Suppressing Poisoning Attacks on Federated Learning for Medical Imaging

Naif Alkhunaizi, Dmitry Kamzolov, Martin Takáč, and Karthik Nandakumar(✉)

Mohamed bin Zayed University of Artificial Intelligence, Abu Dhabi, UAE
{naif.alkhunaizi,kamzolov.dmitry,martin.takac,
karthik.nandakumar}@mbzuai.ac.ae

Abstract. Collaboration among multiple data-owning entities (e.g., hospitals) can accelerate the training process and yield better machine learning models due to the availability and diversity of data. However, privacy concerns make it challenging to exchange data while preserving confidentiality. Federated Learning (FL) is a promising solution that enables collaborative training through exchange of model parameters instead of raw data. However, most existing FL solutions work under the assumption that participating clients are *honest* and thus can fail against poisoning attacks from malicious parties, whose goal is to deteriorate the global model performance. In this work, we propose a robust aggregation rule called Distance-based Outlier Suppression (DOS) that is resilient to byzantine failures. The proposed method computes the distance between local parameter updates of different clients and obtains an outlier score for each client using Copula-based Outlier Detection (COPOD). The resulting outlier scores are converted into normalized weights using a softmax function, and a weighted average of the local parameters is used for updating the global model. DOS aggregation can effectively suppress parameter updates from malicious clients without the need for any hyperparameter selection, even when the data distributions are heterogeneous. Evaluation on two medical imaging datasets (CheXpert and HAM10000) demonstrates the higher robustness of DOS method against a variety of poisoning attacks in comparison to other state-of-the-art methods. The code can be found at https://github.com/Naiftt/SPAFD.

Keywords: Federated learning · Parameter aggregation · Malicious clients · Outlier suppression

1 Introduction

Medical institutions often seek to leverage their data to build deep learning models for predicting different diseases [10,23]. However, limited access to data can impede the learning process [27] or introduce bias towards the local data [18]. Hence, collaborative learning is vital to expand data availability and maximize model performance. Federated Learning (FL) [20] provides a paradigm

L. Wang et al. (Eds.): MICCAI 2022, LNCS 13438, pp. 673–683, 2022.
https://doi.org/10.1007/978-3-031-16452-1_64

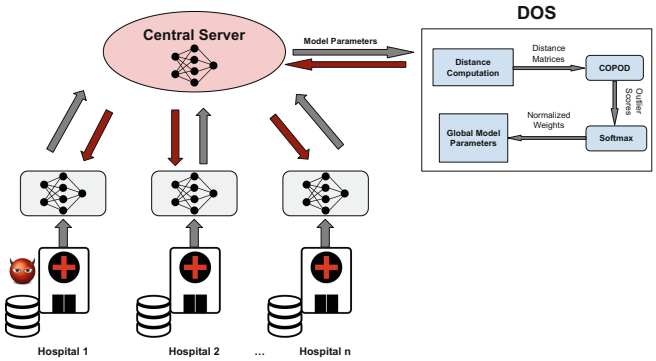

Fig. 1. Federated Learning (FL) in the presence of a malicious client. Distance-based Outlier Suppression (DOS) aggregation rule (right) can be applied to robustly aggregate local parameter updates.

where multiple institutions/devices can use their data to train local models and exchange their model parameters regularly with a central server. The server aggregates the local model parameters to update a global model, which is shared back to the clients. This allows all parties to preserve their data locally and offers better results while maintaining data privacy. Recent works have underscored the importance of FL in medical imaging [9,30].

Typically, FL algorithms (e.g., FedAvg [24]) operate under the assumption that clients participate in the protocol honestly and their data distributions are identical. However, the competitive nature of the participants makes them vulnerable to poisoning attacks. Real-world FL is susceptible to a variety of targeted/untargeted attacks [12,34] and byzantine faults [19]. Techniques proposed to tackle non-iid data [20] cannot handle malicious parties that attempt to deteriorate the global model performance. Though many robust aggregation rules have been proposed for byzantine-tolerant FL [5,14,33], such methods often require a careful choice of hyperparameters (e.g., prior knowledge of proportion of malicious clients) and are vulnerable to more sophisticated attacks [11].

The core contribution of this work is a novel robust aggregation rule called Distance-based Outlier Suppression (DOS) for byzantine-tolerant FL. The proposed framework (see Fig. 1) enables the central server to suppress local parameter updates from malicious clients during aggregation. The DOS framework is enabled by computing distances (Euclidean and cosine) between local parameter updates of different clients and detecting outliers in this distance space. A state-of-the-art, parameter-free outlier detection algorithm called COPOD [21] is applied to compute the outlier score for each client based on the above distances. Finally, the outlier scores are mapped to client-specific weights, and weighted average of local parameter updates is utilized for global model update. The proposed DOS aggregation rule demonstrates high robustness against diverse poisoning attacks under both iid (CheXpert) and non-iid (HAM10000) settings.

2 Related Work

Most FL algorithms aim to minimize the following loss function:

$$\min_{\theta \in \mathbb{R}^d} \{\mathbf{F}(\theta) := \sum_{i=1}^{n} \alpha_i \mathbf{F}_i(\theta)\}, \quad (1)$$

where n is the number of clients, F_i is the loss function for the i^{th} client, and α_i is a value between 0 and 1 that weights the contribution of the i^{th} client. Popular aggregation rules such as FedAvg [24] either assign equal weights to all clients ($\alpha_i = 1/n, \forall\ i = 1, 2, \ldots, n$) or assign weights to the clients based on the relative size of their training sets. Such schemes have shown effective outcomes under the assumption of honest clients and iid data distributions. FedProx [20] was introduced to tackle heterogeneity and non-iid data across clients. Progressive Fourier Aggregation (PFA) was introduced in [6] to improve stability by preventing an abrupt drop in accuracy. While these methods promise convergence, they do not consider noisy parameters [32] or malicious parties that attempt to hinder learning or cause it to converge to poor local minima.

Since FL requires communication between the server and clients, it is susceptible to random network errors that can deliver abnormal parameters to the server or malicious parties that attempt to corrupt the learning process. Such failures are defined as Byzantine failures [19]. In such settings, there are two types of attacks: (i) targeted attacks that cause the global model to misclassify a selected class (e.g., backdoor attacks [2], model poisoning attacks [3,4], and data poisoning attacks [25]) and (ii) untargeted attacks that harm the model's performance across all classes [20]. Moreover, there are techniques designed to make the attack more stealthy [34]. Other potential attacks include label-flipping, where the model is trained on incorrect labels [12], and crafted model attacks [11] that formulate the attack as an optimization problem. In this paper, we mainly focus on untargeted attacks since they are more lethal to the learning process.

Several robust aggregation rules have been proposed to defend against Byzantine failures. Krum [5] selects the parameter update that is closest to the mean parameter update of all clients. Coordinate-Wise-Median [33] calculates the median because it is less sensitive to outliers as opposed to the mean. Trimmed Mean [33] sorts the parameter values and discards a fraction of each coordinate's smallest and largest values. Bulyan was proposed in [14] to further improve the robustness. However, most of these methods require hyperparameter selection (e.g., proportion of outliers to be discarded) and are susceptible to crafted model attacks [11]. More recently, Robust Federated Aggregation (RFA) [28], which relies on aggregation using the geometric median, and SparseFed [26], which uses global top-k update sparsification and device-level gradient clipping, have been proposed to mitigate poisoning attacks.

3 Proposed Method

We propose Distance-based Outlier Suppression (**DOS**), a robust FL aggregation rule (see Algorithm 1) that can defend against different untargeted poisoning

Algorithm 1. Distance-based Outlier Suppression (DOS)

Require: Initialize parameter vector $\theta^0 \in \mathbb{R}^d$ for global model

for $t = 0, 1, \ldots, T-1$ **do**
  Server: Server broadcasts θ^t to all clients
  for all $i \in [n]$ in parallel **do**
    Client i: Learn local parameters θ_i^{t+1} using suitable local optimizer.
    Send $\hat{\theta}_i^{t+1}$ to server. Note that $\hat{\theta}_i^{t+1} \neq \theta_i^{t+1}$ for malicious clients.
  end for
  Server: **DOS** aggregation rule:
    Compute Euclidean and cosine distance matrices M_E and M_C, respectively
    Compute outlier scores $\mathbf{r}_E = COPOD(M_E)$, $\mathbf{r}_C = COPOD(M_C)$
    Compute average outlier score $\mathbf{r} = (\mathbf{r}_E + \boldsymbol{r}_C)/2$
    Compute normalized client weights as $w_i = \frac{\exp(-r_i)}{\sum_{j=1}^{n} exp(-r_j)}$
    Update global model based on weighted average of local parameters

$$\theta^{t+1} = \sum_{i=1}^{n} w_i \hat{\theta}_i^{t+1}$$

end for

attacks on FL as long as the proportion of clients experiencing byzantine-failures is less than 50%. Suppose that there are n clients in the FL setup, and their goal is to learn the global model parameters θ that minimizes the objective function in Eq. (1). Note that a proportion p of these clients ($p < 50\%$) can be malicious and may not share the same objective as the other honest clients. Similar to FedAvg, the global model parameters are initialized to θ^0. In each of the T communication rounds, the current global model parameters θ^t are broadcast to all clients. Each client $i \in [n]$ learns the local model parameters θ_i^{t+1} based on their local data using a suitable optimizer and sends $\hat{\theta}_i^{t+1}$ back to the server. While $\hat{\theta}_i^{t+1}$ is expected to be a faithful version of θ_i^{t+1} (except for known transformations like compression) for honest clients, this may not be the case for malicious clients. The server computes the updated global model parameters using the DOS aggregation rule. The proposed DOS aggregation rule consists of three key steps:

Distance Computation: The server starts with calculating the Euclidean and cosine distances between the local parameters sent by the clients as follows:

$$d_{ij}^E = ||\hat{\theta}_i - \hat{\theta}_j||_2, \qquad d_{ij}^C = 1 - \frac{\hat{\theta}_i^T \hat{\theta}_j}{||\hat{\theta}_i||_2 ||\hat{\theta}_j||_2},$$

where $i, j = 1, 2, \ldots, n$. Here, the time index t is skipped for convenience. The $(n \times n)$ distance matrices $M_E = [d_{ij}^E]$ and $M_C = [d_{ij}^C]$ are then computed and utilized for outlier detection. Unlike existing methods such as [5,14,33], where outlier detection is performed directly on the model parameter space, the proposed DOS method performs outlier detection in the distance space.

Outlier Score Computation: After exploring various anomaly detection methods such as Random Forest [29], Local Outlier Factor (LOF) [7], and K-means [22], we selected the Copula Based Outlier Detection (COPOD) method [21] due to several factors. Firstly, other methods require choosing the percentage of abnormal data points (in our case clients) in advance, whereas COPOD is parameter-free, thus making it more robust. Moreover, COPOD is known to be computationally efficient even in high dimensional settings because it utilizes the marginals of the joint distributions, thereby allowing for greater flexibility and individual modeling of each dimension. Finally, the COPOD method also has desirable interpretability properties since it is based on modeling the tail probabilities along each dimension. In general, the COPOD function takes a $(n \times d)$ matrix as input, where each row represents a sample and each column represents an attribute of the sample to produce a n-dimensional vector of outlier scores $\mathbf{r} = [r_1, r_2, \ldots, r_n]$, where $r_i \in (0, +\infty)$ represents the relative likelihood that the sample i is an outlier (true outliers are expected to have larger values of r_i). In DOS, we compute $\mathbf{r}_E = COPOD(M_E)$ and $\mathbf{r}_C = COPOD(M_C)$ and average these two vectors to obtain the final outlier score $\mathbf{r} = (\mathbf{r}_E + \mathbf{r}_C)/2$.

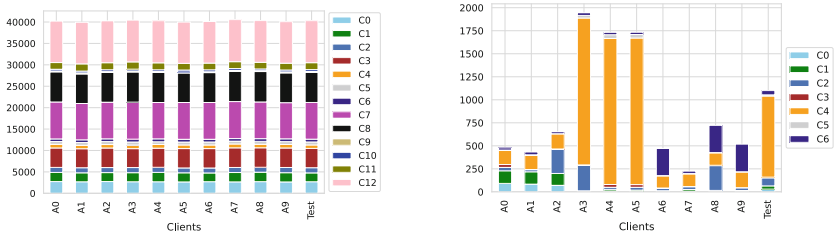

Fig. 2. Left: Distribution of the multiclass multilabel CheXpert dataset among clients where each stacked bar represents the number of positive cases for class 0 to 12. **Right:** Distribution of the multiclass HAM1000 dataset in non-iid case among clients where each stacked bar represents the number of samples for class 0 to 6.

Weighted Average Aggregation: Ideally, the local parameter updates of those clients with higher outlier scores must be suppressed and the local parameters of clients with lower outlier scores must be amplified. To achieve this goal, we apply a softmax function to the outlier score vector with a temperature parameter of -1 and use the resulting output as the normalized weights for each client, i.e.,

$$w_i = \frac{exp(-r_i)}{\sum_{j=1}^{n} exp(-r_i)}, \quad i = 1, 2, \cdots, n.$$

The local parameters of the clients are aggregated using the following rule:

$$\theta^{t+1} = \sum_{i=1}^{n} w_i \hat{\theta}_i^{t+1}.$$

4 Experiments

4.1 Datasets

CheXpert: CheXpert is a large chest X-ray dataset with uncertainty labels and expert comparison [17]. We use 'CheXpert-small', a multi-class multi-label dataset that contains 191,456 chest X-ray images for training. The dataset is imbalanced and has 13 pathology categories. In addition, it has a percentage of uncertain data, and [17] suggests either ignoring the uncertain labels during training or mapping all instances to zero or one. In our experiments, all uncertain labels were mapped to zero. The TorchXRayVision library [8] was used to preprocess the data, where every image has a 224×224 resolution. We divide the training images equally between the clients as shown in Fig. 2 (**Left**), where the last stacked bar represents the testing dataset that we use to measure the global model's performance after each round. We use the CheXpert dataset to evaluate our method in the iid setting.

Table 1. Area under the receiver operating characteristic curve (AUC) with different types of poisoning attack scenarios on the Chexpert dataset. T-M stands for Trimmed Mean [33]. The last column represents the average AUC over five presented scenarios.

	No attack	Label flip 10%	Mix attack 40%	Noise 40%	Noise & Scaled 40%	Avg
FedAvg	0.70	0.69	0.50	0.50	0.50	0.58
Median	0.70	0.69	0.69	0.69	0.54	0.66
T-M	0.70	0.69	0.69	0.50	0.50	0.62
Krum	0.66	0.66	0.64	0.63	0.67	0.65
DOS	**0.70**	**0.69**	0.68	**0.69**	**0.67**	**0.68**

HAM10000: HAM10000 is a multi-class dataset consisting of 10,015 dermoscopic images from different populations [31]. It consists of 7 categories of pigmented lesions where every image was resized to 128×128. The train-test split is 8,910 and 1,105 images, respectively. We use this dataset to test our method in the non-iid case as shown in Fig. 2 (**Right**).

4.2 FL Setup

In all our experiments, we assume $n = 10$ clients. The well-known ResNet-18 [16] model, as well as a custom Convolutional Neural Network (CNN) with two convolutional layers, a ReLU activation function, and two fully connected layers are used as the architectures for both global and local models. Each client only has access to its local dataset and the local models are trained using Stochastic Gradient Descent (SGD) [13] with a learning rate of 0.01. We implement all our experiments using the Pytorch framework [1]. Using a batch size of 16 for CheXpert, each client is trained for a total of 100 rounds with 1 local step in

each round that goes through the entire dataset. As for the HAM10000 dataset, we iterate through the whole dataset with a batch size of 890 for each client. We train it for a total of 250 rounds with 5 local steps for each client. This experiment was performed to show that the batch size does not impact the DOS aggregation rule. The evaluation metric is the Area Under the Receiver operating characteristic curve (AUC), which is calculated after each round of communication on the testing dataset. For the CheXpert dataset [17], the AUC was calculated for each class and the macro average was taken over all classes. For the HAM1000 dataset [31], the average AUC was computed for all possible pairwise class combinations to minimize the effect of imbalance [15].

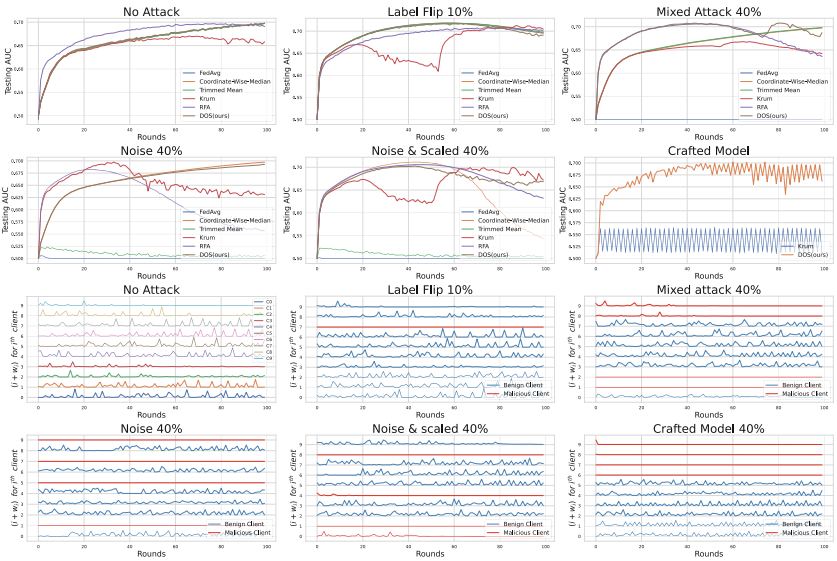

Fig. 3. Performance on CheXpert dataset using ResNet-18 model: The first two rows show AUC on a test set after each round for six scenarios in the following order (left to right): No Attack, Label Flip 10%, Mix Attack 40%, Noise 40%, Noise and Scaled 40%, and Crafted Attack 40%. The bottom two rows show the normalized weights of each client after each round.

4.3 Poisoning Attacks and Baseline Aggregation Rules

We assume that up to $p \leq 40\%$ of these clients could be malicious, i.e., at most 4 out of the 10 clients are malicious. We consider 5 different attacks on the CheXpert dataset: (i) **Label Flip** 10% - label-flipping by one of the clients, (ii) **Mix Attack** 40% - transmission of Gaussian noise by two clients and label-flipping by two clients, (iii) **Noise** 40% - transmission of Gaussian noise by four clients, (iv) **Noise and Scaled** 40% - transmission of a mix of Gaussian noise and scaled parameters by four clients, and (v) **Crafted** 40% - crafted model attack with four clients based on [11] (with the aggregation rule being

unknown to the attacker). On the HAM1000 dataset, 2 attacks were considered: (i) **Noise** 30% - transmission of Gaussian noise by three clients, and (ii) **Mix Attack** 40% - transmission of Gaussian noise by two clients, scaled parameter by a factor of 100 by one client, and scaled parameter by a factor of -0.5 (directionally opposite to the true parameters) by one client. The robustness of the proposed DOS method was benchmarked against the following aggregation rules: FedAvg [24], Coordinate-Wise-Median [33], Trimmed Mean [33] and Krum [5]. For both datasets, a **No Attack** scenario was also considered to evaluate the performance of DOS when all the clients are benign (honest).

4.4 Results and Discussion

Figure 3 compares all the aggregation rules under six different scenarios on the CheXpert dataset. Table 1 summarizes the average AUC for all the aggregation rules and it can be observed that DOS aggregation rule performs consistently well against all attack scenarios. In contrast, all the other aggregation rules had lower accuracy values in one or more scenarios. Furthermore, it can be observed from Fig. 3 that the weights of the malicious clients (shown in red) are almost always close to zero, indicating that the DOS method is able to effectively suppress the parameter updates from malicious clients. The performance of DOS method is also comparable to that of RFA [28], with DOS method having a marginal edge when the proportion of malicious clients was higher (40%). Only in the case where 40% of clients transmitted Gaussian noise, the RFA method had a significantly lower accuracy (0.625) compared to the DOS method (0.69).

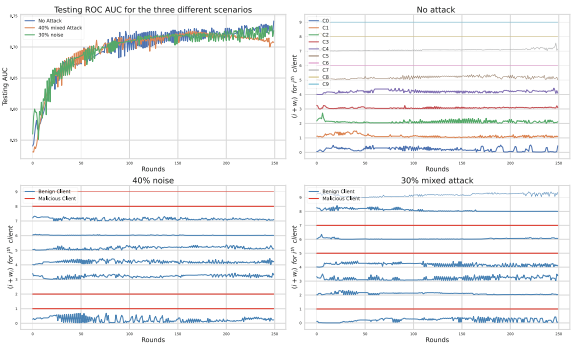

Fig. 4. Performance on HAM10000 dataset using a custom CNN model. The top-left plot shows AUC on the test set for DOS aggregation for the 3 scenarios. The normalized weights of each client after each round for the 3 scenarios are also depicted.

Figure 4 shows the performance of the DOS rule under three different scenarios on the HAM10000 dataset. In all cases, DOS performed with a high AUC score without dropping its overall performance. The main advantage of DOS lies in its ability to account for both Euclidean and cosine distance, thereby

addressing both positive and negative scaling. It also successfully detects Gaussian noise attacks and label-flip attacks, leveraging the strengths of the COPOD method. Since DOS combines three different approaches, it is hard to design model poisoning attacks that can successfully circumvent it.

Proportion of Malicious Clients: Since our approach treats malicious weight updates as outliers, the protocol will fail when a majority of clients are malicious. We conducted experiments by fixing the number of clients and increasing the proportion of malicious clients from 10% to 60%, in steps of 10%. As expected, the DOS approach was robust until the proportion of malicious clients was less than or equal to 50% and failed when the proportion was 60% (e.g., for HAM10000 dataset, the AUC values were 0.695, 0.697, 0.696, 0.711, 0.710, and 0.554 for 10%, 20%, 30%, 40%, 50%, and 60% corruption, respectively. In comparison, the AUC without any attack was 0.70). This is the reason for choosing the proportion of malicious clients as 40% for most of our experiments. Thus, the DOS method is appropriate only for the honest majority scenario.

Different Number of Clients: We conducted experiments by fixing the proportion of malicious clients to 40% and increasing the number of clients from 5 to 40. The AUC values were 0.725, 0.700, 0.692, and 0.674 for 5, 10, 20 and 40 clients, respectively. We observe a minor degradation in accuracy when the number of clients increases and it requires more rounds to converge. Therefore, the DOS approach may be more suitable for cross-silo FL settings.

Non-IID Setting: We have conducted experiments under five different non-iid settings, where we randomly partition the HAM10000 dataset among 10 clients. For all the five experiments, the convergence trends and the final accuracy were similar (between 0.69 and 0.71) to the iid case, showing that the DOS method can work well in the non-iid setting.

4.5 Conclusions and Future Work

We presented Distance-based Outlier Suppression (DOS), a novel robust aggregation rule that performs outlier detection in the distance space to effectively defend against Byzantine failures in FL. We proved the effectiveness and robustness of our method using two real-world medical imaging datasets. In future, we aim to prove theoretical convergence and extend this framework to ensure fairness between FL clients, based on the contribution of their local datasets.

References

1. PyTorch: an imperative style, high-performance deep learning library
2. Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D., Shmatikov, V.: How to backdoor federated learning. In: International Conference on Artificial Intelligence and Statistics, pp. 2938–2948. PMLR (2020)

3. Bhagoji, A.N., Chakraborty, S., Mittal, P., Calo, S.: Model poisoning attacks in federated learning. In: Proceedings of Workshop on Security Machine Learning (SecML) 32nd Conference Neural Information Processing Systems (NeurIPS), pp. 1–23 (2018)
4. Bhagoji, A.N., Chakraborty, S., Mittal, P., Calo, S.: Analyzing federated learning through an adversarial lens. In: International Conference on Machine Learning, pp. 634–643. PMLR (2019)
5. Blanchard, P., El Mhamdi, E.M., Guerraoui, R., Stainer, J.: Machine learning with adversaries: Byzantine tolerant gradient descent. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
6. Chen, Z., Zhu, M., Yang, C., Yuan, Y.: Personalized retrogress-resilient framework for real-world medical federated learning. In: de Bruijne, M., et al. (eds.) MICCAI 2021. LNCS, vol. 12903, pp. 347–356. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-87199-4_33
7. Cheng, Z., Zou, C., Dong, J.: Outlier detection using isolation forest and local outlier factor. In: Proceedings of the Conference on Research in Adaptive and Convergent Systems, pp. 161–168 (2019)
8. Cohen, J.P., et al.: TorchXRayVision: a library of chest X-ray datasets and models (2020). https://github.com/mlmed/torchxrayvision, https://github.com/mlmed/torchxrayvision
9. Dayan, I., et al.: Federated learning for predicting clinical outcomes in patients with COVID-19. Nat. Med. **27**(10), 1735–1743 (2021)
10. Esteva, A., et al.: Dermatologist-level classification of skin cancer with deep neural networks. Nature **542**(7639), 115–118 (2017)
11. Fang, M., Cao, X., Jia, J., Gong, N.: Local model poisoning attacks to {Byzantine-Robust} federated learning. In: 29th USENIX Security Symposium (USENIX Security 2020), pp. 1605–1622 (2020)
12. Fu, S., Xie, C., Li, B., Chen, Q.: Attack-resistant federated learning with residual-based reweighting. arXiv preprint arXiv:1912.11464 (2019)
13. Gardner, W.A.: Learning characteristics of stochastic-gradient-descent algorithms: a general study, analysis, and critique. Signal Process. **6**(2), 113–133 (1984)
14. Guerraoui, R., Rouault, S., et al.: The hidden vulnerability of distributed learning in byzantium. In: International Conference on Machine Learning, pp. 3521–3530. PMLR (2018)
15. Hand, D.J., Till, R.J.: A simple generalisation of the area under the roc curve for multiple class classification problems. Mach. Learn. **45**(2), 171–186 (2001)
16. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
17. Irvin, J., et al.: CheXpert: a large chest radiograph dataset with uncertainty labels and expert comparison. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 590–597 (2019)
18. Kaushal, A., Altman, R., Langlotz, C.: Health care AI systems are biased. Scientific American, vol. 17 (2020)
19. Lamport, L.: The weak Byzantine generals problem. J. ACM (JACM) **30**(3), 668–676 (1983)
20. Li, T., Sahu, A.K., Talwalkar, A., Smith, V.: Federated learning: challenges, methods, and future directions. IEEE Signal Process. Mag. **37**(3), 50–60 (2020)
21. Li, Z., Zhao, Y., Botta, N., Ionescu, C., Hu, X.: COPOD: copula-based outlier detection. In: 2020 IEEE International Conference on Data Mining (ICDM), pp. 1118–1123. IEEE (2020)

22. Likas, A., Vlassis, N., Verbeek, J.J.: The global k-means clustering algorithm. Pattern Recogn. **36**(2), 451–461 (2003)
23. Liu, Z., Xiong, R., Jiang, T.: Clinical-inspired network for skin lesion recognition. In: Martel, A.L., et al. (eds.) MICCAI 2020. LNCS, vol. 12266, pp. 340–350. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-59725-2_33
24. McMahan, B., Moore, E., Ramage, D., Hampson, S., Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial Intelligence and Statistics, pp. 1273–1282. PMLR (2017)
25. Muñoz-González, L., et al.: Towards poisoning of deep learning algorithms with back-gradient optimization. In: Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, pp. 27–38 (2017)
26. Panda, A., Mahloujifar, S., Nitin Bhagoji, A., Chakraborty, S., Mittal, P.: SparseFed: mitigating model poisoning attacks in federated learning with sparsification. In: Proceedings of AISTATS, pp. 7587–7624 (2022)
27. van Panhuis, W.G., et al.: A systematic review of barriers to data sharing in public health. BMC Public Health **14**, 1144 (2014)
28. Pillutla, K., Kakade, S.M., Harchaoui, Z.: Robust aggregation for federated learning. IEEE Trans. Signal Process. **70**, 1142–1154 (2022)
29. Primartha, R., Tama, B.A.: Anomaly detection using random forest: a performance revisited. In: 2017 International Conference on Data and Software Engineering (ICoDSE), pp. 1–6. IEEE (2017)
30. Sheller, M.J., et al.: Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data. Nat. Sci. Rep. **10**(1), 12598 (2020)
31. Tschandl, P., Rosendahl, C., Kittler, H.: The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. Sci. Data **5**(1), 1–9 (2018)
32. Wei, X., Shen, C.: Federated learning over noisy channels: convergence analysis and design examples. IEEE Trans. Cogn. Commun. Netw. **8**(2), 1253–1268 (2022)
33. Yin, D., Chen, Y., Kannan, R., Bartlett, P.: Byzantine-robust distributed learning: Towards optimal statistical rates. In: International Conference on Machine Learning, pp. 5650–5659. PMLR (2018)
34. Zhou, X., Xu, M., Wu, Y., Zheng, N.: Deep model poisoning attack on federated learning. Future Internet **13**(3), 73 (2021)