finally, delete all clients in b from P and start the next round of clustering. Label Cluster ends when there are no clients left in P.

3.2 Vertical Comparison

The non-i.i.d of data leads to the gradients uploaded by each client converge to multiple directions and makes robust algorithms challenging to defend against attackers.

Therefore, we design Vertical Comparison and try to use the characteristics of the gradients in adjacent epochs to identify and exclude attackers among the clusters obtained

Algorithm 2: Vertical Comparison

Input: $cluster_k$: the cluster to be checked, $\tilde{m}$: multiple, G: gradients of all nodes.

d_t = **for** g_t^i, g_t^{i-1} *in* G **do**
  $d_t = s_{t,t-1}^i(g_t^i, g_{t-1}^i) \cup d_t$
for d_t^i *in* d_t **do**
  c = 0;
  for d_t^j *in* d_t **do**
    if $i \neq j$ *and* $d_t^i > d_t^j \times \tilde{m}$ **then**
      c = c + 1;
  if $c > len(cluster_k) / 2$ **then**
    $cluster_k = cluster_k - client_i$;

by LC. We calculate the Euclidean distance:

$$s_{t,t-1}^i = \sqrt{\sum_{j=1}^{n}(g_t^{ij} - g_{t-1}^{ij})^2} \tag{2}$$

between adjacent gradients to represent the vertical distance, where j is the j_{th} dimension of $client_i$'s gradient. Algorithm 2 shows how VC works within a cluster.

Attackers may appear in any round of FL, but VC needs at least two epochs to compute the vertical distance. Therefore, we perform 3 rounds of cheat-training at the beginning of FL. During cheat-training the gradients g_i uploaded by all clients will be saved in G for subsequent vertical distance calculation. The server will keep n copies of the global model w for each client, and update w_i with g_i then send to client$_i$. Cheat-training has no negative impact on normal clients.

We assume that the number of attackers in a cluster is always less than half. The attacks can generally divide into two types, S1: the vertical distance of attacker is greater than that of normal clients, S2: the vertical distance of attacker is smaller. To figure out the type of attack, the server uses the Euclidean distance-based K-Means algorithm to divide all clients in each cluster into two groups and treat clients in the smaller group as malicious. Comparing the means of the vertical distances in the two groups, suppose that a_m is the mean value of the suspected malicious group (smaller group), a_n is the mean value of another group (bigger group).

$$a = \frac{\sum_{i=1}^{k} d_t^i}{k} \tag{3}$$

If $a_m > a_n$, the attack can be classified into S1. The server will exclude those clients whose vertical distance d_t is more than $\tilde{m}$ times that of more than half of the clients in the same cluster. Otherwise, the attack type is S2, and clients whose vertical distance value is less than $1/\tilde{m}$ of the distance of

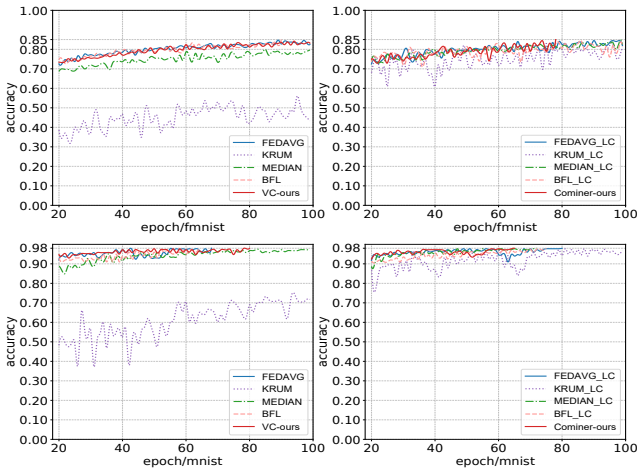

Figure 2: Comparison of the model accuracy before and after using LC of different robust algorithms.

more than half of clients in the same cluster will be excluded. Algorithm 2 shows the situation of S1.

Moreover, in LC process, attackers can lie about their label vectors to get into a wrong cluster or even an entire cluster with only attackers. This can result in a cluster with more malicious clients than normal ones. In order to solve this problem, we treat all clients as being in one big cluster and execute VC one more time in this virtual big cluster. Therefore, VC is effective as long as less than half of all clients are attackers, which is easy to guarantee in real scenarios.

4 EXPERIMENTS

In this section, we will evaluate the effect of Cominer on the optimization of FL and compare with some existing robust algorithms, including:

Krum [3]. The aggregation rule of Krum is to choose the gradient with smallest sum of Euclidean distances from other $m - q - 2$ clients is used as the aggregation result, where q and m is the number of attackers and all clients.

Median [26]. Calculate the median of each dimension of all gradients.

BFL [18]. All clients are partitioned into two clusters based on the gradients' cosine similarity and exclude one cluster according to the distance between the clients in two clusters.

We implement the FL system by PyTorch 1.8, and simulate 100 clients with each client uses the same three-layer full connected neural network. Besides, we implement 1 master process as the server. In each round of FL, each client will only perform one epoch of local training with the batch size of 32. The experiments involve two public datasets: MNIST [11] and Fashion-MNIST [23]. The threshold θ is set 0.5, $\tilde{m}$ is

Table 2: Attacks launched in different epochs

	REVERSE				CONSTANT				NORMAL			
	1	5	30	60	1	5	30	60	1	5	30	60
FEDAVG_LC	0.803	0.680	0.780	0.813	0.170	0.110	0.220	0.247	0.543	0.437	0.600	0.517
KRUM_LC	0.795	0.787	0.803	0.813	0.787	0.760	0.773	0.767	0.770	0.800	0.820	0.660
MEDIAN_LC	0.775	0.803	0.807	**0.837**	0.811	0.817	0.823	**0.843**	0.827	**0.810**	**0.840**	0.833
BFL_LC	0.810	0.777	0.623	0.820	0.100	0.583	0.413	0.687	0.595	0.453	0.643	0.507
Cominer-ours	**0.843**	**0.813**	**0.840**	**0.837**	**0.833**	**0.837**	**0.850**	**0.843**	**0.840**	0.805	0.833	**0.850**

set to 2 and the threshold of BFL is 0.02. All experiments are running on a computer with Intel i7-8550U CPU @1.8GHz.

No more than 50 clients are randomly selected as attackers and launch three different untargeted attacks, including: 1) **Reverse**, the attackers multiply their gradients by a negative constant(-2 in our experiments) before uploading to the server; 2) **Constant**, the malicious clients upload a constant gradient vector instead of the original gradient – the gradient is set to a constant vector of all ones; 3) **Normal**, the attacker randomly generate its gradient with a mean of 0 and a standard deviation of 1. It is worth emphasizing that the attackers in our experiments act independently and would not conduct cooperative attacks. Moreover, there are no sybil attacks[6], attackers will not forge multiple identities.

4.1 Results

We divide the experiments into two parts.

The first is LC mitigates the additional impact of robust algorithms on federated models: We first conduct experiments without attack, and the results are shown in Fig. 2.

We preset two test accuracy targets, 0.85 and 0.98, for the two datasets, and the training will stop when the test accuracy reaches the preset accuracy. The result shows that after adding LC into FL, the accuracy of the federated model obtained by using Krum and Median has improved. Krum's model accuracy has increased by 27.0% on FMNIST dataset and 22.7% on MNIST dataset, with an average increase of 24.85%. The model accuracy of Median increased by 3.5% on FMNIST dataset, and 1.0% on MNIST, with an average increase of 2.25%. Moreover, the addition of LC to federated training makes the federated model converge faster. The number of training rounds of Cominer reduces by 26.25% compare to use VC alone. Moreover, the federated model obtained by Cominer is the first to reach the preset accuracy compare to other robust algorithms.

The second is Cominer performs better at resisting attacks: To adapt to the more realistic situation, we conduct experiments with attackers present at different stage of FL to compare different algorithms under FMNIST dataset. The experimental results are shown in Table 2, and the three middle attack rounds are round 5, round 30 and round 60.

The results show that Cominer can almost achieve optimal results compared with other robust algorithms. The model accuracy of Cominer can reach more than 80% under any attack, which is higher than other algorithms. The results show that MEDIAN is the closest algorithm to Cominer and even better than Cominer in some cases. In most cases, the accuracy of the federated model obtained through Cominer can be optimal. We averaged the model accuracy of the three middle attack rounds. The average values are that the accuracy of Cominer is 7.57% higher than the worst robust algorithm and 2.80% higher than the best under REVERSE attack, 39.50% and 2.05% under CONSTANT attack, and 28.25% and 0.45% under NORMAL attack. The results prove that Cominer can deal well with attacks that appear at any stage of federated training. Our method can resist attacks stably and efficiently.

5 CONCLUSION AND FUTURE WORK

This paper aims at improving FL efficiency in the presence of attackers when data is non-i.i.d. We developed Cominer, which is composed of Label Cluster and Vertical Comparison. LC partitions all clients into different clusters according to the similarity of data labels, maintaining the diversity of data distributions by keeping more clients. VC can improve the robustness by identifying the difference between the vertical distances of the gradients uploaded by the clients. These two components afford Cominer both high training accuracy and high robustness. The experimental results show that Cominer successfully protects FL models from being damaged in case of attacks, and obtains an over 80% accuracy stably. Furthermore, LC can increases the average training accuracy under multiple datasets by 24.85% and 2.25% when compared with KRUM and MEDIAN, respectively, in normal cases with no attacks. In addition to untargeted attacks, federated learning systems can be vulnerable to many other kinds of attacks, such as targeted attacks and coordinated attacks by multiple malicious clients. We will explore the effectiveness of Cominer against these attacks in future research. Moreover, designing an effective attack method is also an interesting line of research.

REFERENCES

[1] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. 2020. How To Backdoor Federated Learning. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 108)*. PMLR, 2938–2948. https://proceedings.mlr.press/v108/bagdasaryan20a.html

[2] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. 2019. Analyzing Federated Learning through an Adversarial Lens. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*. PMLR, 634–643. https://proceedings.mlr.press/v97/bhagoji19a.html

[3] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. 2017. Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent. In *Advances in Neural Information Processing Systems*, Vol. 30. https://proceedings.neurips.cc/paper/2017/file/f4b9ec30ad9f68f89b29639786cb62ef-Paper.pdf

[4] Christopher Briggs, Zhong Fan, and Peter Andras. 2020. Federated learning with hierarchical clustering of local updates to improve training on non-IID data. In *2020 International Joint Conference on Neural Networks (IJCNN)*. 1–9. https://doi.org/10.1109/IJCNN48605.2020.9207469

[5] 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.

[6] John R Douceur. 2002. The sybil attack. In *International workshop on peer-to-peer systems*. Springer, 251–260.

[7] El Mahdi El Mhamdi, Rachid Guerraoui, and Sébastien Rouault. 2018. The Hidden Vulnerability of Distributed Learning in Byzantium. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*. PMLR, 3521–3530. https://proceedings.mlr.press/v80/mhamdi18a.html

[8] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. 2020. Local Model Poisoning Attacks to Byzantine-Robust Federated Learning. In *29th USENIX Security Symposium (USENIX Security 20)*. 1605–1622. https://www.usenix.org/conference/usenixsecurity20/presentation/fang

[9] Clement Fung, Chris JM Yoon, and Ivan Beschastnikh. 2018. Mitigating sybils in federated learning poisoning. *arXiv preprint arXiv:1808.04866* (2018).

[10] Fan Lai, Xiangfeng Zhu, Harsha V. Madhyastha, and Mosharaf Chowdhury. 2021. Oort: Efficient Federated Learning via Guided Participant Selection. In *15th USENIX Symposium on Operating Systems Design and Implementation (OSDI 21)*. 19–35. https://www.usenix.org/conference/osdi21/presentation/lai

[11] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324. https://doi.org/10.1109/5.726791

[12] Liping Li, Wei Xu, Tianyi Chen, Georgios B. Giannakis, and Qing Ling. 2019. RSA: Byzantine-Robust Stochastic Aggregation Methods for Distributed Learning from Heterogeneous Datasets. *Proceedings of the AAAI Conference on Artificial Intelligence* 33, 01 (Jul. 2019), 1544–1551. https://doi.org/10.1609/aaai.v33i01.33011544

[13] 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).

[14] Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. 2021. Ditto: Fair and Robust Federated Learning Through Personalization. In *Proceedings of the 38th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 139)*. PMLR, 6357–6368. https://proceedings.mlr.press/v139/li21h.html

[15] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. 2020. On the Convergence of FedAvg on Non-IID Data. In *International Conference on Learning Representations*. https://openreview.net/forum?id=HJxNAnVtDS

[16] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 54)*. PMLR, 1273–1282. https://proceedings.mlr.press/v54/mcmahan17a.html

[17] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. 2021. Clustered Federated Learning: Model-Agnostic Distributed Multitask Optimization Under Privacy Constraints. *IEEE Transactions on Neural Networks and Learning Systems* 32, 8 (2021), 3710–3722. https://doi.org/10.1109/TNNLS.2020.3015958

[18] Felix Sattler, Klaus-Robert Müller, Thomas Wiegand, and Wojciech Samek. 2020. On the Byzantine Robustness of Clustered Federated Learning. In *ICASSP IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 8861–8865. https://doi.org/10.1109/ICASSP40776.2020.9054676

[19] Jinhyun So, Başak Güler, and A. Salman Avestimehr. 2021. Byzantine-Resilient Secure Federated Learning. *IEEE Journal on Selected Areas in Communications* 39, 7 (2021), 2168–2181. https://doi.org/10.1109/JSAC.2020.3041404

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

[21] Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. 2022. Towards Personalized Federated Learning. *IEEE Transactions on Neural Networks and Learning Systems* (2022), 1–17. https://doi.org/10.1109/TNNLS.2022.3160699

[22] Hongyi Wang, Kartik Sreenivasan, Shashank Rajput, Harit Vishwakarma, Saurabh Agarwal, Jy-yong Sohn, Kangwook Lee, and Dimitris Papailiopoulos. 2020. Attack of the Tails: Yes, You Really Can Backdoor Federated Learning. In *Advances in Neural Information Processing Systems*, Vol. 33. 16070–16084. https://proceedings.neurips.cc/paper/2020/file/b8ffa41d4e492f0fad2f13e29e1762eb-Paper.pdf

[23] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747* (2017).

[24] Chulin Xie, Keli Huang, Pin-Yu Chen, and Bo Li. 2020. DBA: Distributed Backdoor Attacks against Federated Learning. In *International Conference on Learning Representations*. https://openreview.net/forum?id=rkgyS0VFvr

[25] Yihao Xue, Chaoyue Niu, Zhenzhe Zheng, Shaojie Tang, Chengfei Lyu, Fan Wu, and Guihai Chen. 2021. Toward Understanding the Influence of Individual Clients in Federated Learning. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 12 (May 2021), 10560–10567. https://ojs.aaai.org/index.php/AAAI/article/view/17263

[26] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. 2018. Byzantine-Robust Distributed Learning: Towards Optimal Statistical Rates. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*. PMLR, 5650–5659. https://proceedings.mlr.press/v80/yin18a.html

[27] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. 2018. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582* (2018).