Analyzing Poisoning Attacks on Heterogeneous Federated Learning Systems

Sai Vineeth Doddala sdoddala@uci.edu
Rajasekhar Reddy Mekala rmekala@uci.edu
Agniraj Baikani abaikani@uci.edu

Abstract

Federated learning (FL) is a rapidly evolving distributed machine learning paradigm in which participant's data remains across multiple devices which are usually heterogeneous, with only model updates being shared with a central server. However, the distributed nature of FL gives rise to new threats caused by potentially malicious participants. For example, in the training process of federated learning, some malicious nodes can attack the whole training process by sending model updates derived from mislabeled data.

Although a lot of approaches have been proposed for FL training to improve convergence, little has been known about their robustness under malicious attacks. In this work, we compare the performance of popular aggregation algorithms like FedAvg, FedProx, and FedProto against various data poisoning attacks (namely label flipping, model inversion and evasion attacks), and validate their robustness in heterogeneous settings. We also investigate the susceptibility of algorithms to minor attacks from large number of clients against large attacks on few clients. Initial results indicate that while FedProx and FedProto converge faster than FedAvg during normal training, preformance drop is significant for FedProto under malicious attacks in IID settings.

1 Introduction

Algorithmic breakthroughs, viable collection of large scale data, and ever improving computing systems have contributed to the rise of distributed machine learning. Today, many of the companies deploy ML based predictive models across their workflows to improve personalization and user satisfaction. However, there are severe privacy implications associated with machine learning, as the trained model incorporates essential information about the training set. (Ateniese et al., 2015). (Fredrikson et al., 2015) show that sensitive information can be extracted easily from the trained models. These security and privacy concerns has led to fundamental changes in training large-scale ML models while preserving their efficacy.

Federated learning has become the paradigm to offload the training of models to the edge devices like phones and tablets (McMahan et al., 2017).The global model is jointly learnt over data distributed on multiple devices without a centralized data curator that collects and aggregates the dataset. All the individual clients send model parameter values to a central parameter server to create the global model. The key advantage of this approach is the decoupling of model training from the direct access to the raw training data. For applications where the training objective can be specified locally to each device, federated learning reduces security risks by limiting the attack surface to only the client although the data heterogeneity and computational limitations present significant challenges.

Since FL involves aggregation of parameters across multiple clients to learn global parameters, Several synchronous approaches based on simple averaging FedAvg (McMahan et al., 2017), treating the task as a distributed multi-task learning FedProx (Li et al., 2020), merging the prototypes instead of parameters FedProto (Tan et al., 2022) and many asynchronous approaches (Liu et al., 2021), (Chen et al., 2020) have been proposed to improve the aggregation procedure.

While FL systems have provided a way to maintain the raw data at local devices ensuring data privacy, it has led to several other security issues due to the heterogeneity of both edge computing and federated learning. As the edge nodes and edge servers are usually distributed across the globe, not all of them can be trusted. The risk of malicious attacks (Schlesinger et al., 2018) increased significantly as there is no direct validation of the training data during local parameter updates. Figure 1 shows the default architecture of such attacks. Hackers can manipulate the global model with access to significant small edge devices by corrupting local edge nodes.

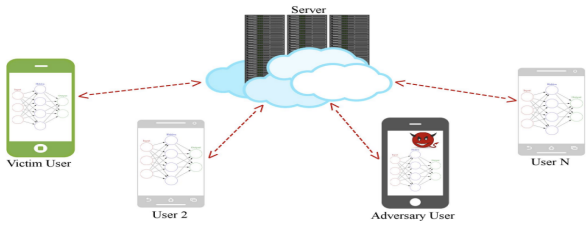

Figure 1: Malicious attacks in collaborative learning settings - Federated learning (Hitaj et al., 2017)

Much work has been done to address security issues in edge federated learning. Although there are many kinds of attacks (Membership Inference (Shokri et al., 2017), Label Leakage (Li et al., 2021a)) possible in these systems (discussed in section 3), Byzantine attacks (Gaussian, Omniscient and Flip bit attacks) and poisoning attacks (data and model poisoning) are the most popular methods and many defense mechanisms have been proposed for the same. In this work, we compare various proposed parameter aggregation algorithms on *MNIST* dataset in heterogeneous environments and validate their performance under various attacks.

2 Motivation

Over the years, the community has come up with many parameter aggregation approaches. Some of them focus on convergence (McMahan et al., 2017), personalization (Arivazhagan et al., 2019), heterogenity (Li et al., 2020). While there have been defenses proposed against specific attacks (Hitaj et al., 2017), (Li et al., 2021a), little is known about their robustness over a group of attacks and quantitative measures of loss in performance. Hence, In this work we plan to

- Analyze the performance of algorithms like FedAvg, FedProx, FedProto against data and model poisoning attacks for robustness.
- Study the loss in performance with varying attack severity and quantify the impact of heterogenity sourced in malicious attacks.
- Identify the heterogenity settings that can cause more damage to the models for the algorithms in study.

3 Related Work

Federated Learning(FL) has been able to address the fundamental problems of privacy, ownership, and locality of data. As training happens in a distributed fashion, many algorithms have been proposed to combine gradients from the distributed devices. One of the first ones is FedAvg Algorithm (McMahan et al., 2016), a heuristic method that combines local stochastic gradient descent (SGD) of each device on a server that performs model averaging. Analyzing FedAvg is a challenging task since the FL setup comes with statistical heterogeneity(non-identically distributed data) and system heterogeneity(network, computation). FedProx (Li et al., 2020), a federated optimization algorithm tackles these challenges theoretically and empirically by using local regularization to optimize each client's local model. Other solutions to handle heterogeneity include FedProto (Tan et al., 2022) where abstract class prototypes are shared between devices and server instead of the gradients. Nonetheless, the distributed nature of FL gives rise to many threats caused by potentially malicious participants. There are some solutions that cluster the local models (Mansour et al., 2020) but self-supervision (Li et al., 2021b), meta-training (Fallah et al., 2020) approaches also have shown optimistic signs and looks to be the trend in personalization

Poisonous attacks are commonly studied in ML problems like anomaly detection (Chen et al., 2017), (Mozaffari-Kermani et al., 2015), spam filtering (Nelson et al., 2008), and recommendations (Fang et al., 2018), (Yang et al., 2017). Popular ML models like SVM (Biggio et al., 2012), dimensionality reduction (Xiao et al., 2018), unsupervised learning (Biggio et al., 2018), and neural networks (Muñoz-González et al., 2017) are found to be compromised by poisonous attacks. However, all these problems are studied in a traditional ML setting where training happens in a centralized server. Poisonous attacks in the context of FL are different because the malicious device only has access to the data/model it holds. Also, the defense strategies like anomaly detection at the centralized server won't work as the distributed device only shares accumulated values and not individual data. Poisonous attacks in FL are broadly divided into model poisoning and data poisoning. Most of the work in this area is done on model poisoning (Zhang et al., 2022), (Sun et al., 2021), (Panda et al., 2021), where a malicious FL device tampers its training process to send adversarial parameters and gradients to the centralized server. Data poisoning on the other hand is a process where a malicious FL device modifies the training data by adding or changing the existing data instances. Data poisoning is preferable since it doesn't involve tampering with the learning process which requires the expertise of the model. Very little work has been done on data poisoning in FL setup (Tolpegin et al., 2020) where the effect of Label Flipping attack on a single FL algorithm is studied. On the contrary, this work focuses on poisoning attacks like Evasion Attack (Biggio et al., 2013) and Model Inversion along with Label Flipping on three different FL algorithms namely FedAvg, FedProx, and FedProto. The robustness of these algorithms against the above mentioned poisoning attacks is studied in a heterogeneous setup which is not been touched upon before.

4 Design

4.1 Algorithms

As introduced before, federated learning enables the computation of local models on client nodes and the aggregation of these models on a server to produce a new, more generic model that is sent back to all the clients. A key point in federated learning is the way specialized models are aggregated at the server. This approach is presented by the FedAvg, FedProx, and FedProto aggregation algorithms, which will be detailed next.

4.1.1 FedAvg

FedAvg (McMahan et al., 2017) algorithm, which consists of a few local alternating stochastic gradient updates at client nodes, followed by a model averaging update at the server. The FedAvg starts with the random initialization of a neural model on a device or on a central server in charge of managing model transfers with client devices. When on-device training is finished, the weights of the local models are sent to the server. The Federated Averaging (FedAvg) algorithm is probably the most widely used method in a Federated setting.

While FedAvg has demonstrated empirical success in heterogeneous settings, it does not fully address the underlying

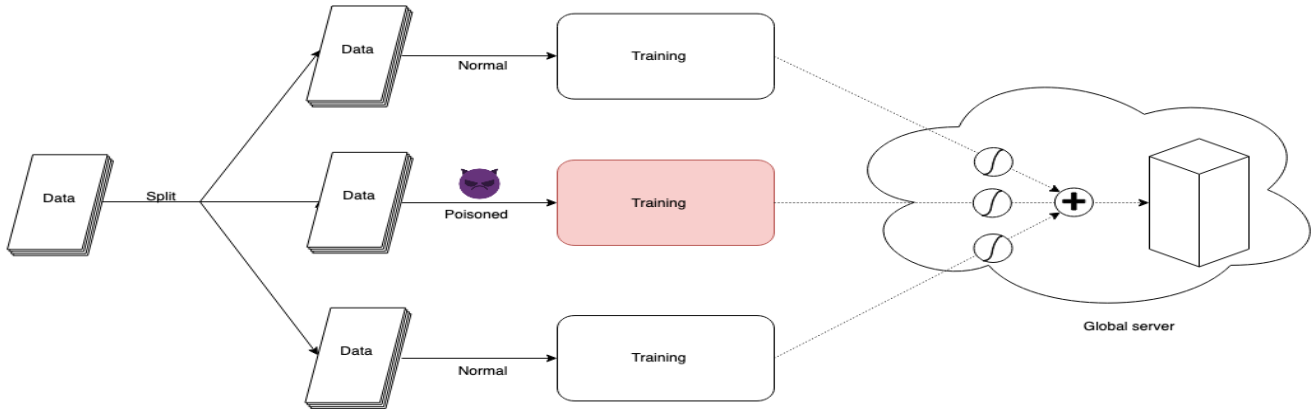

Figure 2: Dataflow in the FL setup

challenges associated with heterogeneity. Here, aggregation is done in a weighted averaging manner, where clients with more data significantly influence the newly aggregated model. For instance, with Non-independent and Identically Distributed (Non-IID) data, weights may be updated entirely differently due to statistical heterogeneity. Thus, averaging all weights that are drastically different causes decremental results. In the context of system heterogeneity, FedAvg does not allow participating devices to perform variable amounts of local work based on the constraints of their underlying systems; instead, it simply drops devices that fail to compute in time.

4.1.2 FedProx

FedProx (Li et al., 2020), can be viewed as a generalization and re-parametrization of FedAvg. The FedProx algorithm is similar to FedAvg in that a subset of devices is selected at each round, local updates are performed, and these updates are then averaged to form a global update. However, FedProx makes the following simple yet critical modifications, which result in overall improvements and convergence. Here, we generalize FedAvg by allowing for variable amounts of work to be performed locally across devices based on their available system resources, as compared to dropping these devices. In FedProx, we add a proximal term to the local subproblem to effectively limit the impact of variable local updates, as too many local updates may potentially cause the methods to diverge due to the underlying heterogeneous data.

FedProx optimization addresses the system and statistical heterogeneity inherent in federated networks. In particular, in highly heterogeneous settings, FedProx demonstrates more stable and accurate convergence behavior relative to FedAvg.

4.1.3 FedProto

In FedProto (Tan et al., 2022), the clients and server communicate the abstract class prototypes instead of the gradients. FedProto aggregates the local prototypes collected from different clients, and then sends the global prototypes back to all clients to regularize the training of local models. The training on each client aims to minimize the classification error on the local data while keeping the resulting local prototypes sufficiently close to the corresponding global ones. FedProto's improved tolerance to heterogeneity can be explained from three perspectives: model inference, communication efficiency, and privacy- preservation.

The FedProto method only transmits prototypes between the server and clients. In general, the size of the prototypes is usually much smaller than the size of the model parameters, resulting in efficient communication. This property also brings benefits to FL in terms of preserving privacy. First, prototypes naturally protect data privacy, because they are 1D vectors generated by averaging the low-dimensional representations of samples from the same class, which is an irreversible process. Second, attackers cannot reconstruct raw data from prototypes without access to local models.

4.2 Dataset and model

We implement the typical federated learning setting where each client owns its local data and transmits and receives information to and from the central server. We used the popular MNIST dataset (LeCun and Cortes, 2010) (70,000 samples, 10 labels) for the experimentation. In addition, for MNIST, a multi-layer CNN classification model with two convolutional layers and two fully connected layers was considered.

4.3 Attacks

For experimentation, a diverse set of attacks were chosen, primarily to understand the performance degradation across all the attack paradigms.

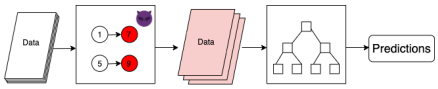

Figure 3: Label Flipping

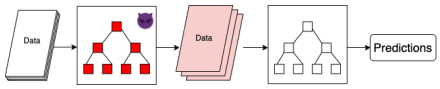

Figure 4: Model based Gradient Inversion

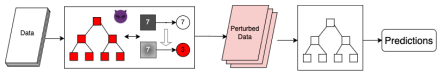

Figure 5: Evasion Attack

4.3.1 Label Flipping

The attacker in the Label Flipping attack flips the output labels. Each malicious participant P_i alters their dataset D_i as follows, given a source class c_{src} and a target class c_{target} from C: Change the output label of malicious instances in D_i that are c_{src} to c_{target}. Figure-3 depicts the design of the Label Flipping attack in the system. For example, in the MNIST dataset, images belonging to the digit 5 are mislabeled as 9, which is the most similar-looking digit. It is not computationally complex, unlike other attacks, so it saves time and energy, which is appealing given that FL is frequently performed on edge devices. It is also simple for non-experts to carry out and does not necessitate any changes or manipulation of participant-side FL software.

4.3.2 Model Inversion - Gradient inversion

Model inversion attacks are possible only with direct access to the local model on the edge nodes. Although there are many variants of model based attacks, the gradient-based model inversion attack was chosen as it involves direct perturbations to the parameter update rules of the local model. Figure-4 accurately describes the mechanism of this attack. The attacker precomputes the gradients on batches of data and updates the datapoints before feeding the actual edge model. These perturbed datapoints lead to inverse updates in the local gradients and eventually mislead the overall learning objective of the global model.

4.3.3 Evasion Attack

Evasion attacks aim to cheat the target model by constructing specific samples called adversarial examples. Usually, some subtle noise added to the input samples cannot be detected by human beings and causes the model to give incorrect classification results. Here, the attacker knows the white-box model of the targets and has test instances in terms of their local datasets. The malicious client generates perturbed images to deceive other clients as well. Figure-5 illustrates the design of the evasion attack in the system. For example, in the MNIST dataset, an attacker perturbs images belonging to the digit 7 to misclassify them as 3. Due to the similarity of training data, algorithms, and model architecture in federated learning systems, client-to-client transfer evasion attacks may have a high success rate. Depending on the optimization objective, evasion attacks can be divided into targeted attacks with class-specific errors and untargeted attacks that do not consider these errors.

4.4 Training

4.4.1 Setup

Figure-2 represents flow of the data in the FL setup. The data is split among the clients in an identical or non-identical manner. A few malicious clients are chosen at random, and attacks are carried out on the data in that client. Label Flipping, Model Inversion and Evasion attacks are used to modify the training data in the clients, which directly affects the model training happening on the FL clients. This results in changes to the gradients and parameters sent from FL clients to the central aggregator.

4.4.2 Hyperparameters

To analyze the robustness of the FL aggregation algorithms under the attacks mentioned above, experiments are performed with a varied range of resources. The number of clients ranges from 10 to 50 (10, 30, and 50 clients). In one of the settings, 10%, 30%, and 50% of the total clients are designated as malicious clients. In each malicious client, 1%, 3%, and 5% of the data available to the client is poisoned.

4.4.3 Implementation

FedAvg, FedProx, and FedProto FL aggregation methods are implemented in PyTorch. All the experiments were run on a GPU with 32 cores and 20 GB of RAM. Twenty clients are used for the MNIST dataset to distribute data in IID (Independent and Identically Distributed) and Non-IID settings. Data distributions to 20 clients and server aggregations were generated using multithreading. All FL methods are adapted to fit this statistically heterogeneous setting in the implementation. In the Non-IID setting, it is assumed that all clients perform learning tasks with heterogeneous statistical distributions. In order to simulate different real-world data heterogeneity scenarios, heterogeneity has been created in both class spaces and data sizes per client.

For heterogeneous distribution of client data, two Non-IID settings are being used. The first one is the pathological Non-IID setting, where, for example, the data on each client only contains a specific number of labels (maybe only two labels), even though the data on all clients contains 10 labels, such as in the MNIST dataset. The second example is a non-IID practical setting in which Dirichlet distribution is used to generate disjoint client training data.

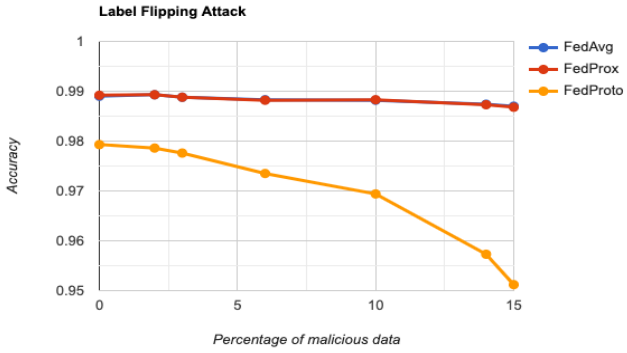

Figure 6: Label Flipping

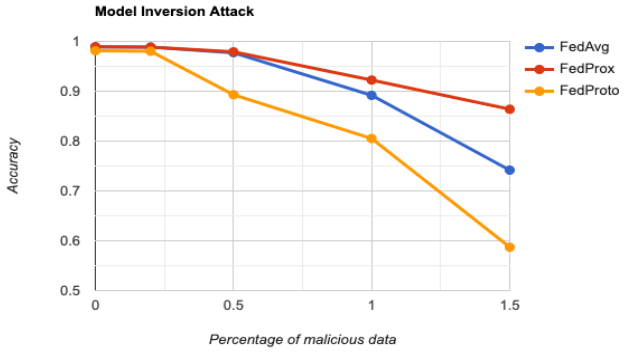

Figure 7: Model based Gradient Inversion

Figure 8: Performance of FedAvg, FedProx, FedProto on various attacks

5 Results

5.1 IID settings

Currently, we have performed experiments on label flipping and model inversion attacks for all 3 algorithms for IID-distributed data on clients. The impact of a label-flipping attack is depicted in Figure-6. We observe that while the impact of label flipping attacks is comparatively low, accuracy drops noticeably when there is roughly 10–15 percent data corruption. This indicates that label flipping attacks require a huge amount of data corruption to significantly impact the training procedure, which is practically not possible for attackers as the FL echosystem has millions of edge nodes. Furthermore, we notice that FedAvg and FedProx are much more robust to this type of attack.

Similar experiments performed on the gradient inversion attack indicate that it only requires about 1-2% of the overall data to significantly impact the performance of the ML model. While the performance of FedProto significantly degrades, FedAvg and FedProx are also not robust to this attack. It is also possible to carry out such an attack using the very large edge nodes available today. Along with the performance degradation studies, experiments are also performed by varying the total number of clients. As expected, the convergence process takes much longer as the number of clients increases. Since the experiments involved a fixed number of global iterations, the performance dropped with an increase in clients. But Figure-9 shows that the performance drop due to the malicious attack remains constant even with a changing number of clients and depends much more on the percentage of malicious data.

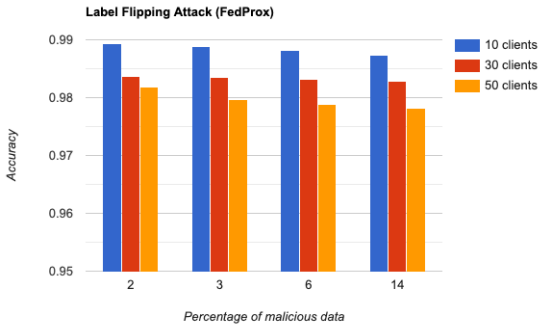

Figure 9: Comparision of performance with number of edge nodes for FedProx

5.2 Heterogeneous settings

Apart from these experiments, we also compare the performance of the models in unbalanced settings without any attack. In case of IID and balanced setting, we notice that both FedAvg and FedProx aggregation strategies converge to high accuracy. But with the heterogeneous distribution of client data in the Non-IID case, FedProx performs slightly better than FedAvg. Overall, from Table 1, we can observe that FedProto achieves the highest accuracy and the least variance in most cases, ensuring uniformity among heterogeneous clients.

Data distribution settings	FedAvg	FedProx	FedProto
For practical Non-IID and unbalanced setting (Dirichlet distribution, alpha=0.1)	0.9692	0.9700	0.9934
For pathological Non-IID and unbalanced setting	0.9396	0.9403	0.9979
For IID and balanced setting	0.9890	0.9892	0.9753

Table 1: Comparison of FL algorithms with heterogeneity test accuracy values. MINIST data split across 20 clients for CNN classification with unbalanced/balanced distributions and ran for 100 global update rounds each.

6 Discussion

7 Conclusion

References

Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. Federated learning with personalization layers. *arXiv preprint arXiv:1912.00818*, 2019.

Giuseppe Ateniese, Luigi V Mancini, Angelo Spognardi, Antonio Villani, Domenico Vitali, and Giovanni Felici. Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. *International Journal of Security and Networks*, 10(3):137–150, 2015.

Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines, 2012. URL https://arxiv.org/abs/1206.6389.

Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Š rndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Advanced Information Systems Engineering*, pages 387–402. Springer Berlin Heidelberg, 2013. doi: 10.1007/978-3-642-40994-3_25. URL https://doi.org/10.1007%2F978-3-642-40994-3_25.

Battista Biggio, Ignazio Pillai, Samuel Rota Bulò, Davide Ariu, Marcello Pelillo, and Fabio Roli. Is data clustering in adversarial settings secure? 2018. doi: 10.48550/ARXIV.1811.09982. URL https://arxiv.org/abs/1811.09982.

Sen Chen, Minhui Xue, Lingling Fan, Shuang Hao, Lihua Xu, Haojin Zhu, and Bo Li. Automated poisoning attacks and defenses in malware detection systems: An adversarial machine learning approach, 2017. URL https://arxiv.org/abs/1706.04146.

Yujing Chen, Yue Ning, Martin Slawski, and Huzefa Rangwala. Asynchronous online federated learning for edge devices with non-iid data. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 15–24. IEEE, 2020.

Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. *Advances in Neural Information Processing Systems*, 33:3557–3568, 2020.

Minghong Fang, Guolei Yang, Neil Zhenqiang Gong, and Jia Liu. Poisoning attacks to graph-based recommender systems. In *Proceedings of the 34th Annual Computer Security Applications Conference*. ACM, dec 2018. doi: 10.1145/3274694.3274706. URL https://doi.org/10.1145%2F3274694.3274706.

Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1322–1333, 2015.

Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. Deep models under the gan: information leakage from collaborative deep learning. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pages 603–618, 2017.

Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010. URL http://yann.lecun.com/exdb/mnist/.

Oscar Li, Jiankai Sun, Xin Yang, Weihao Gao, Hongyi Zhang, Junyuan Xie, Virginia Smith, and Chong Wang. Label leakage and protection in two-party split learning. *arXiv preprint arXiv:2102.08504*, 2021a.

Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10713–10722, 2021b.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2:429–450, 2020.

Yinghui Liu, Youyang Qu, Chenhao Xu, Zhicheng Hao, and Bruce Gu. Blockchain-enabled asynchronous federated learning in edge computing. *Sensors*, 21(10):3335, 2021.

Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020.

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

H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. 2016. doi: 10.48550/ARXIV.1602.05629. URL https://arxiv.org/abs/1602.05629.

Mehran Mozaffari-Kermani, Susmita Sur-Kolay, Anand Raghunathan, and Niraj K. Jha. Systematic poisoning

attacks on and defenses for machine learning in healthcare. *IEEE Journal of Biomedical and Health Informatics*, 19(6): 1893–1905, 2015. doi: 10.1109/JBHI.2014.2344095.

Luis Muñoz-González, Battista Biggio, Ambra Demontis, Andrea Paudice, Vasin Wongrassamee, Emil C. Lupu, and Fabio Roli. Towards poisoning of deep learning algorithms with back-gradient optimization, 2017. URL https://arxiv.org/abs/1708.08689.

Blaine Nelson, Marco Barreno, Fuching Jack Chi, Anthony D. Joseph, Benjamin I. P. Rubinstein, Udam Saini, Charles Sutton, J. D. Tygar, and Kai Xia. Exploiting machine learning to subvert your spam filter. In *Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*, LEET'08, USA, 2008. USENIX Association.

Ashwinee Panda, Saeed Mahloujifar, Arjun N. Bhagoji, Supriyo Chakraborty, and Prateek Mittal. Sparsefed: Mitigating model poisoning attacks in federated learning with sparsification, 2021. URL https://arxiv.org/abs/2112.06274.

Ari Schlesinger, Kenton P O'Hara, and Alex S Taylor. Let's talk about race: Identity, chatbots, and ai. In *Proceedings of the 2018 chi conference on human factors in computing systems*, pages 1–14, 2018.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017.

Jingwei Sun, Ang Li, Louis DiValentin, Amin Hassanzadeh, Yiran Chen, and Hai Li. Fl-wbc: Enhancing robustness against model poisoning attacks in federated learning from a client perspective, 2021. URL https://arxiv.org/abs/2110.13864.

Yue Tan, Guodong Long, Lu Liu, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. Fedproto: Federated prototype learning across heterogeneous clients. In *AAAI Conference on Artificial Intelligence*, volume 1, page 3, 2022.

Vale Tolpegin, Stacey Truex, Mehmet Emre Gursoy, and Ling Liu. Data poisoning attacks against federated learning systems, 2020. URL https://arxiv.org/abs/2007.08432.

Huang Xiao, Battista Biggio, Gavin Brown, Giorgio Fumera, Claudia Eckert, and Fabio Roli. Is feature selection secure against training data poisoning? 2018. doi: 10.48550/ARXIV.1804.07933. URL https://arxiv.org/abs/1804.07933.

Guolei Yang, Neil Gong, and Ying Cai. Fake co-visitation injection attacks to recommender systems. 01 2017. doi: 10.14722/ndss.2017.23020.

Zaixi Zhang, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Fldetector: Defending federated learning against model poisoning attacks via detecting malicious clients, 2022. URL https://arxiv.org/abs/2207.09209.