Analysis on the Vulnerability of Multi-Server Federated Learning Against Model Poisoning Attacks

Lazar Nenovski[1]
L.Nenovski@student.tudelft.nl

Supervisor(s): Lydia Chen[1]**, Jiyue Huang**[1]

[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
February 1, 2024

Name of the student: Lazar Nenovski
Final project course: CSE3000 Research Project
Thesis committee: Lydia Chen, Jiyue Huang, Marco Zuñiga Zamalloa

Abstract— **Federated Learning (FL) makes it possible for a network of clients to jointly train a machine learning model, while also keeping the training data private. There are several approaches when designing a FL network and while most existing research is focused on a single-server design, new and promising variations are arising that make use of multiple servers, witch have the benefit of speeding up the training process. Unfortunately single-server FL networks are prone to model poisoning attacks by malicious participants, that aim to reduce the accuracy of the trained model. This work showcases the inherent resilience of the multi-server design against existing state-of-the-art attacks tailored around single-server FL, as well as propose two novel attacks that exploits multi-server topology in order to reduce the required knowledge an adversary needs to obtain to carry out the attack, while still remaining effective.**

Main findings are as follows: In the event that the malicious party has compromised the entire network, existing single-server attacks are sufficient to completely prevent a model from training. If they are limited to knowledge available only within the local reach of their compromised clients, the effect is minimized to where the attacks might get mitigated without any defences being necessary. However in such cases a correlation can be observed between the location of the compromised clients and the effectiveness of an attack. The novel attacks proposed in this paper exploit this relation in order to remain sufficiently effective while requiring only the same amount of data necessary for the multi-server algorithm to function.

1 Introduction

Due to the rapid growth of the number of smartphones [1], wearable devices and Internet of Things (known as IoT) sensors in the past years, more and more data is being generated outside of cloud servers and data centers, but instead at the edge on client devices. It can be very tempting if companies can somehow utilize that data in training new models without incurring additional cost for processing power and, more importantly, without overstepping any ethical or legal boundaries over utilizing that data when training their models. Therefore recently a new paradigm - Federated Learning (FL) [2]approaches when designing an FL network and while most existing research is focused on a was developed as a way to train a machine learning model on a decentralized network.

Unlike a more traditional deep learning setup where the entire process takes place in a centralized location in the cloud, FL takes a more collaborative approach by aggregating models trained on local data across a potentially large number of clients and then distributing the result back to the clients for the next iteration of training, at the end resulting in a global model trained on the available data of the participating clients. The benefits of this approach is that it makes it possible to train neural network models on data that could otherwise be sensitive to collect due to privacy or legal issues, as well as distributing the computational cost across multiple points in a network.

There are several possible approaches for the design of a FL network topology between servers and clients. The original proposed design has a single-server configuration, where all clients are directly connected with one central cloud server. However since the training process requires uploading and downloading models at every iteration the communication delay can lead to severe bottleneck for the entire process. To address this issue a novel aggregation protocol, called FedMes [3], was developed that utilizes multiple edge servers to propagate the model across the network, by using clients that are located within range of more than one such edge server as a bridge.

Because of the decentralized nature of FL, however, it is susceptible to poisoning attacks [4] if one or more clients intends to sabotage the global model. A poisoning attack is generally categorized into one of two types: untargeted, where the goal of the adversary is to bring the accuracy of the model down, or targeted, aiming to bring the accuracy down for a specific predetermined input. In order to achieve this an adversary can launch a poisoning attack either by manipulating the available data (data poisoning attack) or by uploading malicious gradients for the model it has trained (model poisoning attack). In order to mitigate the effect of an attack launched by malicious clients there have been several robust aggregation algorithms devised to combat this issue in a single-server configuration.

Research gap: While there are several papers that propose and analyze various untargeted attacks on a single server FL networks [5], [6], [7], [4], there has not been much work done for a multi-server configuration. Since both single- and multi-server have the same general goal of decentralized training of a model this can raise the question of how vulnerable a multi-server network is and if there are any additional security concerns that are inherent with such a change of topology. Therefore the focus of this paper is on untargeted model poisoning attack performed upon a multi-server FL network that uses the FedMes algorithm for aggregation.

Main contributions: An overview and analysis of the effects and required adversarial knowledge for untargeted attacks on multi server FL. We examine the difference in effect of several state of the art single-server model poisoning attacks on a multi-server network. However those attacks make quite hefty assumptions for the knowledge that the malicious parties have over the network [8], specifically that the adversary can access the computed gradients or at least their aggregation of every single client in the network. While this might be achievable in a single-server configuration by compromising the communication channel towards the single server, it is even more difficult to obtain the necessary data in a decentralized multi-server configuration, since essentially the malicious party would need to posses more knowledge

about the entire network topology than any other participants, including the edge servers. However due to the difference of how a multi-server network distributes the global model across clients this could provide local knowledge that could be exploited in a different way. For those reason this work sets out to evaluate the robustness of a multi-server network by answering the following research questions:

- **Are existing attacks, designed around single-server, effective on a multi-server network, and if so by how much?**
- **For a single-server the location of a malicious client in the network is irrelevant. Is the effectiveness of a malicious client in a multi-server network dependent on their location?**
- **Does the data sharing necessary for multi-server network to operate provides necessary information for conducting an effective attack, without the need to compromise additional parties?**

To address those issues this work presents the results of simulated FL networks with malicious clients, as well as proposing two novel attacks: "Collaborative MinMax" where an assumption is made that the malicious clients can communicate with each other and share information when conducting their attack, as well as "Isolated Knowledge MinMax" where each malicious client conducts their attack independently of one another during the attack.

The organization of the rest of the paper is as follows: Section 2 describes more in depth the concepts of federated learning in its single- and mult- server variants as well as give further elaboration on model poisoning untargeted attacks. Section 3 gives an overview of the threat model we will consider as well as the topology of the multi-server. Section 4 describes the methodology used for conducting the experiment the results of which are showcased in section 5. Section 6 argues for the ethical consideration taken for conducting this research and the conclusion is summarized in section 7.

2 Background

This section gives general description on existing relevant work for single-server and multi-server federated learning. Afterwards there is a focus on current state of the art untargeted attacks that exist for single-server FL.

2.1 Multi-Server Federated Learning

We will begin by considering a standard single-server federated learning setting [2] and will then extend it to a multi-server design. A visual comparison of both topologies is given in figure 1. In a FL network we have a coordinating server, also called parameter server (PS), and n clients that hold the data which will be used to train the model. During each global epoch of training, the PS selects a subset (which can be of size n) of available clients and sends them the parameters of the current global model. It is expected that each client will compute a stochastic gradient by using the private data they have locally available to update the received model and then upload the update back to the server. After all the clients have responded the results are then combined using some aggregation algorithm, such as averaging, into the new global model. This process is repeated either a set number of times or until convergence is achieved.

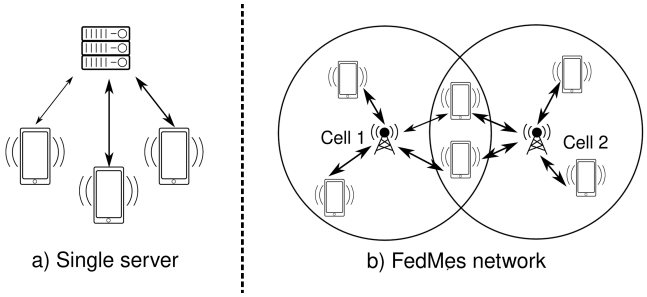


Figure 1: Comparison of single-server and FedMes multi-server FL network, where some participants might be malicious. a) Illustration of a single-server configuration: every participant communicates directly with a single server in the cloud for model upload and download. b) A FedMes network, each client communicates with every edge server within reach, in this case the clients in the middle act as bridge and average the models they receive from both servers and upload them back after their local update.

While the bottleneck in traditional deep learning training is usually in the computational resource available, in FL, due to the constant model upload and download between clients and server, the more important factor is communication latency. The FedMes algorithm [3] addresses this concern by utilizing multiple edge servers (ES), which are usually much closer to the clients, instead of relaying on communication with a single server that can be farther away in the cloud. In such a configuration each ES acts as a local PS within the physical area of its reach, referred to as cell. In order to propagate the global model throughout the entire network in this topology, the clients that are within reach of multiple ES' act as a bridge in-between them. Due to their location those clients receive the update parameters of each cell they are in. They then aggregate those models by averaging, update them with their local epoch and then upload it back to every edge servers within reach. Since the latency between ES and clients is usually much lower than that of a client and a cloud server this method greatly improves the latency for training the global model within the entire network.

2.2 Attacks on FL

There are two general types of attack an adversary would want to launch on a FL network - targeted and untargeted attack. Targeted attacks [9], also called backdoor attacks, have a goal of implanting a specific trigger during training, which causes a specific response from the model. For example in a classification task, the attacker might try to make the resulting global model have good overall accuracy, however for inputs of specific class the model always gives a specific inaccurate response. Untargeted attacks [8], also referred to as Byzantine attacks, aim to reduce the overall accuracy of the global model, which can also lead to preventing convergence from occurring.

Depending on the capabilities of the adversary, there are two main ways of performing those attacks - either through

model poisoning or data poisoning. Data poisoning occurs when the training data available on a compromised client to skew it towards the desired result. The other way is through model poisoning, where the weights of the model are adjusted either during or after training with the goal of manipulating the global model when taken into aggregation.

We will now give a brief summary of the current state of the art untargeted attacks for single server FL. Intuitively it helps to visualize the gradients of the model as a mathematical vector, in which case malicious updates are a point that attempt to steer away from the direction where the rest of the benign clients are pointing towards.

LIE attack: The LIE attack [6] works by introducing specific noise to the parameters sent to the server. The deviation is designed to be able to steer the global model, while being small enough to avoid detection. In order to achieve this it needs to obtain the mean μ, as well as the standard deviation σ of the gradients of all benign clients. Then a coefficient z is computed based on the number of malicious and benign clients in the system. The final gradients reported to the server are computed as $\mu + z\sigma$.

Fang attack: The Fang attack [5] is an optimization attack designed primarily for the Krum aggregation algorithm. The attacker needs to obtain the mean μ, and the aggregation ∇^b (which might be calculated via any aggregation strategy, such as by the Krum algorithm or any other method) of the gradients of every benign client and computes an initial direction $\nabla^p = -sign(\mu)$. The malicious update is then calculates by solving $\nabla^m = \nabla^b + \gamma\nabla^p$ for the coefficient γ. In order to speed up the process the initial value for γ is fixed and its continually halved, until ∇^m would get selected by the Krum algorithm.

MinMax and MinSum attacks: The MinMax and MinSum attacks [7] are both also an optimization type attack, but unlike in a Fang attack, the malicious gradients are updated until they are optimal, instead of stopping until the first sufficient coefficient is reached. Both variants of the attack need the aggregation ∇^b and some deviation ∇^p from the gradients of the benign clients. That can be either in the opposite, a sign flip or a standard deviation away from the inferred good direction. The malicious update is then calculated via solving $\nabla^m = \nabla^b + \gamma\nabla^p$ for γ. In the MinMax variation γ is optimized to be at most the maximum of the minimal distance between any two benign gradients while the MinSum variant optimizes to be the minimum of the sum of squared distance.

2.3 Defenses in FL

In order to mitigate the possibility of untargeted attacks several defense mechanisms for single-server FL have been devised in the form of a robust aggregation algorithm that aims to provide provably similar convergence as averaging the updates [10], [11], [12], [13], [14], while preventing any malicious input from interfering. In general they work by attempting to first filter any suspicious gradients they receive, based on some criteria for determining the outliers, and then combining the rest of the updates deemed to be good.

Currently there are no existing defenses for a multi-server FL configuration. While some of the above mentioned aggregation algorithms could be utilized when a local edge server combines the received updates, for the rest of the papers it is assumed that the network uses a simple averaging over the received updates. Even for a single-server configuration, it is arguable by [8] that a simple averaging is a sufficient defense for a realistic application of FL.

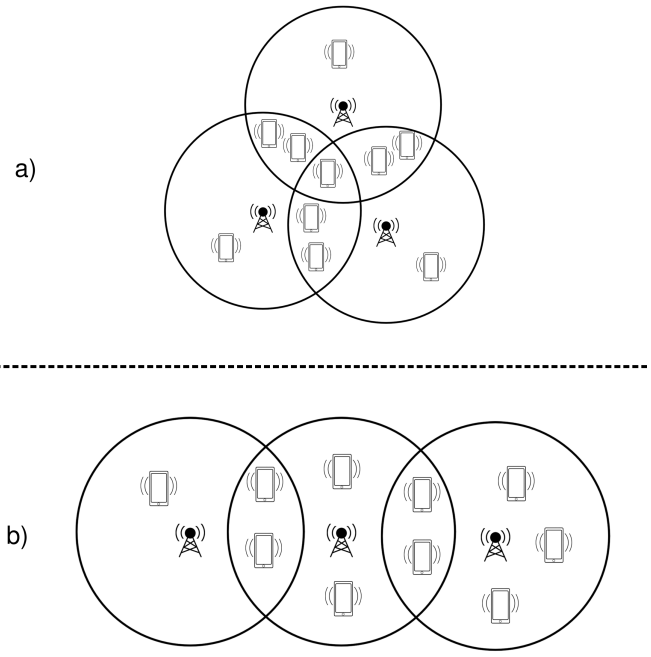


Figure 2: Visualization of the used topologies with the location of all edge servers and all clients used in the simulation: a) dense connected topology and b) sparse connected topology.

This paper focuses on investigating untargeted model poisoning attacks on a multi-server federated learning network which uses the FedMes algorithm for aggregation of the global model.

3 System and Threat Model

In this section we will present the network topology used for evaluating the resilience of the FedMes algorithm against untargeted attacks. Afterwards we will also showcase the possible threat models considered for a potential adversary.

3.1 Network Topology

There are two main topologies that would be used for the experiment, both consisting of 3 edge servers and 10 clients. The first one is the *dense* network where most clients are within reach of more than one edge server and there is a single client within reach of all three servers. Because of that central client this topology will serve as a more direct comparison with a single-server configuration. The second topology is the *sparse* network, which is less tightly connected, and therefore encapsulates more difficulties for both the global model to converge and for an attacker to have an effect. A visualization of both networks is provided in Figure 2

3.2 Threat Model

Here we consider a possible threat models for a model poisoning attack. The design of those is guided by both existing threat models towards a single-server FL network, while also considering the possible complications introduced by a multi-server network.

Adversary's Objective. The goal of the malicious party is to bring down the accuracy of the resulting global model for any input given. In order to achieve this they will construct and upload malicious gradients to the edge servers in the FedMes network. This is referred to as an untargeted model poisoning attack.

Adversary's Capabilities. We assume that the attacker is able to control m out of n clients, either by compromising existing benign clients or injecting malicious ones in the network. It is expected that less than half of the total clients are malicious ($m < n/2$), otherwise the global model would be too easily manipulated. All compromised clients are assumed to have valid data for training of the machine learning task.

For evaluation of the single-server attacks it is assumed that the adversary can access the gradients of all clients across the entire network or those of the clients within the same cell as a compromised client. For the "Isolated knowledge attack" it is assumed that the adversary can only listen in to the communication directed only at the compromised clients. It is assumed that the adversary can freely influence what parameters are broadcasted by the compromised clients.

Adversary's Knowledge. In order to properly assess what damage can be dealt by an adversary upon a Fl network, we will consider three separate levels of knowledge they posses when conducting the experiment. For evaluating the single-server attacks on a multi-server network the adversary is aware of the number and location of each client, as well as the results of their local update send towards the edge servers. This is intended to represent the worst case scenario, since the adversary has more available information then any participating party in the FL training process. A consideration of the effect of such attacks are then scaled down by assuming that the adversary is only aware of the information on every client only within the range of the edge servers the malicious clients are in. The final considered situation the adversary only has access to information on malicious clients.

4 Methodology

This section describes the environment designed for conducting the experiment for the results presented in Section 5. It goes over some of the technical details of the setup, a more in-depth explanation of the novel attack, as well as the metrics considered for evaluating the results.

In order to evaluate the effectiveness of untargeted model poisoning attacks a simulation of a FedMes environment was set up and carried out. The simulation is an FL network, either with a single-server or a multi-server topology, tasked with training a model for an image classification task, with the possible presence of malicious clients. For the evaluation the CIFAR-10 [15] 10 class image classification tasks are considered. The training data of both tasks is split evenly between each client in an iid (independent and identically distributed) fashion. The validation data of size 10000 for both dataset is reserved in accordance to the instruction of the dataset's authors. The performance of the local model available on a client is tested against the validation data and that is taken as a direct indication of the performance of the entire network. In terms of the presented topologies, visualization in Figure 2, the *dense* network is evaluated based on the client located in the range of all servers, for the *sparse* network a client in one of the overlapped regions is used.

Algorithm 1 Collaborative MinMax

```
procedure COLLAB-MINMAX
    gradients ← local_update()
    gradients ← last_recieved_updates()
    gradients ← conspiritors_update()
    mal-gradients ← minmax(gradients)
    return mal-gradients
```

Algorithm 2 Isolated Knowledge MinMax

```
procedure ISOLATED-MINMAX
    available_gradients ← local_update()
    available_gradients ← last_recieved_updates()
    mal_gradients ← minmax(available_gradients)
    return mal_gradients
```

The implementation of the novel attacks is as described in Algorithm 1 and 2. Intuitively the malicious client interprets all gradients within reach, including their own update, as those of benign client and calculates the malicious gradient as a MinMax attack on the available data. In order to better evaluate the effect of the location of the malicious clients within a topology a modified version of MinMax was used, referred in the results as "Veiled MinMax". This implementation works identically to the single-server implementation of MinMax, except for that the malicious clients can only "see" the gradients of the benign clients within the range of the same edge servers.

The design of the neural network models used for the CIFAR-10 task were VGG11 [16] and ResNet [17]. While all attacks considered in this work where originally designed to obtuse the training process they are also evaluated in their ability to degrade the accuracy of an converged network. For that a a pretrained [18] ResNet model, with an accuracy of above 95% on the CIFAR-10 task was utilized.

The code used for the simulation can be obtained online via https://github.com/Riliano/rp-msfl. The implementation of the single-server malicious updates is largely based on the code supplement to [7] with the necessary modifications to simulate a FedMes network. A comparison between the accuracy of the global models in a single-server and multi-server configurations where used as a validation of the correctness of the FedMes implementation.

5 Results

In this section the results of the simulation of the untargeted attacks are presented. First a baseline results of the networks are established and then a comparison is made between the

effects of an adversary with full and partial knowledge over all clients.

The reference values of the performance of the FL networks without any malicious clients present are showcased in Table 1. For the Cifar-10 task with a VGG11 network. The effects of each attack are compared to by running the network when no attacker is present and with no attacker present in a single-server simulation. While only the accuracy of the model is considered for the evaluation, the simulation is set up to terminate if the value of the loss function reaches a value of above 100. In those cases the effectiveness of the attack is judged based on the number of epochs before termination had to occur.

Table 1: Comparison for the effectiveness of existing single-server attacks on a multi-server network training for the Cifar-10 classification task.

Variation	Max_θ	Avg_θ	Completed epochs
Single-server no attacker	68.77	53.01	1200
Fedmes no attacker	66.15	47.12	1200
Fedmes Fang attack	14.03	10.02	1200
Fedmes Lie attack	14.01	10.07	1200
Fedmes MinMax attack	14.76	10.01	50

However we will now consider a more realistic scenario where the adversary is only aware of the gradients of clients within local reach of the malicious ones. A challenge specific for launching attacks on a multi-server network is the location of the malicious clients. While for a single-server it is irrelevant where the malicious clients are, only that they successfully upload malicious updates. That is not the case for multi-server setups, the difference of the effectiveness of an untargeted attack is showcased in Figure 3. Due to MinMax proving to be the most effective attack, a modified version referred to as "Veiled MinMax" is used. It works in the same way as the single-server MinMax with the difference that the malicious update is computed only based on the gradients within reach of the same edge servers of a compromised client. There is an apparent link between the effectiveness of a malicious client and their location.

We will now examine the effectiveness of the two novel attacks, shown in Figure 4 for the dense network and Figure 5 for the sparse one. Both attacks are more effective by a significant margin from the "Veiled MinMax", due to them being able to access a more varied set of gradients for optimizing against. Additionally they are able to exploit more effectively the advantageous location of having malicious clients in strongly overlapped regions, from attack designed around a single server configuration.

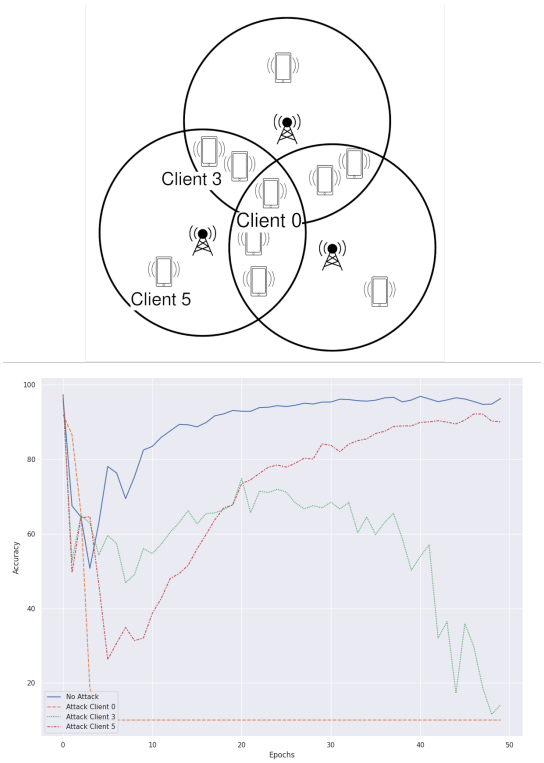


Figure 3: Effects of the veiled MinMax attack on a network from different malicious clients on a ResNet network for the Cifar-10 task. The malicious client in an overlapped region is able to be very effective, while those within a reach of a single server can barley affect the accuracy.

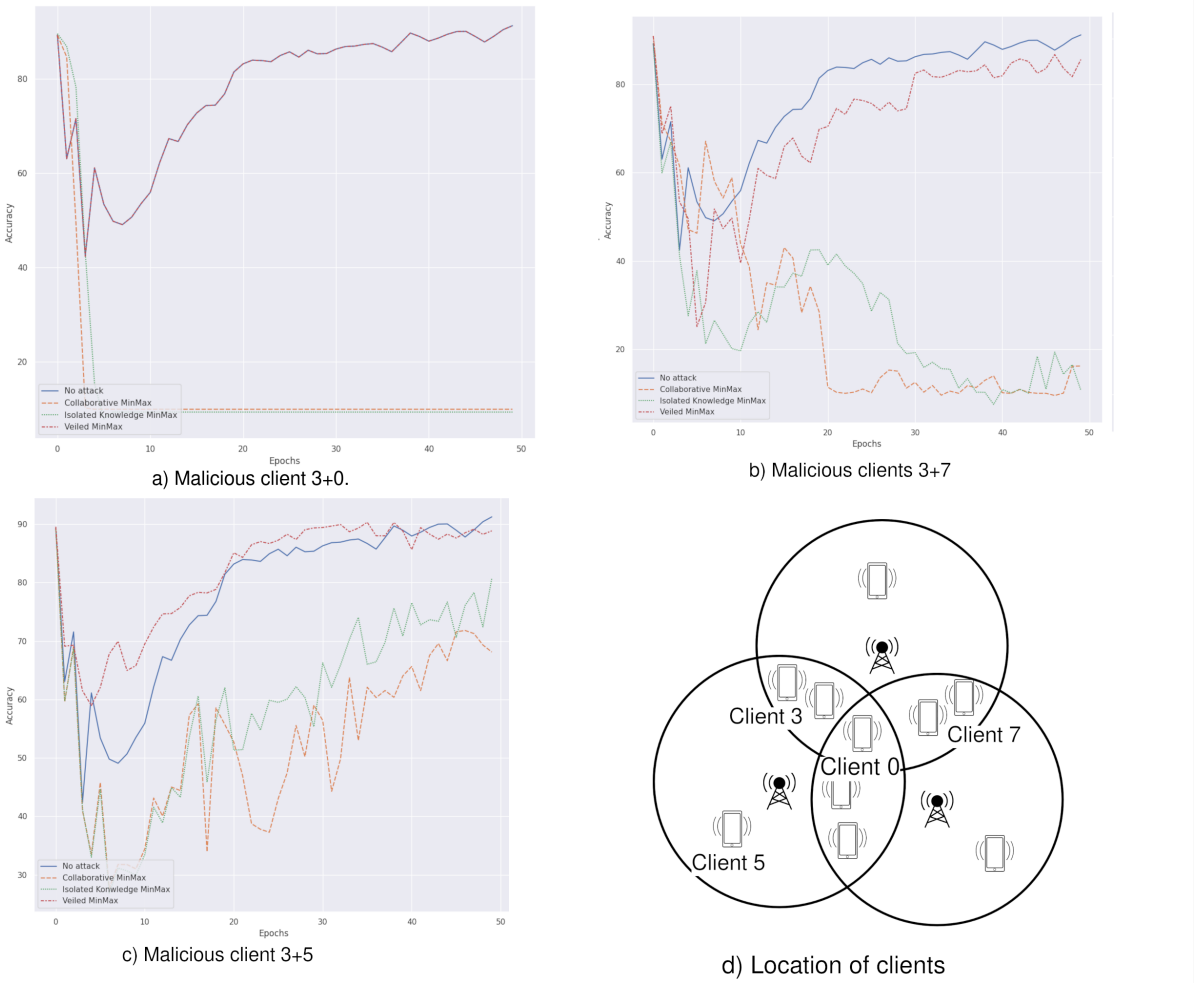
6 Responsible Research

This section is dedicated to ethical considerations taken into account when conducting this research. The main guiding principles used for consideration where based on the Netherlands Code of Conduct for Research Integrity [19]. In order to ensure the validity and reproducibility of the findings discussed in this paper, the full source code for running the simulation, as well as the raw data of the presented results are publicly available online. The code itself is provided together with the necessary instructions to be able to run on most standard computational environment used for data science.

A major ethical motivator when conducting this research was in advancing the technique of federated learning easier to make it more suitable for larger adaptation. Even though this paper provides instructions on how one could launch an attack on such a network, it does so more with the intent to investigate what is the inherent resilience of the design and what information an attacker might require, so that those aspects can be taken into account when a real world multi-server FL network is being deployed.

7 Conclusions

The main goal of this work was to examine the effectiveness of existing untargeted model poisoning attack designed for



lolk

Figure 4: Comparison on the effectiveness of ”Isolated Knowledge MinMax” and ”Collaborative MinMax” against ”Veiled MinMax” in the dense network. The attack is carried out on a pretrained ResNet model for the Cifar-10 task for 50 epochs with a learning rate of 0.00082. The effectiveness of all attacks is primarily determined by the location of the malicious clients, by having a malicious client within reach of many edge servers an attacker can significantly cripple the performance of the network.

single-server FL networks when launched against a multi-server network. If a malicious party is able to also compromise the communication channel of every client across the network, then existing single-server attacks are sufficient to bring down the accuracy of the model trained by that network. If the malicious party is only able to compromise clients within reach of their local server, the simple averaging is sufficient to deter such attacks. In the case that the adversary is aware that the compromised clients are part of a multi-server FL network, they can exploit the data passing through such clients in order to launch attacks with a more severe effectiveness.

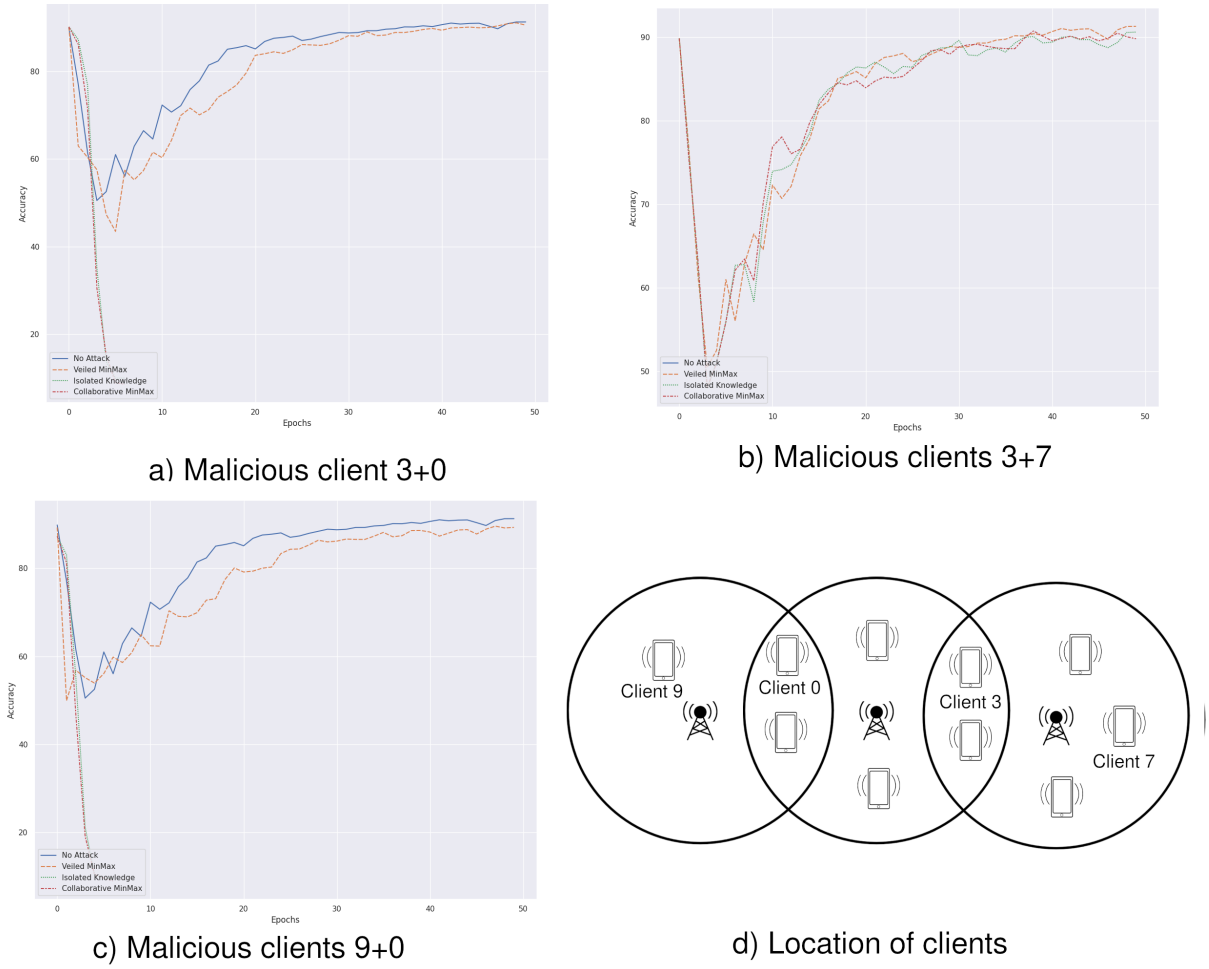


Figure 5: Comparison on the effectiveness of ”Isolated Knowledge MinMax” and ”Collaborative MinMax” against ”Veiled MinMax” in the sparse network. The attack is carried out on a pretrained ResNet model for the Cifar-10 task for 50 epochs with a learning rate of 0.00082. The effectiveness of all attacks are in general less effective, due to less available overlap between the edge servers.

References

[1] Monica Anderson. Technology device ownership: 2015. 2015.

[2] 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.

[3] Dong-Jun Han, Minseok Choi, Jungwuk Park, and Jaekyun Moon. Fedmes: Speeding up federated learning with multiple edge servers. *IEEE Journal on Selected Areas in Communications*, 39(12):3870–3885, 2021.

[4] Lingjuan Lyu, Han Yu, and Qiang Yang. Threats to federated learning: A survey. *arXiv preprint arXiv:2003.02133*, 2020.

[5] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. Local model poisoning attacks to Byzantine-Robust federated learning. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 1605–1622. USENIX Association, August 2020.

[6] Gilad Baruch, Moran Baruch, and Yoav Goldberg. A little is enough: Circumventing defenses for distributed learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[7] Virat Shejwalkar and Amir Houmansadr. Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning. In *NDSS*, 2021.

[8] Virat Shejwalkar, Amir Houmansadr, Peter Kairouz, and Daniel Ramage. Back to the drawing board: A critical evaluation of poisoning attacks on production federated learning. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1354–1371, 2022.

[9] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 2938–2948. PMLR, 26–28 Aug 2020.

[10] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[11] El Mahdi El Mhamdi, Rachid Guerraoui, and Sébastien Rouault. The hidden vulnerability of distributed learning in byzantium, 2018.

[12] Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta. Generalized byzantine-tolerant sgd, 2018.

[13] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. In *International Conference on Machine Learning*, pages 5650–5659. PMLR, 2018.

[14] Luis Muñoz-González, Kenneth T. Co, and Emil C. Lupu. Byzantine-robust federated machine learning through adaptive model averaging, 2019.

[15] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[16] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[18] Yaofo Chen. Pytorch cifar models. https://github.com/chenyaofo/pytorch-cifar-models, 2023.

[19] Universiteiten van Nederland. Netherlands code of conduct for research integrity, 2018.