Robustness Against Untargeted Attacks of Multi-Server Federated Learning for Image Classification

Are Defenses Based on Existing Methods Effective?

Todor Mladenovic[1]

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
January 28, 2024

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

Abstract

Multi-Server Federated Learning (MSFL) is a decentralised way to train a global model, taking a significant step toward enhanced privacy preservation while minimizing communication costs through the use of edge servers with overlapping reaches. In this context, the FedMes algorithm facilitates the aggregation of gradients, contributing to the convergence of the global model. Attacks that aim to reduce the accuracy of the global model are called untargeted attacks. One such attack that is particularly difficult to detect is the Min-Max attack. This paper explores the extension of existing defenses to enhance the robustness of MSFL against the Min-Max attack.

To do this, existing state-of-the-art defenses, including Median, Krum, Multi-Krum, Trimmed-Mean, Bulyan and DnC are extended and examined for their adaptability to this context. We refer to the extended versions of these defenses as FMes-Defenses.

Our results indicate that FMes-Defenses are ineffective in preventing the Min-Max attack from diminishing the accuracy of the global model. Surprisingly, we find even FMes-DnC is inadequate despite it's Single-Server counterpart (DnC) being renowned for mitigating the Min-Max attack.

These findings emphasise the need for novel defenses specifically tailored to the nuances of MSFL. While representing a significant stride in communication efficiency, MSFL, complemented by the FedMes algorithm, may require additional measures to ensure robust security against sophisticated untargeted attacks. This research contributes valuable insights into the challenges and importance of enhancing the security of MSFL in its ongoing development.

1 Introduction

With rising awareness and concerns about data privacy, in 2016 Google introduced Federated learning (FL) [1]. A new decentralised approach to machine learning where in contrast to distributed learning, the data used in training never leaves the device it comes from. Instead, devices receive a global model from a server, train it locally with their respective data, then send the updates back to the server which aggregates all the updates it received to get a new global model.

Multi-Server Federated Learning (MSFL) is proposed for scenarios where there are multiple edge servers, each can reach a subset of the participants and the server-reaches overlap in certain regions. In MSFL using the FedMes aggregation, devices in regions where server reaches overlap (devices reached by more than one server) are given additional weight to their updates during aggregation [2]. This allows for reduced communication between the servers and in many cases results in MSFL with FedMes outperforming alternatives like Hierarchical and Single-Server FL.

Various types of attacks on FL exist and they differ by purpose. Though FL was devised to keep data private, inference attacks can be employed in attempts to derive client devices' data by monitoring the gradient updates [3]. Free-rider attacks, where users pretend to be contributing to training can be used to gain access to commercially high valued global models [4]. Poisoning attacks aim to reduce the robustness of the global models [3]. Targeted poisoning attacks (backdoor attacks) aim to manipulate the global model into generating a specific erroneous output upon receiving a specific input, while not affecting the models performance on other inputs. Untargeted poisoning attacks on the other hand attempt to generally decrease the accuracy of the global model.

All the different types of attacks pose threats to the sanctity of FL but untargeted attacks can corrupt a relatively large population of clients and be very hard to detect [5]. Making them a serious threat to production FL. Additionally, if the FedMes aggregation gives added weight to a malicious update from an untargeted attack, this update in turn has a greater impact on the global accuracy. Making the attack more potent. This increase in potency is not so clear with the other types of attacks.

In Single-Server FL, an untargeted attack known as Min-max, is very effective against the common state-of-art defenses [6]. However a state-of-art defense called DnC(divide-and-conquer) is presented as the solution that effectively mitigates the effects of the Min-Max attack[6] when data is iid. What happens when an attack such as this is launched from the most overlapping regions in MSFL with the FedMes aggregation, such as in Figure 1? The devices in these regions are given more weight to their updates based only on their geographical location. Can the DnC defense be adapted to effectively work in this context?

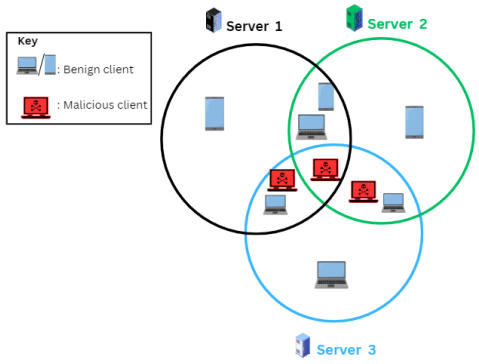


Figure 1: Visualisation of attackers in most overlapping areas of Multi-Server Federated Learning.

Research question: *"How effectively can state-of-the-art defenses, originally designed for Single-Server Federated Learning, be extended to Multi-Server Federated Learning with FedMes, to mitigate the Min-Max attack's impact on the accuracy of an image classifier's global model?* To answer this question a quantitative approach will be taken. The common state-of-art defenses (Median, Krum, Multi-Krum,

Trimmed-Mean, Bulyan) as-well as DnC are extended to the new setting of MSFL with FedMes. We refer to the extended versions of these defenses with the prefix 'FMes-' (e.g. the extended version of DnC is FMes-DnC). A series of experiments are run on these FMes-Defenses (the term we use to refer to all the defenses we extend to this setting) in order to collect data to answer the following **sub-questions**:

1. Is MSFL with FedMes vulnerable to the Min-Max attack when no defense is present?
2. How effective are FMes-Defenses based on common state-of-art defenses at preventing the Min-Max attack from reducing the global model accuracy?
3. To what extent does the FMes-DnC defense succeed in preventing the Min-Max from reducing the global model accuracy?

Contribution: Existing research on untargeted attacks and their respective defenses have mostly been centered around Single-Server and Hierarchical FL [7] [8]. MSFL algorithms such as FedMes are promising new concepts that have potential to be relevant to the future of machine learning. This paper for the first time applies existing defenses for untargeted attacks to MSFL, in order to investigate the difficulties this novel setting presents for security. Ensuring MSFL can be made robust to attacks is crucial if it's ongoing development is to progress.

The structure of the following sections is as follows. Section 2 will go into further detail of relevant concepts and related work. In section 3, specifics of the system model can be found. Following this section 4 will describe the methodology used in creating the FMes-Defenses. Section 5 continues on with the specifics of the implementation used for experiments. Then in section 6 is where the analyses of the experiment results can be found. In section 7 the ethical aspects and steps taken to ensure reproducibility of the research will be discussed. Followed by the limitations of the research in section 8. Finally, section 9 contains the conclusion accompanied by suggestions for future work.

2 Related Work

The Related Works section introduces foundational concepts for this paper. Subsection 2.1 explores MSFL and FedMes. Subsection 2.2 describes common state-of-the-art defenses examined in a new context. Subsection 2.3 discusses two common state-of-the-art attacks and compares their effectiveness with Min-Max, detailed in Subsection 2.4. Subsection 2.5 presents the theory behind the DnC defense.

2.1 Multi-Server Federated Learning and FedMes

Fundamentally, FedMes is an aggregation strategy proposed for MSFL, in which the regions of server reaches overlap [2]. To understand what this entails, it's helpful to highlight why it was created and how it differs from alternatives such as Single-Server and Hierarchical FL. Then the exact algorithm and it's advantages in intended contexts can be appreciated.

Single-Server FL is the simplest from of FL. In it, all updates from client devices ultimately need to communicate with a single server [2]. This server aggregates the global model using the updates from all clients. Due to the constant communication (downloading/updating of models) between clients and this server, it can cause communication bottlenecks. Furthermore, if we consider this central server to be a cloud server (being in the cloud would allow it reach many client devices), the distance between clients and the server causes propagation delay during communication [9].

Edge computing offers a solution to tackle the latency of cloud computing architectures [10]. Edge servers can be used to replace the relatively slow single-cloud-server architecture. Due to the relatively small number of clients a single edge server can reach, multiple edge servers would need to be utilised [2].

Hierarchical FL is a popular way to train a global model through these edge servers [11]. In each epoch clients download a model from their edge server, train it with their data, then send the updated model back to their edge server. Periodically the edge servers will send their global models upstream, where global models of multiple servers will be aggregated and sent back downstream to the edge servers and further to the clients those edge servers reach.

MSFL is a novel alternative which reduces communication costs further [12]. It avoids communication between edge servers completely up until the end of training. By having the edge servers' regions overlap, devices in these regions act as a sort of bridge of communication. Edge servers' global models to impact each other without having to send their models further upstream periodically.

FedMes is an aggregation strategy that's proposed for this setting [2]. When an edge server is aggregating all of it's received updates into an average, a weight is given to each update. This weight is proportional to the size of the dataset that trained the update, as-well as the number of servers that reach the client it comes from. A benefit of this aggregation is not only the reduced communication costs, but also the speed of convergence. Particularly with non-iid data and topologies with densely populated regions with many clients in overlapping regions.

2.2 Common State-of-Art Defenses

In the following chapters, 'common state-of-art defenses' will refer to those described in this subsection. That is not to say that other researched defenses do not exist but they are beyond the scope of this paper as they are not as common in research papers as the ones mentioned. Some of the techniques not covered include those that employ clustering, coordinate-wise median, geometric median and norm-bounding [3].

In addition the Median aggregation, the relevant common state-of-art defenses include:

- *Krum* - Uses Euclidean distance between the updates sent in by clients to identify the gradient-update least likely to be malicious [13]. Where n is number of updates and m the upper bound of malicious clients, it selects the gradient closest to it's $n - m - 2$ neighbours.
- *Multi-Krum* - A variant of Krum, designed to increase speed of convergence [13] by incorporating aggregation of multiple updates. The implementation used iteratively selects updates to aggregate and removes them

from a list called 'remaining updates'. The stopping condition is when the number of 'remaining updates' is equal to $2 \cdot m + 2$, where m is the upper bound of malicious clients.

- *Trimmed Mean* - A coordinate-wise aggregation, it excludes specific weights of updates as opposed to entire updates themselves [14]. The largest and smallest weights along each dimension are excluded. The updates are refactored such that they are sorted along each dimension (e.g. update in first position contains the smallest weights of each dimension which could have all correlated to updates sent in by differing clients before refactoring. The implementation used then excludes updates in the first $m/2$ and last $m/2$ positions, where m is the upper bound of malicious clients.
- *Bulyan* - A combination of Multi-Krum and a coordinate-wise aggregation similar to Trimmed Mean, it was designed to catch even those malicious updates close to benign updates in the square Euclidean norm space [5]. After performing Multi-Krum to exclude distant updates, the coordinate-wise median is found. Then a coordinate-wise sort is performed based on distance to the median. After which the weights at the end of this sorted list of updates are excluded such as in Trimmed Mean.

2.3 Less Effective State-of-Art Untargeted Attacks

These state-of-art attacks will not be evaluated in this paper due to superior performance of the min-max attack [6]. However, since the comparison to these attacks is made, the specific attacks being referred to are the following:

- *LIE(Little is enough)* - An untargeted attack that finds malicious gradients close to the average of benign clients by adding small perturbations to this average [15]. It assumes knowledge of benign clients' updates but does not make assumptions about the aggregation strategy in use. In the single server the min-max was found to significantly outperform LIE, despite both assuming the same threat model [6].
- *Fang* - This untargeted attack is one that finds malicious gradients targeting a specific aggregation strategy, with knowledge of the benign client's updates [16]. In spite of this added knowledge of aggregation strategy, it is shown to be quite ineffective for iid and very imbalanced non-iid datasets [6].

2.4 Min-Max Attack

An untargeted attack created with the consideration that "FL platforms can conceal the details and/or parameters of their robust AGRs to protect the security of the proprietary global models" [6]. Even without knowledge of the specific aggregation (such as those in subsection 2.2), it is capable of significantly reducing the global accuracy of a FL model (except for non-iid data with Multi-Krum). Making it more realistic and harder to defend then commonly researched untargeted attacks such as fang.

However, the attack presupposes that in an epoch, malicious clients possess knowledge of all the updates benign clients transmit to the servers before the malicious clients send their update. In reality, there are a number of practices that could be deployed to prevent this. One of which is Secure Multi-Party Computation [17]. A form of encryption that could be used to protect the privacy of the benign updates.

The attack finds a malicious gradients that is in closer proximity to every benign gradient than the maximum distance of any two benign gradients. Every malicious client has the same gradient update in an epoch.

2.5 DnC Defense

The DnC defense is proposed as a defense which addresses the issue of common state-of-art defenses not being effective against attacks like the Min-Max attack (at least for iid data) [6].

It employs SVD (Singular Value Decomposition) to detect outliers [18]. This is done by assigning outlier scores based on how much a gradient deviates along the dimension of maximum variance (found by projecting updates on the 1st Principal Component). However, due to the large memory requirement, dimensionality reduction must first be performed. Only then is SVD performed on the reduced set of dimensions.

3 System Model

The System Model section contextualizes the topics discussed in Section 2 and outlines the study's assumptions. Subsection 3.1 details the components and interactions within the MSFL system. Subsection 3.2 presents the implementation of the FedMes aggregation used in the experiments. Subsection 3.3 addresses assumptions regarding malicious clients.

3.1 Multi-Server Federated Learning Components and Interactions

In our MSFL model there are really only 2 types of components. The servers and the clients.

The clients refer to the devices that hold the data. Only on these devices is the image classifier being trained. They only communicate with the servers that can reach them. From which they download a model, that they use as a starting point for their training. If multiple servers can reach them their starting model before training is the average of the models those servers send them. After they are done with training, they send their model back to the server(s).

The servers refers to edge servers. They can reach a subset of the clients. They first send an initial model to the clients they reach. After training is complete, they receive updated models from those clients. Using the aggregations described in subsection 3.3, they aggregate the multiple update models into an average. Trying to do so in a Byzantine-Safe manner.

3.2 FedMes Implementation

The implementation, inspired by the original pseudocode provided for FedMes was designed to work with cases when there is coordinate-wise aggregation [2]. Algorithm 1 is pseudocode for the implementation of the FedMes aggregation used in the study. *selected_updates* refers to the list of

all updates used in aggregation (a tensor of tensors). *selected_indices* is a list of lists, in which every element in inner lists represent the client id that the element in the same position in *selected_updates* comes from. The last parameter *overlap_weight_index* is a list where the client id is the index for the weight associated with how overlapping of a region the client comes from. In it we associate a weight 1 wit a non-overlapping region, 2 if it has 2 overlaps and 3 if it has 3 overlaps. Note: since data-set sizes of clients is equal in this study, the implementation omits the consideration it.

Algorithm 1: fedmes_elementwise

Input: selected_indices, selected_updates, overlap_weight_index;
Output: a fedmes aggregate;

$total\_selected\_weight \leftarrow$ list of zeros with dimensions of selected_indices;

$fedmes\_mean\_selected \leftarrow$ list of zeros with dimensions of selected_indices;

foreach i ***in*** *range(len(selected_indices))* **do**
    client_weights $\leftarrow$ overlap_weight_tensor[selected_indices[i]];
    fedmes_mean_selected $+= selected\_updates[i] \times client\_weights$;
    total_selected_weight $+= client\_weights$;

Return $fedmes\_mean\_selected \div total\_selected\_weight$;

3.3 Threat Model

Malicious Client's Goal:
Find gradients that when aggregated into the global model, will reduce it's accuracy.

Malicious Client's Knowledge:

- Does not know the defense mechanism used in the aggregation process (since min-max attack is AGR-agnostic).
- Knows the updates sent in current epoch by benign clients that share an edge server with them.
- Knows how the reaches of servers overlap, so that they know where the most overlapping areas are.

Malicious Client's Capabilities:

- Is selected in each epoch.
- Cannot control more than 10% of selected clients in an epoch. A constraint decided upon due the critic of modern research into FL attacks, assuming unrealistic capabilities of malicious participants when it comes to control over benign clients [5].

4 Methodology

The Methodology section outlines the approach used to extend state-of-the-art defenses to MSFL with FedMes. Subsection 4.1 gives a general overview of the process for developing FMes-Defenses. Subsection 4.2 provides detailed information on FMes-DnC.

4.1 The FMes-Defense Aggregations

This study introduces several new aggregation strategies. More accurately it introduces a framework to extend existing defenses to FedMes. Current state-of-art defenses have not been implemented to run in a MSFL setting, let alone with FedMes as the underlying aggregation methodology. To evaluate the effectiveness of techniques introduced by the common state-of-art defenses (sub-section 2.2) and DnC in this unique context, required the introduction of distinctive changes to these aggregations. Once the framework is applied, since the fundamentals of the aggregation are changed, they can be considered as FMes-Defense aggregations.

Accounting for Multi-Server: As opposed to the Single-Server FL setting, when working with MSFL, multiple edge servers perform aggregations on their clients independently. Each FMes-Defense is therefore called by each edge server and aggregates only the updates of clients that server reaches. Since an Edge server reaches less clients the question of if the upper bound of malicious client for the specific edge server (a hyper-parameter of the defenses) needs to be lower than when considering an aggregation for all clients? The locations of attackers cannot be assumed however, and all attackers could be attacking the same server. Therefore, we determine the upper bound must effectively remain unchanged.

Incorporating FedMes: The general approach used in the study to make a defense FMes is illustrated by algorithm 2. In it *all_updates* is the list of all updates from clients the server reaches. *n_attackers* is the aforementioned m_{server} and *overlap_weight_index* is the same as from algorithm 1.

Algorithm 2: General FMes-defenses

Input: all_updates, n_attackers, overlap_weight_index;
Output: a more robust fedmes aggregate;

$selected\_updates \leftarrow$ empty list;

$selected\_indices \leftarrow$ empty list;

Use techniques from state-of-art defense of choice to add updates from $all\_updates$ that pass to $selected\_updates$, while keeping track of the client each parameter (of each update) comes from in $selected\_indices$

Return fedmes_elementwise(selected_indices, selected_updates, overlap_weight_index);

The reasoning behind this algorithm is to try mitigate the effects of the added weight due to overlapping regions. That is why we first attempt to prune the malicious updates with the outlier detection technique before adjusting for fedMes. Though in some cases this is not possible.

For *FMes-Median* and *FMes-Krum* the generic framework from algorithm 2 doesn't apply since they pick a single update to use as the aggregate. Instead the update of clients are re-added to the updates list based on the client's region. After this the Median or Krum of the updates list is taken. This is done to incorporate the bias of FedMes for overlapping regions. Though it is important to note that FedMes is clearly described to use an average more inline with a mean.

4.2 FMes-DnC Defense

The FMes-DnC Defense uses the outlier detection techniques introduced by the DnC Defense (subsection 2.5). As we can see in algorithm 3, this algorithm takes additional parameters compared to the normal FMes-defenses. n_iters determines how many iterations of dimensionality reduction and SVD outlier detection are performed. $filter\_frac$ is used to control the number of updates that are pruned in a single iteration. Finally $subsample\_size$ is used to determine the dimensions that the updates should be reduced to before performing SVD.

The values for n_iters, $filter\_frac$ and $subsample\_size$ used in the implementation are 2, 2 and 10000. These values were determined empirically by considering. By observing which provided the best accuracy, while considering the runtime of the algorithm which scales quite poorly with n_iters and $subsample\_size$.

Algorithm 3: FMes-DnC

Input: all_updates, n_attackers, overlap_weight_index n_iters, filter_frac, subsample_size;
Output: a robust fedmes aggregate;

selected_indices_sets ← empty list;
gradient_dimension ← size of the second dimension of all_updates;
for _ ***in*** *range(niters)* **do**
    random_dimensions ← sort (randomly selected a total of subsample_size dimensions);
    subsampled_gradients ← all_updates[:, random_dimensions];
    mean_gradients ← mean gradient of subsampled_gradients;
    centered_gradients ← subsampled_gradients - mean_gradients;
    _, _, right_singular_vector ← SVD(centered_gradients);
    top_right_singular_vector ← first column of right_singular_vector;
    outlier_scores ← squared projections of centered_gradients along top_right_singular_vector;
    selected_indices_sets.append(indices of lowest outlier scores);
selected_indices ← intersection of selected_indices_sets;
selected_updates ← subset of all_updates using selected_indices;
Return fedmes_elementwise(selected_indices, selected_updates, overlap_weight_index);

5 Experimental Setup

The Experimental Setup section details the specifications utilized in the experiment implementation. Subsection 5.1 outlines the datasets and their usage. Subsection 5.2 provides information on the learning models and their parameters. Subsection 5.3 describes the network topology and attack cases evaluated in the study. Subsection 5.4 summarizes how experimental results are gathered.

5.1 Data-sets and Distribution

CIFAR10 is an iid data-set with 60,000 RGB $32x32$ images of the 10 digits[6]. 50000 samples are used for training the global model. 20 clients are assigned 2500 samples each. 5000 of the remaining samples are used for testing and the last 5000 used for validation.

Fashion-MNIST is an iid dataset consisting of 70,000 28x28 grey-scale images of the 10 digits [19]. 60,000 samples are used for training and the remaining samples are split into two randomly sampled equal-sized sets. One to be used for validation and the other for testing. Again the training data is spread equally among the client devices, which each get 3000 samples.

5.2 Learning Model and Hyper-parameters

AlexNet and VGG11 will be used as the CNN models for both data-sets. AlexNet is a large CNN with 8 layers [20]. VGG11 is the 11 layered variant of the VGG CNN[21] [22].

For all data-set-learning-model combinations, the maximum epochs is set to 1500. There are early stopping conditions which can result in training stopping before epoch 1500. The first is if the loss in the model rises above 10. The second is if neither best loss or best accuracy improve for 250 epochs. Additionally an SGD optimizer is consistently used with a decay factor of 0.5.

For Cifar10 with AlexNet batch sizes of 250 samples are used. Additionally, the learning rate starts of at 0.8. The decay factor is applied epoch 1000 and 1200.

With Cifar 10 and VGG11, the batch size is 165 samples. The learning rate is 0.5 at epoch 0. Then the decay will be applied at epoch 800, 900, 980 and 1000.

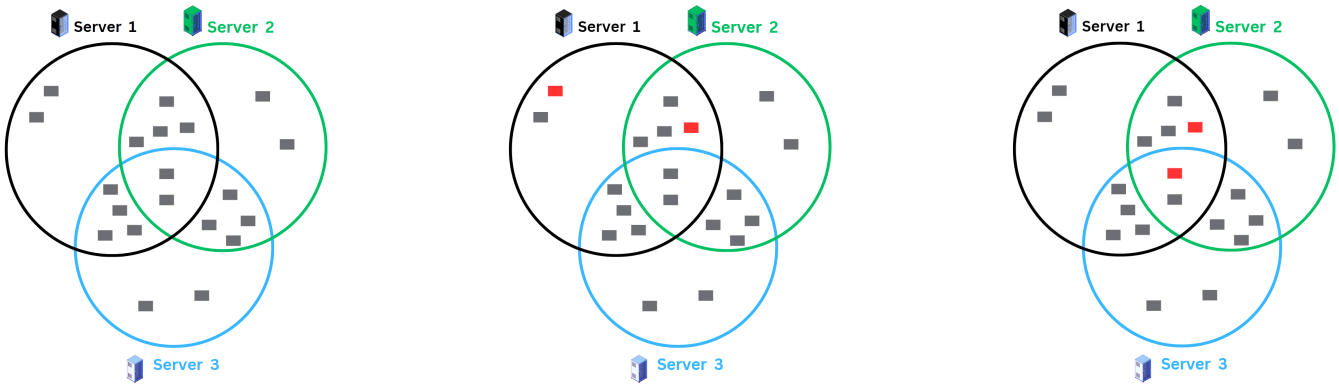
For Fashion-MNIST and AlexNet the batch size is 200. Learning rate starts at 0.5 and the decay factor is applied to it at epoch 600 and 800. When VGG11 is used, the only thing that's different is the batch size is 150.

5.3 Network Topology, Client Selection & Attack cases

In figure 2a, we can see the layout of the topology. Its a symmetric topology which consists of 3 servers that in each epoch will select a total of 20 clients. Each server selects 2 client that's only in their region, 8 clients that are in their and another server's regions and 2 clients that all the servers reach. The reason for having many of the selected devices be in overlapping regions is as mentioned in the background section, FedMes works particularly well when there are many clients within the overlapping regions. An assumption worth mentioning about the implementation of this topology, if a client in an overlapping region is selected in an epoch, it is selected by all servers that reach it.

The server topology will be modified to accommodate two attack cases which will be used to examine the effect of having malicious clients launching attacks from more heavily overlapping regions. The First attack case can be seen in figure 2b. It consists of an attacker that's in a region where two servers overlap and an attacker in a region where there's no overlaps.

The second attack case (figure 2c) is where there's again one attacker in a region where two servers overlap, but now



(a) Network Topology (b) Attack case 1 (c) Attack case 2

Figure 2: Visualisations of relevant Network Topology with grey squares as benign clients and red squares as malicious clients.

the other attacker is the one device in the region where all the server's regions overlap.

5.4 Gathering Experimental Results

First **Experiment 1** will be conducted with no defense and the two attack cases from subsection 5.3, as-well as the case with no attackers. Each case will be run 3 times, retraining the model ab initio each time.

Experiment 2 is run 3 times for each of the relevant aggregation strategies (FMes-Median, FMes-Trimmed-Mean, FMes-Krum, FMes-Multi-Krum, FMes-Bulyan and FMes-DnC) and dataset-model combinations (cifar10-Alexnet, FashionMNIST-VGG, etc). In each run, the model is trained from the beginning. In a single run this is done for the stronger attack case (determined by Experiment 1).

During a run the validation accuracy and loss of the global model are measured at each epoch. Additionally, the test accuracy of the gradient with the best validation accuracy during training is recorded. These metrics are all considered in the next Section.

6 Evaluation of Results

In the Evaluation of Results section the experimental results are presented and used to motivate answers to the sub-questions mentioned in section 1. Subsection 6.1 addresses sub-question 1. Subsection 6.2 addresses sub-question 2 and subsection 6.3 addresses sub-question 3. Subsection 6.4 further discusses and reflects on the results.

6.1 Is Multi-Server Federated Learning with FedMes Vulnerable to Min-Max?

To evaluate if MSFL with FedMes is vulnerable to the Min-Max attack, we conduct Experiment 1 (subsection 5.4). The results from this experiment can be seen in table 1.Comparing the no attack accuracy to both attack cases' accuracy for each scenario reveals the attack cases are very capable at reducing the accuracy of the MSFL global model's when no defense is present. Proving **FedMes is vulnerable to the Min-Max attack**

Table 1 illustrates that attack case 2 tends to impact accuracy more compared to attack case 1. Considering the attacker topology (Subsection 5.3) and the weighted updates from overlapping regions in FedMes, the difference in performance between the two cases was surprisingly modest, except for the Fashion-MNIST and AlexNet scenario (where the gap in achieved test accuracy is 12.0).

Despite attack case 2 generally appearing more potent, Fashion-MNIST-VGG is interesting since it's the only scenario where the best achieved test accuracy indicates that is not the case. However, looking at the performance against the validation set in figure 3 paints a different picture. We see attack case 2 reaches a lower peak validation accuracy of 14.0 while attack case 1 reaches 15.1. Additionally, attack-case 2 causes loss in the model to get high enough to trigger an early stopping condition by epoch 92, while attack case 1's loss reaches this threshold at epoch 157.

Figure 3: Validation Accuracy against Epochs for most effective runs of Attack case 1 and Attack case 2 against FedMes with no defense (Fashion-MNIST with VGG case)

.

With all this in mind we can conclusively determine that

Table 1: Table of Mean Test Accuracy and Standard Deviation from 3 runs of each attack case from section 5.3 and the case with no attackers. No Defense is present in these runs. Results from all runs can be found in Appendix A.

	Cifar10				Fashion-MNIST			
	AlexNet		VGG		AlexNet		VGG	
Aggregation Strategy	**Accuracy**	σ	**Accuracy**	σ	**Accuracy**	σ	**Accuracy**	σ
No-Attack	50.9	0.34	52.6	0.4	82.6	1.05	81.9	0.16
Attack case 1	20.6	0.31	15.2	1.06	35.5	3.9	14.8	0.7
Attack case 2	19.5	0.34	15.1	1.21	23.5	3.47	16.0	0.93

our MSFL model with FedMes is highly vulnerable to Min-Max. Furthermore, we conclude that attack case 2 is the more potent attack and continue with it in our next Experiment.

6.2 How effective are the Common Defense techniques?

The results used to evaluate performance of the common state-of-art defenses against the Min-Max attack, are gathered by Experiment 2 (subsection 5.4) with attack case 2. They can be seen in table 2. Ultimately the results show that the **FMes-Defenses based on the common state-of-art defenses are ineffective against the Min-Max attack.**

Whats interesting to observe about the results is the cases in which the defenses on average perform even worse against the Min-Max then when no defense is present. In table 2 these are marked in red. And we can see that Multi-Krum and Bulyan are outperformed by no defense in 3 out of the 4 examined data-set-learning-model combinations. Suggesting perhaps that the Multi-Krum approach might fundamentally not be suitable for MSFL with FedMes (Bulyan's first step is Multi-Krum, see subsection 2.2).

All of the common based FMes-Defenses are not robust enough. Based on the test accuracy in table 2, the best of them are FMes-Krum and FMes-Trimmed-Mean, as they generally achieve the best accuracy. However, even in the best scenario of FMes-Trimmed-Mean for Fahion-MNIST with VGG, the average 40.0 accuracy achieved is not even 50% of the accuracy when no attack is present(found in table 1). The standard deviation for it is also very large at 8.49, showing instability. When we compare this to what is considered robust in comparable literature it is far from it [5].

6.3 Is FMes-DnC Defense Robust?

To understand the effectiveness of FMes-DnC against the Min-Max attack, table 2 can be inspected. It's apparent by observing the test accuracy achieved by FMes-DnC that the **FMes-DnC Defense is not robust against the Min-Max attack.**

In table 2 it can be seen that it's outperformed by FedMes with no defense for Cifar10 with Alexnet. This is a drastic contrast to this data-set-learning-model combination with DnC in Single Server FL with iid data. There DnC 'effectively filters malicious gradients' and mitigates attack impact [6].

Even the best run of the best data-set-learning-model combination (Fashion-MNIST with VGG) for FMes-DnC is lackluster. We can see this by observing figure 4. The global model's validation accuracy steadily increases for 200 epochs before experiencing a strong crash. It however shortly recovers and for the next 200 epochs seems to be learning well. Ultimately, learning stops prematurely due to loss in model getting too high, with Test accuracy peaking at 53.3. This accuracy is considerably higher than that of the other FMes-Defenses but it's rendered less impressive by the instability of the defense as concluded by a large standard deviation of 10.5 for Fashion-MNIST with VGG.

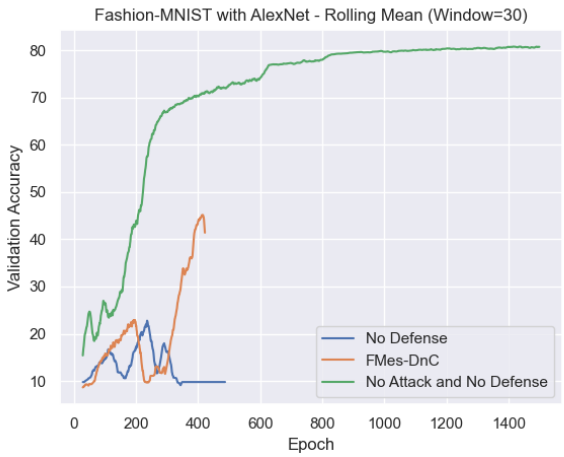


Figure 4: Validation Accuracy against Epochs for most effective runs of FMes-DnC and no-defense-no-attack, and the most potent run of attack case 2 with no defense. This graph uses a rolling mean to better visualize the training.

6.4 Discussion

FWhile we expected MSFL with FedMes to be vulnerable to the Min-Max attack, the magnitude of the threat to security surpassed predictions. The common state-of-art defenses to untargeted attacks are less than effective against Min-Max in Single-Server FL, therefore their failure when adapted to MSFL was foreseen. However, it was curious to observe that sometimes these defenses were inferior in mitigating the Min-Max than FedMes without any defense. This was then superimposed by the failure of to safeguard the global model. This underscores the potential insufficiency of existing defenses tailored for Single Server FL when applied to MSFL.

Our study, while informative, does not definitively answer the central research question outlined in Section 1. As it is

Table 2: Table of Mean Test Accuracy and Standard Deviation from 3 runs of each FMes-Defense with Attack case 2.

	Cifar10				Fashion-MNIST			
	AlexNet		VGG		AlexNet		VGG	
Aggregation Strategy	**Accuracy**	σ	**Accuracy**	σ	**Accuracy**	σ	**Accuracy**	σ
FMes-Median	19.4	0.69	16.5	0.56	26.1	3.44	17.3	1.23
FMes-Krum	25.2	7.65	19.1	1.34	30.7	4.97	25.5	8.2
FMes-Multi-Krum	18.8	0.42	12.9	1.09	24.7	14.08	15.7	1.85
FMes-Trimmed-Mean	20.7	1.06	14.8	1.48	40.0	8.49	16.2	2.96
FMes-Bulyan	20.4	1.0	11.9	2.01	23.0	1.79	13.0	1.18
FMes-DnC	19.3	1.06	17.4	0.36	39.4	10.5	17.4	1.56

possible that alternative approaches to extending the examined defenses could produce different results. Nevertheless, it suggests that extending Single-Server FL defenses to MSFL effectively is a difficult task to say the least. Highlighting the nuanced challenges introduced by MSFL, which existing defenses are ill-prepared to mitigate.

7 Responsible Research

The attack and defense nature of the paper raises questions surrounding the ethically of conducting such research. The primary concerns revolve around:

- *Potential Exploitation by Malicious Parties* - There is a risk that malicious parties could exploit the findings of our research to target vulnerabilities in future production MSFL. This poses implications, including financial losses and potential threats to the integrity of critical systems. With potential widespread adoption of MSFL with FedMes, it is imperative to conduct thorough research on its robustness against attacks before it's deployment. Despite the evident benefits of such an algorithm, the absence of published research addressing its security since it's initial publication in December 2021 is concerning. Therefore, it is paramount to investigate these risks diligently to prevent ambitious businesses from deploying MSFL with FedMes without fully comprehending the associated vulnerabilities.
- *Integrity in Reporting* - Ensuring the utmost integrity in reporting our research findings is essential to prevent the fabrication of a false sense of security or insecurity regarding the robustness of MSFL with the FedMes aggregation. As neutral researchers, free from external influences or motivations, our commitment lies solely in the pursuit of truthful and transparent inquiry. To uphold this integrity, our study prioritizes transparency by openly addressing the limitations of our research. Additionally, to facilitate reproducibility and scrutiny, we have made the experimental environment, including all code and raw results, freely accessible on GitHub[1].

By addressing these ethical considerations conscientiously, we aim to contribute meaningfully to the advancement of knowledge while mitigating potential risks and promoting integrity in research practices.

8 Limitations

The main identified limitations are:

1. The experiments were only run on iid data. Especially given the fact that MSFL in practice would likely work with non-iid. To note is DnC in Single-Server FL is not robust against Min-Max with non-iid data.
2. Only one symmetric topology was used in the experiments. In addition to other symmetric topologies, evaluating performance of defenses for asymmetric topologies could reveal new interesting insights.
3. The study was limited to image classification tasks. Perhaps trying similar experiments with language models would yield different results.
4. Evaluating the alternative to FedMes, Multi-Server FedAvg aggregation would be a valuable addition [12]. We focus only on FedMes as we hypothesise it is more vulnerable due to the weights associated with overlapping regions. So if it could be made safe, we assume FedAvg can too.
5. The extended defenses could have also been evaluated against other untargeted attacks as-well as other types of attacks.
6. More state-of-art defenses can be extended to the setting.
7. We did not investigate effects of applying a defense on the aggregation that happens on the client devices when they aggregate updates from the edge servers they reach before training. This too could have undiscovered positive effects on robustness.

9 Conclusion

In conclusion, this study investigated the effectiveness of state-of-the-art defenses originally designed for Single-Server Federated Learning (FL) in mitigating the Min-Max attack in the context of Multi-Server Federated Learning (MSFL) with FedMes. Despite developing the new FMes-Defenses by extending existing defenses (FedMes, including Median, Krum, Multi-Krum, Bulyan, Trimmed-Mean, and DnC) to MSFL, none were able to adequately preserve the accuracy of the global model.

Notably, DnC, which is effective in the Single Server FL context, proved ineffective against the Min-Max attack when

[1] https://github.com/Todor-cmd/rp-msfl

adapted to MSFL as FMes-DnC. Suggesting **defenses designed for Single-Server FL are ineffective against Min-Max in MSFL with FedMes**. Within the constraints of the defenses evaluated and approach used to used to extend them to MSFL.

The research highlights a need for novel defense mechanisms tailored to the unique challenges of this setting. Future research should focus on addressing the limitations identified in this study (section 8) and developing innovative defenses specifically designed to mitigate the effects of the Min-Max attack on the global model in MSFL with FedMes.

References

[1] K. Martineau, "What is federated learning?" 2022, accessed on 13-11-2023. [Online]. Available: https://research.ibm.com/blog/what-is-federated-learning

[2] D.-J. Han, M. Choi, J. Park, and J. Moon, "Fedmes: Speeding up federated learning with multiple edge servers," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3870–3885, 2021.

[3] L. Lyu, H. Yu, X. Ma, C. Chen, L. Sun, J. Zhao, Q. Yang, and P. S. Yu, "Privacy and robustness in federated learning: Attacks and defenses," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–21, 2022.

[4] Y. Fraboni, R. Vidal, and M. Lorenzi, "Free-rider attacks on model aggregation in federated learning," *CoRR*, vol. abs/2006.11901, 2020. [Online]. Available: https://arxiv.org/abs/2006.11901

[5] V. Shejwalkar, A. Houmansadr, P. Kairouz, and D. 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)*, 2022, pp. 1345–1371.

[6] V. Shejwalkar and A. Houmansadr, "Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning." Network and Distributed Systems Security (NDSS), 2021.

[7] H. Zhu and Q. Ling, "Byzantine-robust aggregation with gradient difference compression and stochastic variance reduction for federated learning," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 4278–4282.

[8] H. Zhou, Y. Zheng, H. Huang, J. Shu, and X. Jia, "Toward robust hierarchical federated learning in internet of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 5, pp. 5600–5614, 2023.

[9] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Communications Surveys Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[10] M. Talebkhah, A. Sali, M. Marjani, M. Gordan, S. J. Hashim, and F. Z. Rokhani, "Edge computing: Architecture, applications and future perspectives," in *2020 IEEE 2nd International Conference on Artificial Intelligence in Engineering and Technology (IICAIET)*, 2020, pp. 1–6.

[11] M. S. H. Abad, E. Ozfatura, D. GUndUz, and O. Ercetin, "Hierarchical federated learning across heterogeneous cellular networks," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 8866–8870.

[12] Z. Qu, X. Li, J. Xu, B. Tang, Z. Lu, and Y. Liu, "On the convergence of multi-server federated learning with overlapping area," *IEEE Transactions on Mobile Computing*, vol. 22, no. 11, pp. 6647–6662, nov 2023.

[13] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/f4b9ec30ad9f68f89b29639786cb62ef-Paper.pdf

[14] D. Yin, Y. Chen, K. Ramchandran, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," 03 2018.

[15] G. Baruch, M. Baruch, and Y. Goldberg, "A little is enough: Circumventing defenses for distributed learning," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2019/file/ec1c59141046cd1866bbbcdfb6ae31d4-Paper.pdf

[16] M. Fang, X. Cao, J. Jia, and N. Gong, "Local model poisoning attacks to Byzantine-Robust federated learning," in *29th USENIX Security Symposium (USENIX Security 20)*. USENIX Association, 2020, pp. 1605–1622. [Online]. Available: https://www.usenix.org/conference/usenixsecurity20/presentation/fang

[17] V. Chen, V. Pastro, and M. Raykova, "Secure computation for machine learning with spdz," 01 2019.

[18] YongchangWang and L. Zhu, "Research and implementation of svd in machine learning," in *2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS)*, 2017, pp. 471–475.

[19] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *CoRR*, vol. abs/1708.07747, 2017. [Online]. Available: http://arxiv.org/abs/1708.07747

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., vol. 25. Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

[21] S. Gupta, "Vgg-11 architecture," opengenus.org, https://iq.opengenus.org/vgg-11/, May 2020, [Online; accessed 15-Jan-2023].

[22] S. Liu and W. Deng, "Very deep convolutional neural network based image classification using small training sample size," in *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, 2015, pp. 730–734.

A Results from all runs

Table 3: Test Accuracy for all cases and all runs.

	Cifar10						Fashion-MNIST					
	AlexNet			VGG			AlexNet			VGG		
Aggregation Strategy/ Attack Case	**Run 1**	**Run 2**	**Run 3**	**Run 1**	**Run 2**	**Run 3**	**Run 1**	**Run 2**	**Run 3**	**Run 1**	**Run 2**	**Run 3**
No-Attack	50.5	50.9	51.3	52.7	53.0	52.0	81.9	81.8	84.0	82.0	82.1	81.7
Attack case 1	21.0	20.7	20.3	13.8	15.6	16.3	40.8	31.5	34.0	15.8	14.5	14.2
Attack case 2	12.1	20.0	19.4	13.7	16.7	14.9	20.7	28.3	21.2	16.8	16.5	14.7
FMes-Median	19.6	20.2	18.6	16.9	16.8	15.7	2.8	28.3	21.2	16.9	16.0	19.1
FMes-Krum	36.0	19.6	19.9	18.7	20.9	17.7	24.2	36.3	31.6	36.9	21.3	18.2
FMes-Multi-Krum	18.8	18.3	19.4	13.3	13.9	11.4	16.5	44.5	13.1	16.1	17.8	13.3
FMes-Trimmed-Mean	22.6	20.5	20.2	16.3	15.3	12.8	42.2	49.1	28.7	15.1	20.3	13.3
FMes- Bulyan	21.0	19.0	21.2	9.4	14.3	12.0	22.3	21.3	25.6	13.1	14.4	11.6
FMes-DnC	18.3	20.6	20.6	17.8	17.0	17.6	37.1	27.9	53.3	15.6	18.8	18.3