2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA) | 978-1-6654-1623-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/TPSISA52974.2021.00022

The Impact of Data Distribution on Fairness and Robustness in Federated Learning

Mustafa Safa Ozdayi
Department of Computer Science
The University of Texas at Dallas
Richardson, TX, USA
mustafa.ozdayi@utdallas.edu

Murat Kantarcioglu
Department of Computer Science
The University of Texas at Dallas
Richardson, TX, USA
muratk@utdallas.edu

***Abstract*—Federated Learning (FL) is a distributed machine learning protocol that allows a set of agents to collaboratively train a model without sharing their datasets. This makes FL particularly suitable for settings where data privacy is desired. However, it has been observed that the performance of FL is closely related to the similarity of the local data distributions of agents. Particularly, as the data distributions of agents differ, the accuracy of the trained models drop. In this work, we look at how variations in local data distributions affect the *fairness* and the *robustness* properties of the trained models in addition to the accuracy. Our experimental results indicate that, the trained models exhibit higher bias, and become more susceptible to attacks as local data distributions differ. Importantly, the degradation in the fairness, and robustness can be much more severe than the accuracy. Therefore, we reveal that small variations that have little impact on the accuracy could still be important if the trained model is to be deployed in a fairness/security critical context.**

***Keywords*-Federated Learning, Algorithmic Fairness, Adversarial Machine Learning**

I. Introduction

Federated Learning (FL) [1] is a distributed machine learning protocol. Through FL, a set of agents can collaboratively train a model without sharing their data with each other, or any other third party. This makes FL suitable to settings where data privacy is desired. In this regard, FL differs from the traditional distributed learning setting in which data is first centralized at a place, and then distributed to the agents [2, 3].

The decentralized nature of datasets in FL brings some unique challenges. In particular, several works have observed that the accuracy of the trained models can drop significantly when the data distributions of the agents are different (the so-called non-i.i.d. setting), and have tried to mitigate this negative effect [4, 5, 6, 7, 8, 9].

To to best of our knowledge, the existing literature have almost exclusively analyzed the impact of data distributions on the accuracy. In contrast, in this work, we look at how data distributions of agents affect the *fairness* and *robustness* of the trained models. That is, we ask, *how does the fairness and robustness properties of the trained models change with the distribution of agents' datasets?* In this work, we quantify the bias exhibited by the model based on the class-wise accuracies, (see Section III-A), and quantify the robustness based on the resilience against *model poisoning* attacks (see Section II-B and III-B).

Briefly, our analysis reveals that, just as for accuracy, fairness and robustness of the trained models degrade as the distribution of local datasets differ. More importantly though, we show that this degradation can be much more severe than the degradation of the accuracy. We believe these observations can be important in several contexts. For example, the existing defenses in the FL literature are usually tested against different ratios of adversarial agents. However, as we show that the models become more susceptible to attacks as local data distributions change, it is also important to test the effectiveness of the defenses under different local data distributions.

The rest of the paper is organized as follows: in Section II, we provide the necessary background and discuss the related works. In Section III, we provide our experimental results, and show how robustness and fairness of the trained models change with the local data distributions. Finally in Section IV, we discuss our results and provide some concluding remarks.

II. Background and Related Work

In this section, we provide the necessary background on FL and explain model poisoning attacks.

A. Federated Learning (FL)

At a high level, FL is multi-round protocol between an aggregation server and a set of agents in which agents jointly train a model. Formally, participating agents try to minimize the average of their loss functions,

$$\arg\min_{w \in R^d} f(w) = \frac{1}{K} \sum_{k=1}^{K} f_k(w),$$

where f_k is the loss function of kth agent. For example, for neural networks, f_k is typically empirical risk minimization under a loss function L such as cross-entropy, i.e.,

978-1-6654-1623-8/21/$31.00 ©2021 IEEE
DOI 10.1109/TPSISA52974.2021.00022

$$f_k(w) = \frac{1}{n_k} \sum_{j=1}^{n_k} L(x_j, y_j; w),$$

with n_k being the total number of samples in agent's dataset and (x_j, y_j) being the jth sample.

Concretely, FL protocol is executed as follows: at round t, server samples a subset of agents S_t, and sends them w_t, the model weights for the current round. Upon receiving w_t, kth agent initializes his model with the received weight, and trains for some number of iterations, e.g., via stochastic gradient descent (SGD), and ends up with weights w_t^k. The agent then computes his update as $\Delta_t^k = w_t^k - w_t$, and sends it back to the server. Upon receiving the update of every agent in S_t, server computes the weights for the next round by aggregating the updates with an aggregation function $\mathbf{g}\colon R^{|S_t| \times d} \rightarrow R^d$ and adding the result to w_t. That is, $w_{t+1} = w_t + \eta \cdot g(\{\Delta_t\})$ where $\{\Delta_t\} = \cup_{k \in S_t} \Delta_t^k$, and η is the server's learning rate. For example, original FL paper [1] and many subsequent papers on FL [10, 11, 12, 13, 14] consider weighted averaging to aggregate updates. In this context, this aggregation is referred as Federated Averaging (FedAvg), and yields the following update rule,

$$w_{t+1} = w_t + \eta \frac{\sum_{k \in S_t} n_k \cdot \Delta_t^k}{\sum_{k \in S_t} n_k}.$$

In practice, rounds can go on indefinitely, as new agents can keep joining the protocol, or until the model reaches some desired performance metric (e.g., accuracy) on a validation dataset maintained by the server.

It has been shown that models trained via FL can perform better than locally trained models at agents' side in various settings [1, 15]. In contrast, as noted before, it has also been observed that the performance of FL drops drastically when local data distributions of agents differ significantly, i.e., when data is distributed in a non-i.i.d. fashion among agents [4, 5, 6, 7, 8, 9].

B. Backdoor Attacks and Model Poisoning

Training time attacks against machine learning models can roughly be classified into two categories: targeted [10, 11, 16, 17], and untargeted attacks [18, 19]. In untargeted attacks, the adversarial task is to make the model converge to a sub-optimal minima or to make the model completely diverge. Such attacks have been also referred as *convergence attacks*, and to some extend, they are easily detectable by observing the model's accuracy on a validation data.

On the other hand, in targeted attacks, adversary wants the model to misclassify only a set of chosen samples with minimally affecting its performance on the main task. Such targeted attacks are also known as *backdoor attacks*. A prominent way of carrying backdoor attacks is through *trojans* [16, 17]. A trojan is a carefully crafted pattern that is leveraged to cause the desired misclassification. For example, consider a classification task over cars and planes and let the adversarial task be making the model classify blue cars as plane. Then, adversary could craft a brand logo, put it on *some* of the blue car samples in the training dataset, and only mislabel those as plane. Due to this attack, the attacked model would potentially learn to classify a blue car with the brand logo as a plane. At the inference time, adversary can present a blue car sample with the logo to the model to activate the backdoor. Ideally, since the model would behave correctly on blue cars that do not have the trojan, it would not be possible to detect the backdoor on a clean validation dataset.

In FL, the training data is decentralized and the aggregation server is only exposed to model updates. Given that, backdoor attacks are typically carried by constructing malicious updates. That is, adversary tries to create an update that encodes the backdoor in a way such that, when it is aggregated with other updates, the aggregated model exhibits the backdoor. This has been referred as *model poisoning* attack [10, 11, 12]. For example, an adversary could control some of the participating agents in a FL instance and train their local models on trojaned datasets to construct malicious updates.

III. Experiments

We now show how fairness and robustness of the trained models are impacted due to differences in local data distributions via experiments. The general setting of our experiments are as follows: we are training models via FL for classification tasks among 10 agents where in each around, an agent trains his local model for 2 epochs with a batch size of 256 with the Adam optimizer [20]. We train until the model converges on the training dataset, and do our measurements on a held-out test dataset.

We use two datasets: Fashion MNIST [21], and CIFAR10 [22]. On Fashion MNIST, we use the LeNet model [23], and on CIFAR10, we use a 6-layer convolutional neural network, consisting of 3 convolutional and 3 fully-connected layers.

To simulate different data distributions, we distribute the initial datasets to agents using Dirichlet distribution with different concentration parameters (denoted as α) as in [24]. Higher values of α indicate the distribution of agents' local datasets over the classes in the dataset are more similar.

Our implementation is in PyTorch [25], and our code is publicly available at https://github.com/TinfoilHat0/Fairness-Robustness-in-FedLearning. All the reported results are averaged over 5 runs, and are presented in mean $\pm$ std format.

A. Fairness

We quantify the bias of the model as the difference between the highest and the lowest accuracy across the

Table I: Effect of data distribution on fairness for Fashion MNIST (top), and CIFAR10 (bottom) datasets.

Dirichlet α	Accuracy	Bias
1.0	90.08 ± 0.17	24.48 ± 1.84
0.5	89.06 ± 0.31	37.3 ± 5.83
0.25	88.02 ± 1.81	45.78 ± 25.62
Dirichlet α	Accuracy	Bias
1.0	76.52 ± 0.76	29.03 ± 2.3
0.5	75.17 ± 0.62	35.6 ± 6.04
0.25	72.27 ± 1.81	42.6 ± 14.79

classes. That is, if M_c is model $M's$ accuracy on class c, we have that,

$$\text{Bias}_M = \max_c (M_c) - \min_c (M_c),$$

where the*higher values for bias indicates the model is less fair across classes.*

Our results are presented in the Table I. As can be seen from these results, bias exhibited by the model increases as local dataset distributions differ more. Interestingly though, the rate of increase in the bias seems much higher than the degradation in the accuracy. For example, for Fashion MNIST dataset, we see that, as we go from $\alpha = 1$ to $\alpha = 0.25$, the overall accuracy drops by merely about 2%, but the bias of the model almost doubles.

B. Robustness

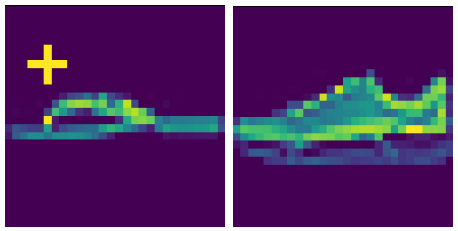
We now test how robustness of the model changes against *model poisoning* attacks for different data distributions. In this experiment, we designate one of the agents (out of ten) as the adversary. We fix his dataset to 1000 samples uniformly sampled from the initial training dataset. The corrupt agent carries a backdoor attack by (i) adding a trojan to pattern to his samples, (ii) and mislabeling all his dataset as a chosen target class (see Figure 1). We measure the success of adversary's attack (denoted as *backdoor accuracy*) by measuring the accuracy of the model on the poisoned test dataset (whose samples have the trojan pattern, and labeled as the target class chosen by the adversary). *Lower values for backdoor accuracy indicates the model is more robust.*

We present the results for robustness in Table II. As can be seen, the trained models become more susceptible to attack as local data distributions become less similar. The results seem more pronounced on Fashion MNIST than on CIFAR10. We believe this is due to the fact that, the attack is already quite successful on $\alpha = 1$ CIFAF10, so, there is less room to improve the attack in that setting.

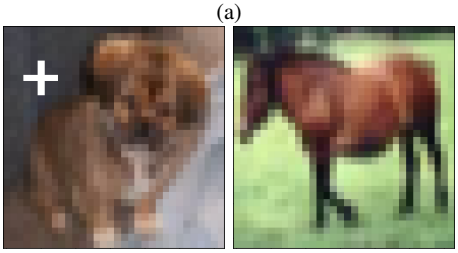
Regardless, we believe the main takeaway of our result is that, any defense mechanism deployed in the FL setting should be tested against different data distributions, in addition to different adversarial agents ratios as is commonly done.

Table II: Effect of data distribution on robustness for Fashion MNIST (top), and CIFAR10 (bottom) datasets.

Dirichlet α	Accuracy	Backdoor Accuracy
1.0	89.87 ± 0.14	64.34 ± 5.12
0.5	89.58 ± 0.2	69.78 ± 2.58
0.25	88.47 ± 0.77	77.34 ± 2.84
Dirichlet α	Accuracy	Backdoor Accuracy
1.0	75.72 ± 0.34	89.66 ± 0.81
0.5	74.64 ± 0.13	90.37 ± 0.4
0.25	71.75 ± 1.45	91.49 ± 0.68



(a)



(b)

Figure 1: Samples from trojaned base classes and corresponding target classes. Trojan pattern is a 5-by-5 plus sign that is put to the top-left of objects. For Fashion MNIST (a), the backdoor task is to make model classify trojaned inputs as sneakers. For CIFAR10 (b), it is to make model classify trojaned samples as horses. For Fashion MNIST, we note that original images are in grayscale, and these figures are normalized as they appear in training/test datasets.

C. Robustness under a Defense

Based on our insight from the robustness experiments, we now look at how the robustness is impacted when a defense is deployed under different distributions. To do so, we deploy a recent defense introduced in [26] called *robust learning rate* (RLR). This approach merely modifies the learning rate of aggregation server, per round and per dimension, based on the sign information of agents' updates, and is agnostic to the aggregation function itself.

Table III: Effect of data distribution on robustness under RLR defense for Fashion MNIST (top), and CIFAR10 (bottom) datasets.

Dirichlet α	Accuracy	Backdoor Accuracy
1.0	88.99 ± 0.26	0.48 ± 0.64
0.5	87.96 ± 0.79	3.62 ± 2.88
0.25	84.12 ± 1.4	0.38 ± 0.44
Dirichlet α	Accuracy	Backdoor Accuracy
1.0	74.09 ± 0.54	20.22 ± 6.66
0.5	69.38 ± 1.75	13.26 ± 3.09
0.25	57.07 ± 3.01	29.89 ± 20.99

Concretely, the defense introduces a hyperparameter called the *learning threshold*, denoted as θ at the server-side. For every dimension where the sum of signs of updates is less than θ, the learning rate is multiplied by -1. This is *to maximize the loss on that dimension rather than minimizing it*. That is, with a learning threshold of θ, the learning rate for the ith dimension is given by,

$$\eta_{\theta,i} = \begin{cases} \eta & \left|\sum_{k\in S_t} \operatorname{sgn}(\Delta^k_{t,i})\right| \geq \theta, \\ -\eta & \text{otherwise.} \end{cases}$$

For example, consider FedAvg and let η_θ denote the learning rate vector over all dimensions, i.e., $[\eta_{\theta,1}, \eta_{\theta,2}, \ldots, \eta_{\theta,d}]^\top$. Then, the update rule with the RLR defense takes the form,

$$w_{t+1} = w_t + \eta_\theta \odot \frac{\sum_{k\in S_t} n_k \cdot \Delta^k_t}{\sum_{k\in S_t} n_k},$$

where $\odot$ is the element-wise product operation.

The experimental results under this defense is presented in Table III. If we compare Table II (robustness with no defense), and Table III, we see that the defense diminishes the attack significantly as indicated by the smaller backdoor accuracies. However, at least for CIFAR10 case, the defense becomes substantially weak for $\alpha = 0.25$ as evidenced by the low accuracy, and high backdoor accuracy values. Therefore, we can see that local data distributions can substantially tamper with the robustness of the model, even when a defense is deployed.

D. Interplay between Fairness and Robustness

We also look at how a defense might affect the fairness of the model. For example, we see that the RLR defense substantially reduces the backdoor accuracy, but it comes at the cost of some accuracy. We wonder, whether the defense itself can degrade the fairness of the model, and if it does, how this degradation is impacted by the data distributions. Our experiments with RLR indicates that, the defense itself might make the model less fair, and this effect is increased as local data distributions differ more. In Table IV, we present

Table IV: Interplay between fairness and robustness under backdoor attack without any defense for Fashion MNIST (top), and CIFAR10 (bottom) datasets.

Dirichlet α	Accuracy	Backdoor Accuracy	Bias
1.0	89.87 ± 0.14	64.34 ± 5.12	29.1 ± 4.57
0.5	89.58 ± 0.2	69.78 ± 2.58	32.8 ± 10.11
0.25	88.47 ± 0.77	77.34 ± 2.84	33.88 ± 4.5
Dirichlet α	Accuracy	Backdoor Accuracy	Bias
1.0	75.72 ± 0.34	89.66 ± 0.81	35.16 ± 3.96
0.5	74.64 ± 0.13	90.37 ± 0.4	32.02 ± 4.75
0.25	71.75 ± 1.45	91.49 ± 0.68	56.5 ± 20.66

Table V: Interplay between fairness and robustness under backdoor attack with RLR defense for Fashion MNIST (top), and CIFAR10 (bottom) datasets.

Dirichlet α	Accuracy	Backdoor Accuracy	Bias
1.0	88.99 ± 0.26	0.48 ± 0.64	31.84 ± 5.83
0.5	87.96 ± 0.79	3.62 ± 2.88	32.62 ± 7.78
0.25	84.12 ± 1.4	0.38 ± 0.44	51.56 ± 10.93
Dirichlet α	Accuracy	Backdoor Accuracy	Bias
1.0	74.09 ± 0.54	20.22 ± 6.66	45.52 ± 7.18
0.5	69.38 ± 1.75	13.26 ± 3.09	63.1 ± 8.81
0.25	57.07 ± 3.01	29.89 ± 20.99	90.26 ± 1.36

the bias of the model under the model poisoning attack without any defense. In Table V, we present the bias of the model under the model poisoning attack with RLR defense. As can be seen, we observe that the bias exhibited by the model increases under the defense even though the attack itself is much less successful.

IV. Discussion and Conclusion

In this work, we have explored how the differences in local data distributions of the participating agents in a FL setting affect the fairness and robustness properties of the trained models. Our brief experimental analysis have indicated that, just as for accuracy, as the local distributions among agents differ, the fairness and robustness of the trained models degrades. Further, we have showed that these properties might degrade much faster than the accuracy.

Regardless, our work is a limited exploration of the topic, and we believe there are some avenues for further work. Most importantly, we wonder whether the existing algorithms that are developed to remedy the accuracy drop in non-i.i.d. settings [4, 5, 6, 7, 8, 9], can remedy the fairness and robustness issues as well. Also, the robustness notions we considered in our work was merely against model poisoning attacks. It might be interesting to see how robustness of the models change against privacy attacks [27].

Acknowledgment

The research reported herein was supported in part by NSF awards, CNS-1837627, OAC-1828467, IIS-1939728, DMS-1925346, CNS-2029661, OAC-2115094 and ARO award W911NF-17-1-0356

REFERENCES

[1] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. Communication-efficient learning of deep networks from decentralized data. *arXiv preprint arXiv:1602.05629*, 2016.

[2] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marc'aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, et al. Large scale distributed deep networks. In *Advances in neural information processing systems*, pages 1223–1231, 2012.

[3] Mu Li, David G Andersen, Jun Woo Park, Alexander J Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J Shekita, and Bor-Yiing Su. Scaling distributed machine learning with the parameter server. In *11th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 14)*, pages 583–598, 2014.

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

[5] Hao Wang, Zakhary Kaplan, Di Niu, and Baochun Li. Optimizing federated learning on non-iid data with reinforcement learning. In *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, pages 1698–1707, 2020. doi: 10.1109/INFOCOM41043.2020.9155494.

[6] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127*, 2018.

[7] Mustafa Safa Ozdayi, Murat Kantarcioglu, and Rishabh Iyer. Improving accuracy of federated learning in non-iid settings. *arXiv preprint arXiv:2010.15582*, 2020.

[8] Neta Shoham, Tomer Avidor, Aviv Keren, Nadav Israel, Daniel Benditkis, Liron Mor-Yosef, and Itai Zeitak. Overcoming forgetting in federated learning on non-iid data. *arXiv preprint arXiv:1910.07796*, 2019.

[9] Kevin Hsieh, Amar Phanishayee, Onur Mutlu, and Phillip Gibbons. The non-iid data quagmire of decentralized machine learning. In *International Conference on Machine Learning*, pages 4387–4398. PMLR, 2020.

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

[11] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2938–2948, 2020.

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

[13] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191, 2017.

[14] Robin C Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*, 2017.

[15] Andrew Hard, Chloé M Kiddon, Daniel Ramage, Francoise Beaufays, Hubert Eichner, Kanishka Rao, Rajiv Mathews, and Sean Augenstein. Federated learning for mobile keyboard prediction, 2018. URL https://arxiv.org/abs/1811.03604.

[16] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.

[17] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. In *25nd Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-221, 2018*. The Internet Society, 2018.

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

[19] Jeremy Bernstein, Jiawei Zhao, Kamyar Azizzadenesheli, and Anima Anandkumar. signsgd with majority vote is communication efficient and fault tolerant. *arXiv preprint arXiv:1810.05291*, 2018.

[20] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

[22] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). 2009. URL http://www.cs.toronto.edu/~kriz/cifar.html.

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

[24] Harsh Bimal Desai, Mustafa Safa Ozdayi, and Murat Kantarcioglu. Blockfla: Accountable federated learning via hybrid blockchain architecture. In *Proceedings of the Eleventh ACM Conference on Data and Application*

Security and Privacy, pages 101–112, 2021.

[25] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[26] Mustafa Safa Ozdayi, Murat Kantarcioglu, and Yulia R Gel. Defending against backdoors in federated learning with robust learning rate. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 9268–9276, 2021.

[27] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting unintended feature leakage in collaborative learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 691–706. IEEE, 2019.