H2020 – ICT-13-2018-2019

Machine Learning to Augment Shared Knowledge in Federated Privacy-Preserving Scenarios (MUSKETEER)

Grant No 824988

D6.3 Security of Federated Machine Learning Algorithms

November 21

Imprint

Contractual Date of Delivery to the EC: **30th November 2021**

Author(s): **Luis Muñoz-González (Imperial College London), Muhammad Zaid Hameed (Imperial College London), Alexander Matyasko (Imperial College London), Emil Lupu (Imperial College London), Ambrish Rawat (IBM), Giulio Zizzo (IBM), Mark Purcell (IBM)**
Participant(s): **Imperial College London, IBM, Tree, UC3M**
Reviewer(s): **Angel Navia Vázquez (UC3M), Giacomo Fecondo (FCA-ITEM)**

Project: **Machine learning to augment shared knowledge in federated privacy-preserving scenarios (MUSKETEER)**

Work package: **WP6**
Dissemination level: Internal
Version: **1.0**

Contact: **Emil Lupu –** **e.c.lupu@imperial.ac.uk**
Website: **www.MUSKETEER.eu**

Legal disclaimer

The project Machine Learning to Augment Shared Knowledge in Federated Privacy-Preserving Scenarios (MUSKETEER) has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 824988. The sole responsibility for the content of this publication lies with the authors.

Copyright

© MUSKETEER Consortium. Copies of this publication – also of extracts thereof – may only be made with reference to the publisher.

Executive Summary

This deliverable D6.3 – Security of federated machine learning algorithms – is the only deliverable for task T6.3 (Assessing the security of machine learning algorithms under the different privacy operation modes) in WP6. This includes a report with a comprehensive evaluation of the robustness of the different algorithms developed in the MUSKETEER Machine Learning Library (MMLL) against different attacks both at training (poisoning attacks) and test time (evasion attacks). The assessment is performed for both supervised and unsupervised learning tasks across the different Privacy Operation Modes (POMs) considered in the project. The defensive mechanisms evaluated in this deliverable are already described in D5.4 and D5.5.

Document History

Version	Date	Status	Author	Comment
0.1	25 Nov 2021	For internal review	Imperial/IBM	First draft
	26 Nov 2021	Completed	Angel Navia Vázquez Giacomo Fecondo	Internal review
0.2	29 Nov 2021	Completed	Imperial/IBM	Changes made following review comments received.
1.0	30 Nov 2021	Final review	IBM	Final

Table of Contents

List of Figures

List of Tables

List of Acronyms and Abbreviations

Abbreviation	Definition
ADAM	Adaptive Moment Estimation
AFA	Adaptive Federated Averaging
COMED	Coordinate-wise Median
DNN	Deep Neural Network
FA	Federated Averaging
MMLL	MUSKETEER Machine Learning Library
MNIST	Modified National Institute of Standards and Technology dataset
NN	Neural Network
CNN	Convolutional Neural Network
PGD	Projected Gradient Descent
POM	Privacy Operation Mode

1 Introduction

1.1 Purpose

In deliverables D5.2 and D5.3 in WP5 we have shown that the standard federated learning techniques implemented in the MUSKETEER Machine Learning Library (MMLL) are vulnerable to different types of attacks. Thus, at training time, federated learning algorithms are vulnerable to poisoning attacks, where the behaviour and performance of the learning algorithms can be manipulated by the attackers by poisoning the datasets of some of the participants or by sending malicious model updates to the aggregator. Thus, as shown by [Blanchard et al. 2017], just a single attacker is enough to compromise standard aggregation methods such as Federated Averaging (FA) [McMahan et al. 2017]. At test time, similar to centralized machine learning algorithms, federated learning algorithms are also vulnerable to evasion attacks, where the attackers aim to produce errors in the predictions of the resulting machine learning model by injecting small perturbations to the original data points sent to the model for their evaluation. These manipulated data points are commonly known as adversarial examples [Biggio et al. 2013], [Szegedy et al. 2013].

In WP5 we have developed different mechanisms to defend and mitigate the effect of such attacks for the different algorithms in the MMLL library. At training time, we have included different robust aggregation mechanisms capable of defending against both data and model poisoning attacks as well as mechanisms to eliminate poisoning points in the datasets provided by the different clients at training time. We have also implemented strategies based on adversarial training [Madry et al. 2018] to defend against adversarial examples at test time, limiting significantly the success of evasion attacks. All these defensive mechanisms were described in deliverables D5.4 (for poisoning attacks) and D5.5 (for evasion attacks). Some of these defences are only available for certain Privacy Operation Modes (POMs). The reason is that the use of homomorphic encryption limits the type of mathematical operations that can be performed and, thus, many of these robust techniques cannot be implemented with the operations available when working with encrypted data. To still provide some protection even under such limitations, for example, in the case of poisoning attacks, we have used data pre-filtering techniques to mitigate the attacks. As discussed in D5.4, there is a trade-off between the privacy offered by the different POMs and their robustness to e.g., poisoning attacks.

In this deliverable we aim to provide a comprehensive assessment of the robustness of the different algorithms implemented in the MMLL library. Compared to the results reported in the deliverables associated to WP5, in this deliverable, D6.3, we provide a more systematic analysis of the robustness against both poisoning and evasion attacks for the algorithms and defensive techniques implemented across the different POMs and for the different learning tasks supported in the library, i.e., supervised and unsupervised learning. The assessment we

provide here and the way in which it has been conducted also provides a testing framework that can be used when assessing the robustness of algorithms deployed in a federated learning setting. Unless otherwise specified in subsequent sections, all the results reported are applicable to v2.2.0 of MMLL library (pre-release, dated 5th November 2021) and do not take into account modifications made after this date. In particular, there has not been sufficient time to evaluate and report on modifications made to the MMLL library (under the same version number) on the 17th and 24th November 2021.

The empirical evaluation reported in this deliverable endorses the capacity of the different defensive mechanisms implemented in the MMLL library to mitigate the effect of a comprehensive set of attacks, achieving the target figures described in the KPIs for the project for the robustness of the algorithms implemented in MUSKETEER i.e., mitigating effect of up to 20% malicious users, and even offering additional capabilities such as the one to identify the malicious users.

1.2 Related Documents

This deliverable is closely related to the work undertaken in WP5. Thus, we use the threat model described in deliverable D5.1 as a reference for describing the attack scenarios and settings used in our assessment of the robustness and security of the federated learning algorithms implemented in the platform. This assessment relies on the attacks considered and described in D5.2 for poisoning attacks at training time and D5.3 for evasion attack at test time.

The assessment in this deliverable not only includes the evaluation of the robustness of the standard federated learning algorithms developed in WP4, which, as shown in D5.2 and D5.3, are very brittle in the presence of an adversary, but also assesses the performance of the defensive techniques developed in WP5 (see D5.4, D5.5, D5.6 and D5.7) when the system is under attack.

Compared to previous deliverables in WP5, in this deliverable we aim to provide a more comprehensive evaluation of the security and robustness of the MMLL across the different POMs and for different machine learning tasks, including supervised and unsupervised learning.

1.3 Document Structure

The rest of the document is structured as follows: In Section 2 we provide the details of the datasets used for the evaluation of the robustness of the algorithms in MMLL. In Section 3 we

present the security assessment of the supervised learning algorithms in MMLL library against poisoning attacks across the different POMs. Section 4 analyses the robustness of the unsupervised learning algorithms against poisoning attacks. Section 5 analyses the robustness provided by data pre-processors against data poisoning. Section 6 includes the evaluation of the robustness of the supervised algorithms in the platform against evasion attacks at test-time. Finally, Section 7 concludes this deliverable.

2 Datasets used for the evaluation.

We have used the following datasets in the different experiments in this deliverable:

- **MNIST dataset** [LeCun et al. 2010] which is a grayscale hand digit recognition dataset with 10 classes of digits ranging from 0 to 9. The input sample is a grey-scale image of size 28×28 and dataset contains 60000 training samples and 10000 test samples.
- BMNIST dataset which is transformed MNIST [LeCun et al. 2010] with even/odd classes. The dataset splits are identical to the MNIST dataset.
- **Fashion MNIST (F-MNIST) dataset** [Xiao et al. 2017] which is a grayscale image dataset with 10 classes representing fashion categories with class labels from 0 to 9. The input sample is a grayscale image of size 28×28 and dataset has 60000 training samples and 10000 test samples. It shares the same image size and number of classes as MNIST but is more challenging dataset.
- **2D-Synthetic dataset** which is a synthetic dataset with input samples consisting of 2 features from D4.6. This dataset will be used for training a K-Means clustering algorithm.
- **Pima dataset** which is a binary classification dataset with input samples consisting of 8 features from D4.6. This dataset will be used for testing our outlier detection schemes for training a logistic classifier.

3 Robustness of Supervised Learning Algorithms against Poisoning Attacks

In this section, we evaluate the robustness of the supervised learning algorithms in the MMLL library against data and model poisoning attacks across the different POMs, which includes algorithms for linear and non-linear classification and regression. Similar to D5.6 and D5.7, we

analyse the overall performance of standard Federated Averaging (FA) [McMahan et al. 2017], Coordinate-Wise Median (COMED) [Yin et al. 2018], and Adaptive Federated Averaging [Muñoz-González et al. 2019] aggregation rules, which are supported by the MMLL library.

This section is organised as follows: First, we describe the attacks used for our assessment. An in-depth review of all the attacks against supervised learning algorithms in federated settings are described and provided in deliverable D5.2. Second, we provide an overview of robust defences implemented to mitigate the attacks and the applicability of our robust defences across the different POMs in the MMLL library. The interested reader may refer to deliverable D5.4 for a detailed overview of robust defences. Finally, we report our empirical results for the algorithms implemented in the different POMs.

3.1 Summary of the Attacks used for the Assessment.

There are currently no known methods to evaluate the robustness of a federated machine learning algorithm other than subjecting them to known attacks and evaluating the impact of the attacks i.e., robustness cannot be measured a priori. Here we summarise the state-of-the-art attacks used for the evaluation (the detailed description and classification of data and model poisoning attacks in federated learning can be found in deliverable D5.2).

Label flipping attacks

In indiscriminate data poisoning attacks, the goal of the attacker is to degrade the performance of the trained model. A natural way to decrease the model's performance in supervised settings through data poisoning is to corrupt the labels of the training inputs by random discrete noise. For label-flipping attackers, we sample the labels for the training inputs from the multinomial distribution, so the malicious participant sends updates which increase the entropy of the prediction distribution. In the case of multiple colluding attackers, all the attackers use the same distribution to flip the labels.

Byzantine attacks

[Blanchard et al. 2017] proposed a model poisoning attack where the malicious participants sample the parameters from a random distribution and send the random values to the aggregator. The random values are drawn from a Gaussian distribution with a very large variance. Thus, when using standard aggregation rules, such as FA, that rely on computing a weighted average of the parameters, the large values of the malicious random model updates dominate the computation of the average. Even a single Byzantine attacker can completely

degrade the performance of the model. The effect is further exaggerated when multiple attackers collude.

Stealthy Model Poisoning attacks

[Bhagoji et al. 2019] proposed a stealthy model poisoning attack. The attacker's objective is to degrade the performance of the model maximally, while also ensuring that the malicious updates are similar to the benign updates. To evaluate the robustness of our models, we used a variation of [Bhagoji et al. 2019] stealthy model poisoning attack. Our attack has two steps. In the first step, we train the malicious client on the client's benign data to get the benign updates. Then, we trained the malicious client to maximize the training loss subject to the regularization that the updates are similar to the benign updates. Then, the attacker's objective is as follows:

$$\arg\min_{W_k} \; -\sum_{i=1}^{n_k} L({x_i}^k, {y_i}^k;\, W_k) \;+\; \lambda ||{W^b}_k - W_k||$$

where L is the training loss, ${W_k}^b$ is the benign updates, and λ is the regularisation strength. A higher value of λ decreases the attack's strength and makes the detection of malicious clients more difficult. We select the appropriate value of the regularisation weight λ using a grid search to maximise the effectiveness of the attack.

3.2 Summary of Defences Available

Standard Federated Averaging relies on the computation of the average of the values provided by the participants for each parameter in the machine learning model. The mean of values is very brittle and sensitive to the outliers. A single adversary or a single faulty user can dramatically affect the performance of the final model. Please refer to deliverable D5.4 to see the detailed discussion about the limitations of Federated Averaging. To overcome the vulnerability of the standard aggregation methods implemented in the MMLL library, we have implemented Robust API with support of two robust aggregation methods capable of mitigating the effect of poisoning attacks and faulty users: 1) coordinate-wise median and [Yin et al. 2018], and 2) Adaptive Federated Averaging [Muñoz-González et al. 2019].

Coordinate-wise median

[Yin et al. 2018] proposed robust aggregation method, so called COMED, which relies on the computation of the median of the values provided by the clients at each training iteration for each parameter in the machine learning model. The median of the values is robust statistics with a breakdown point of 0.5.

Adaptive Federated Averaging

COMED is an effective technique, but it does not allow to identify the malicious clients sending malicious or faulty model updates. This can be limiting from a practical perspective, as it can hinder the investigation of potential problems in the platform (compromised or malicious clients and faults) and to have mechanisms for clients' accountability. To overcome these limitations, in MUSKETEER we proposed Adaptive Federated Averaging (AFA) [Muñoz-González et al. 2019], a robust aggregation technique for federated learning in supervised learning settings that allows, not only to mitigate the effect of poisoning attacks, but also to identify the malicious or bad model updates at each training round. AFA uses Hidden Markov Model (HMM), which is used to estimate the probability of a client providing a good model update for every participant in the federated learning task. A comprehensive description of the AFA algorithm can be found in [Muñoz-González et al. 2019] and deliverables D5.4 and D5.7.

3.3 Applicability of the Defences for each POM

In Table 1. Applicability of defences for each POM in supervised learning tasks., we show the general applicability of the different defensive mechanisms implemented in the MMLL library to defend against poisoning attacks across the different POMs. Our robust defences are implemented using an abstract *RobustAPI* and support any parametric machine learning algorithms, including linear models and non-linear models, such as Neural Networks. To benefit from the defences we have implemented, an algorithm must be implemented to make use of the *RobustAPI*. This approach facilitates the deployment of the defences across several algorithms implemented in the library and enables users to make use of the defences when using MUSKETEER platform and developing their own customised solution. Unfortunately, our RobustAPI cannot be implemented for POM 2 and POM 3, where the computation of the aggregated model in the server is done in the encrypted domain, which has limited operations, and thus, robust aggregation methods cannot be applied. For POM 4 and POM 5 the applicability depends on the degree of access to the model updates that is awarded to the aggregator. POM 6 operates in multiple ways which significantly increases the difficulty of

providing support for its defence against malicious clients. When client updates are sent in a round robin fashion, robust aggregation methods cannot be applied because the aggregator does not have access to the individual updates. Similarly, when the information exchanged with the aggregator is not gradient based e.g., covariance matrices as specified in D4.7, robust aggregation techniques cannot be applied. Only, the Logistic Classifier and Multiclass Logistic Classifier models present under POM 6 in the MMLL library are gradient based. Unfortunately, support for the Robust API in these algorithms was not available at the time of the evaluation.

It is also important to note that in all the POMs where robust aggregation methods are not possible, label sanitization and/or outlier detection can be used to mitigate as far as possible poisoning attacks. The MMLL library can also be easily extended to support novel aggregation rules.

Table 1. Applicability of defences for each POM in supervised learning tasks.

POM	Label Sanitization	Outlier Detection	Robust API
POM 1	✔	✔	✔
POM 2	✔	✔	N/A
POM 3	✔	✔	N/A
POM 4	✔	✔	N/A*
POM 5	✔	✔	✔*
POM 6	✔	✔	✔*

3.4 Assessment of POM 1

Here, we report the experimental evaluation for POM 1. We evaluated the robustness of two neural network models trained on MNIST and F-MNIST datasets under normal operation and against label-flipping attackers, byzantine attacks with strength $s = 5.0$, and stealthy model poisoning attackers. The two neural networks have the following network architecture:

- A network with two fully connected layers (256 and 64 units). We will refer to this architecture as NN in our experiments.
- A network with two convolutional layers (8 and 16 filters). We will refer to this architecture as CNN in our experiments.

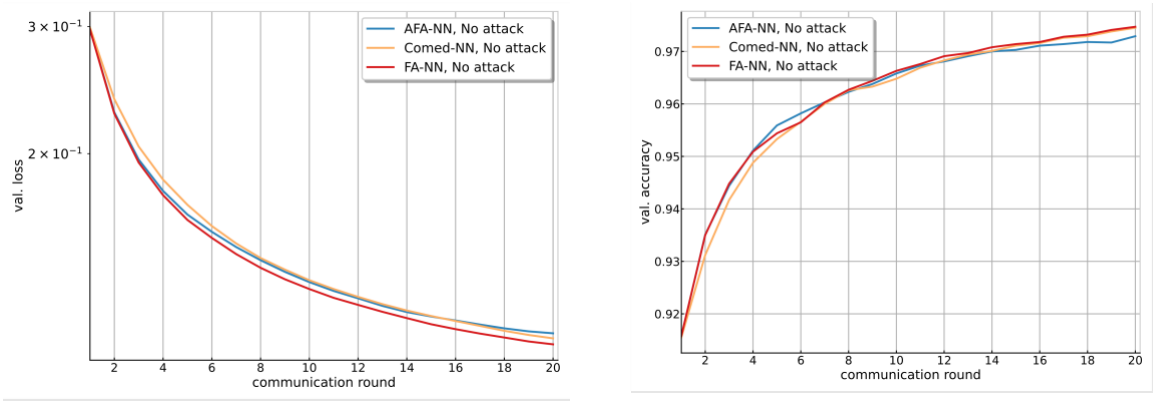
We used the same training parameters in all experiments. We trained the models for 20 communication rounds with 30 participants. The number of local epochs is 10, and the batch size is 128. The local optimizer is Adam with a learning rate of 0.0003. The number of malicious

participants is set to 20% of all participants, i.e., 6 out of 30 participants. We consider four types of colluding attack scenarios, similar to those scenarios used in D5.6 and D5.7:

1. All malicious participants perform a label flipping attack using the same noise distribution to flip the labels.
2. All malicious participants send random model updates drawn from the same noise distribution with $s = 5.0$.
3. All malicious participants send stealthy model updates fitted the same malicious data. The regularisation weight λ is set to $1e^{-4}$ and 10 for NN and CNN, respectively.
4. Three sets of colluding attackers, where 2 malicious participants are label-flipping attackers, 2 malicious participants are byzantine, and the remaining 2 participants are stealthy model poisoning attackers.

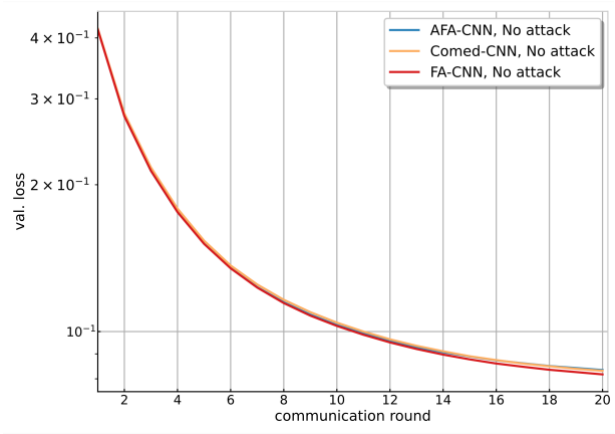
3.4.1 MNIST dataset

In this section, we present the results of the assessment of the MMLL library on the MNIST dataset with all benign participants, with 6 label flipping attackers, with 6 Byzantine attackers, with 6 stealthy model poisoning attackers, and with a combination of all attackers in Figure 1-Figure 10. As we can see in Figure 1 and Figure 2, our robust aggregation methods do not impact the model's performance when all participants are benign. In Figure 3 and Figure 4, we present the results of federated learning in the presence of label flipping attackers. The results confirm that FA is not robust to label flipping attackers. We can observe that both AFA and COMED are able to mitigate label flipping attackers and produce a model with the performance similar to the model learned with all benign participants. Also note, that AFA successfully detects and blocks all label flipping attackers at communication round 6 (this event is shown in the plots with a red cross).

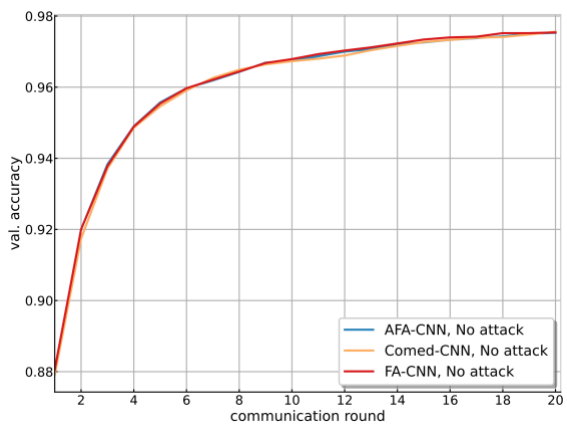


(a) validation loss **(b) validation accuracy**

Figure 1: Validation performance of the NN model on MNIST when all participants are benign

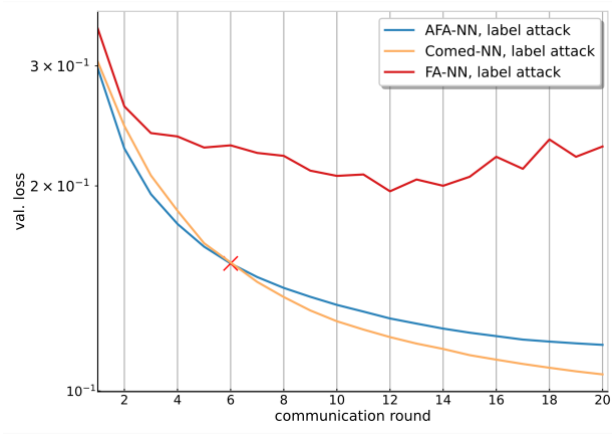


(a) validation loss

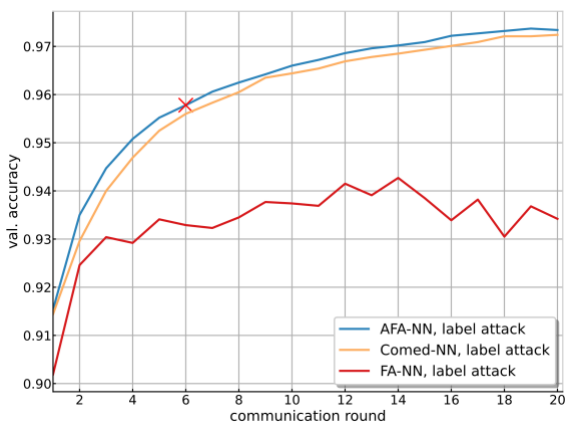


(b) validation accuracy

Figure 2: Validation performance of the CNN model on MNIST when all participants are benign

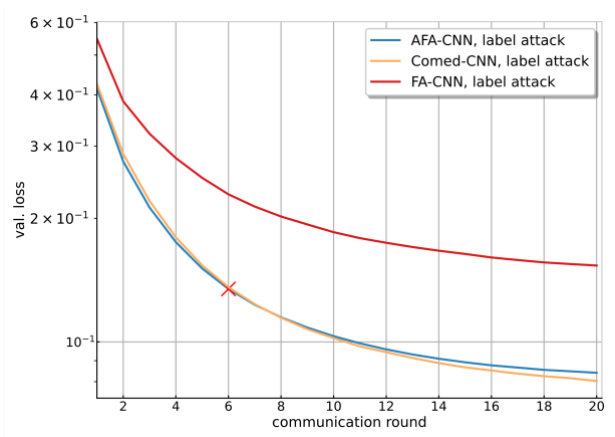


(a) validation loss

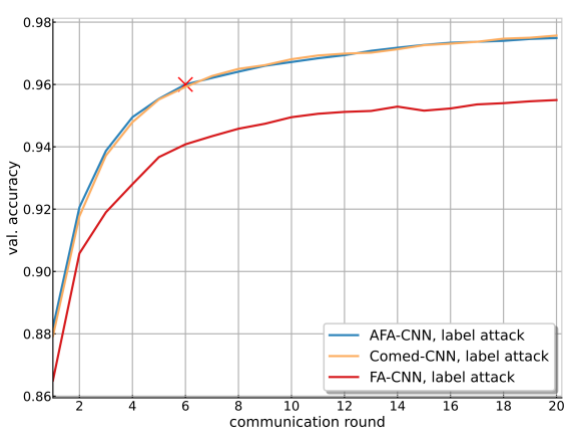


(b) validation accuracy

Figure 3: Validation performance of the NN model on MNIST against 6 untargeted label flipping attackers



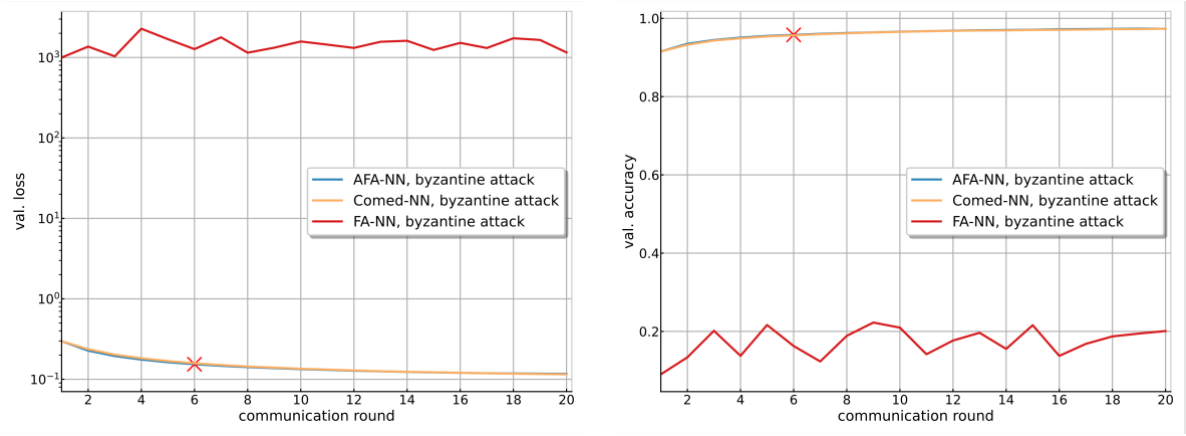
(a) validation loss



(b) validation accuracy

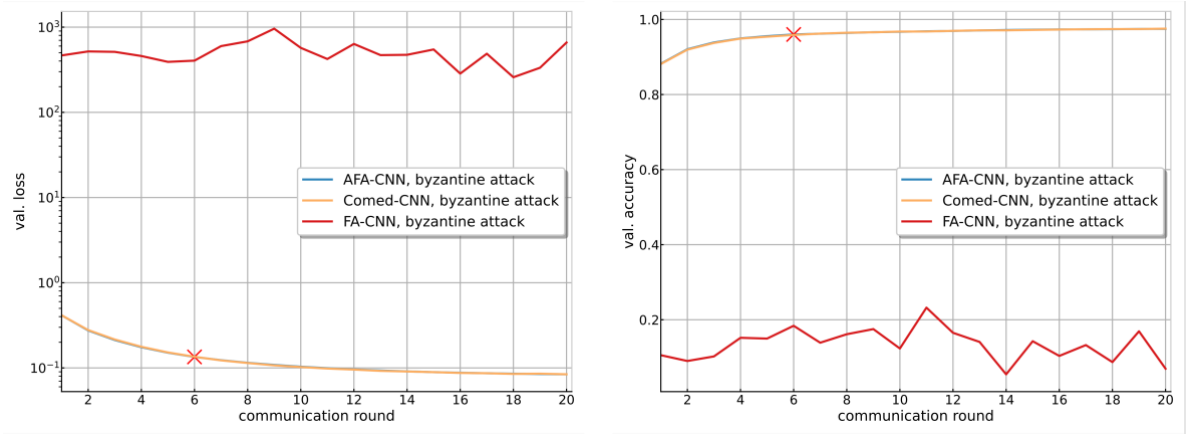
Figure 4: Validation performance of the CNN model on MNIST against 6 untargeted label flipping attackers

In Figure 5 to Figure 8, we present the results of federated learning with Byzantine model poisoning and stealthy model poisoning attackers. Coordinated Byzantine attack significantly decreases the model's validation loss and accuracy when using non-robust aggregation rule i.e., FA. In contrast, our two defensive methods significantly reduce the influence of Byzantine attackers and allow to train the model with a performance close to the baseline model trained when all clients are benign. The stealthy model poisoning attack is the strongest model poisoning attack that we considered in the experiments. AFA aggregation rule outperforms COMED defence against stealthy model poisoning attacks.



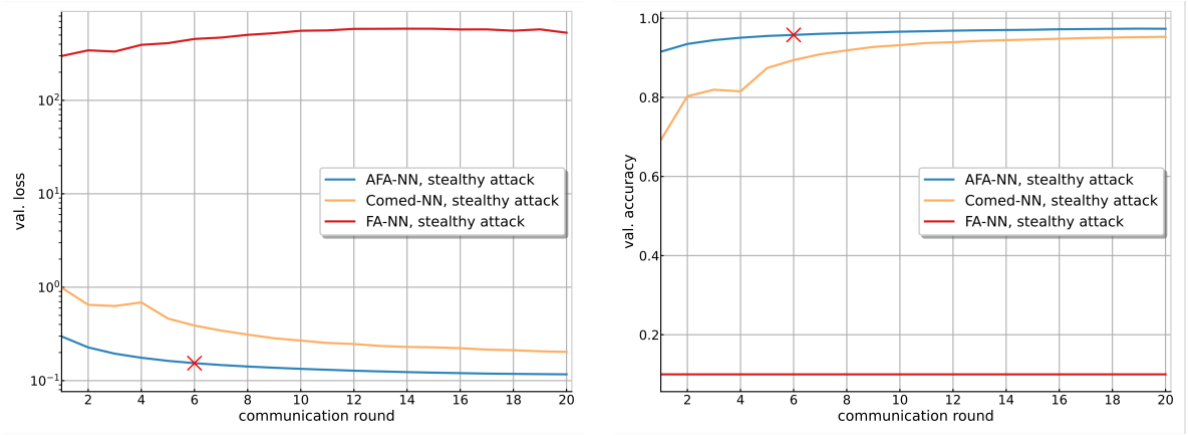
(a) validation loss **(b) validation accuracy**

Figure 5: Validation performance of the NN model on MNIST against 6 Byzantine attackers



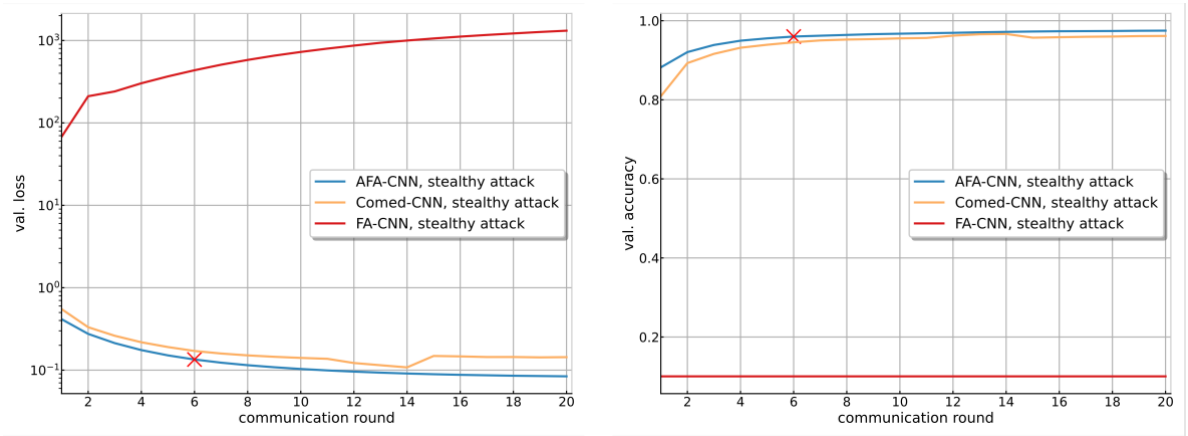
(a) validation loss **(b) validation accuracy**

Figure 6: Validation performance of the CNN model on MNIST against 6 Byzantine attackers



(a) validation loss **(b) validation accuracy**

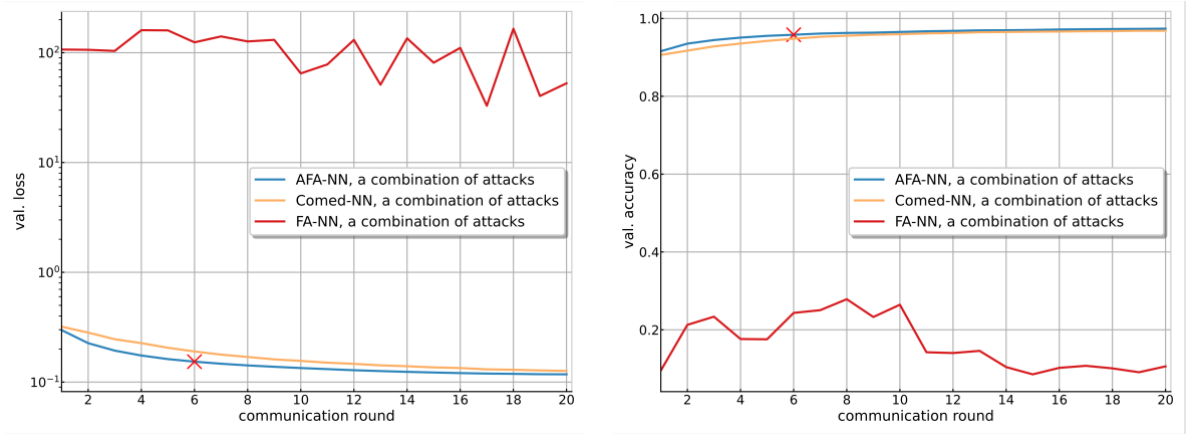
Figure 7: Validation performance of the NN model on MNIST against 6 model stealthy attackers



(a) validation loss **(b) validation accuracy**

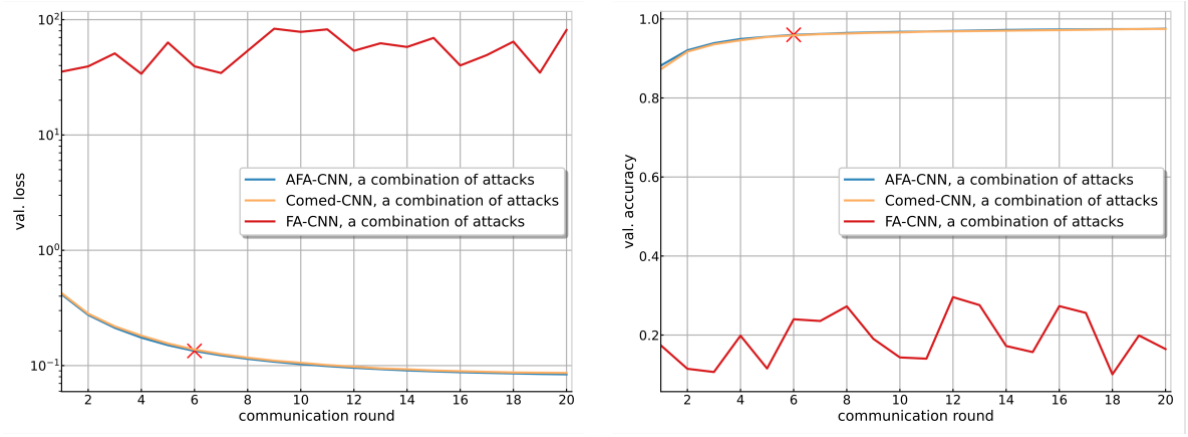
Figure 8: Validation performance of the CNN model on MNIST against 6 model stealthy attackers

Finally, in Figure 9 and Figure 10, we present the results of the empirical evaluation against a combination of all attackers. Both COMED and AFA are able to successfully mitigate three groups of colluding attackers, which demonstrate the effectiveness of our defensive methods.



(a) validation loss **(b) validation accuracy**

Figure 9: Validation performance of the NN model on MNIST against a combination of all attackers

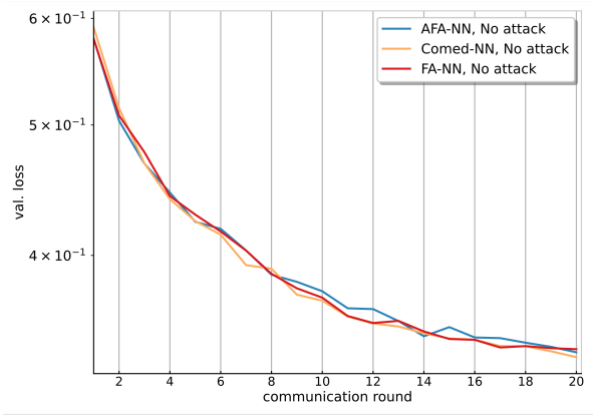


(a) validation loss **(b) validation accuracy**

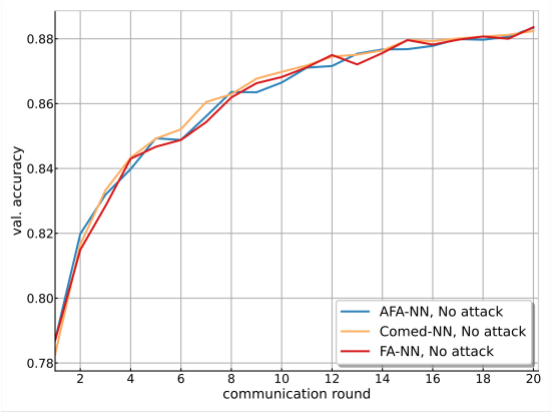
Figure 10: Validation performance of the CNN model on MNIST against a combination of all attackers

3.4.2 Fashion MNIST dataset

In this section, we present the results of the assessment of the MMLL library on the F-MNIST dataset with all benign participants, with 6 label flipping attackers, with 6 Byzantine attackers, with 6 stealthy model poisoning attackers, and with a combination of all attackers in Figure 11 to Figure 20. The results are similar to the results on the MNIST dataset. Without any defence in the presence of the attackers, the performance of Federated Averaging can be drastically impacted by malicious participants.

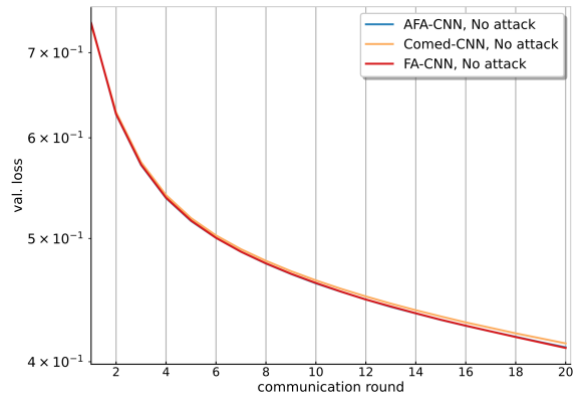


(a) validation loss

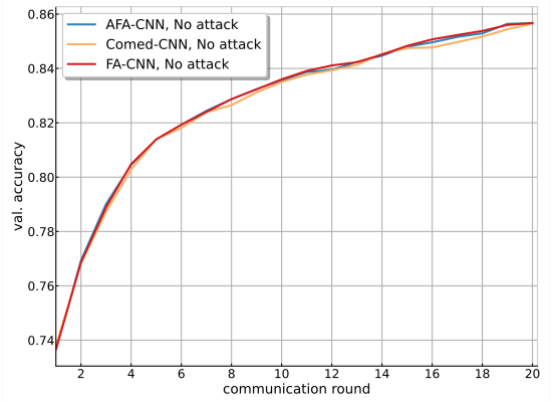


(b) validation accuracy

Figure 11: Validation performance of the NN model on FMNIST when all participants are benign

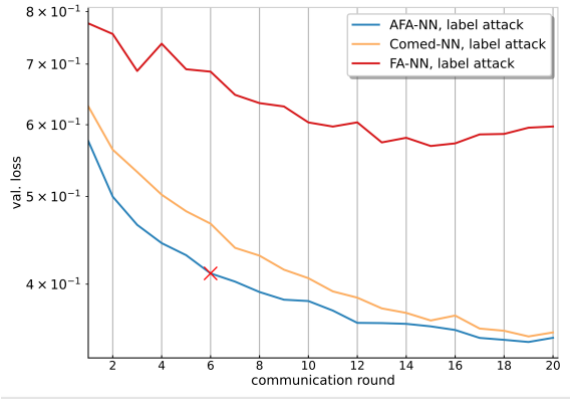


(a) validation loss

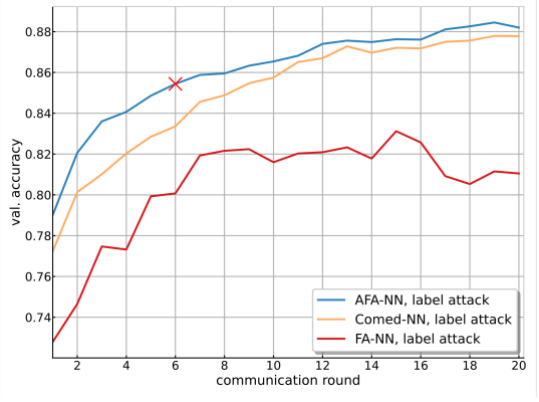


(b) validation accuracy

Figure 12: Validation performance of the CNN model on FMNIST when all participants are benign

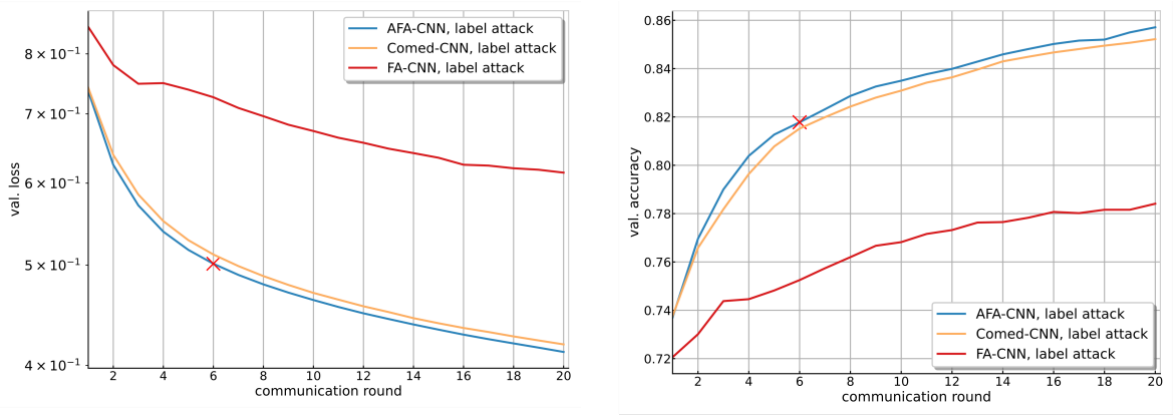


(a) validation loss



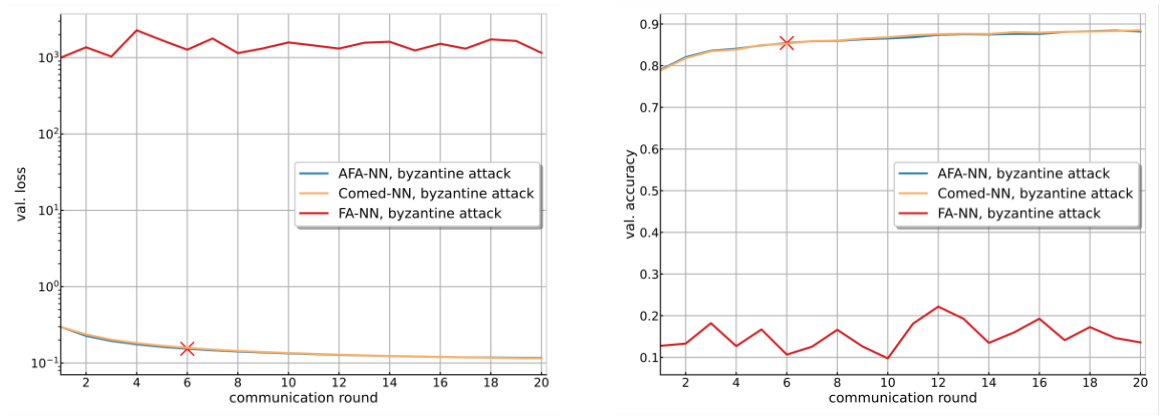
(b) validation accuracy

Figure 13: Validation performance of the NN model on FMNIST against 6 untargeted label flipping attackers



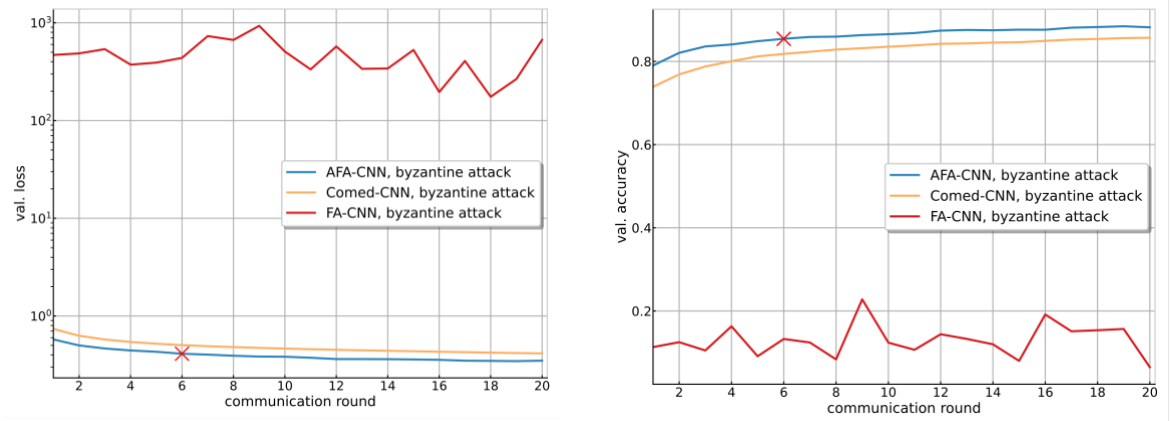
(a) validation loss (b) validation accuracy

Figure 14: Validation performance of the CNN model on FMNIST against 6 untargeted label flipping attackers



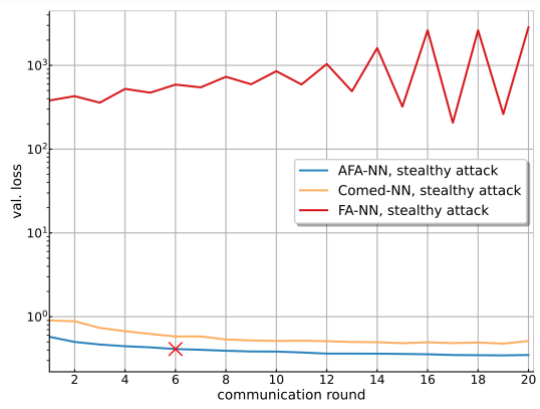
(a) validation loss (b) validation accuracy

Figure 15: Validation performance of the NN model on MNIST against 6 Byzantine attackers

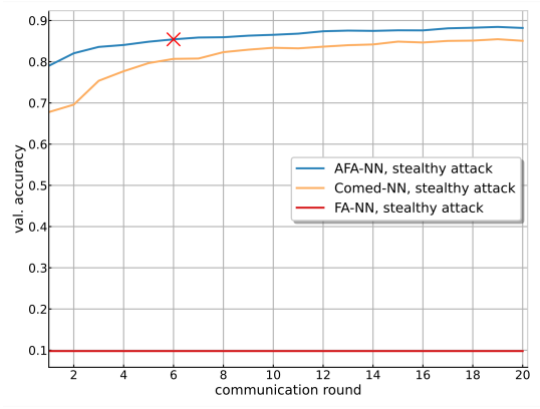


(a) validation loss (b) validation accuracy

Figure 16: Validation performance of the CNN model on FMNIST against 6 Byzantine attackers

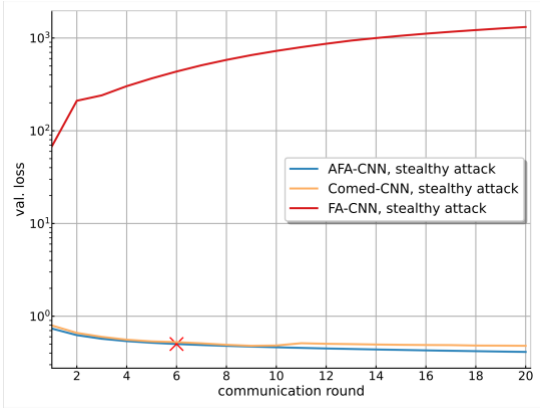


(a) validation loss

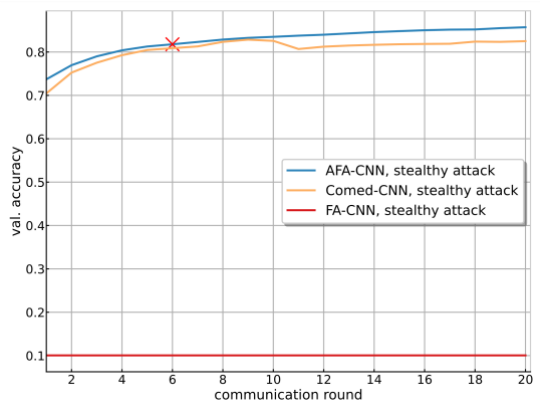


(b) validation accuracy

Figure 17: Validation performance of the NN model on FMNIST against 6 model stealthy attackers

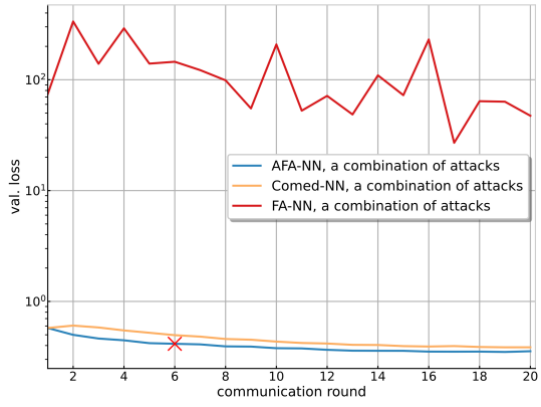


(a) validation loss

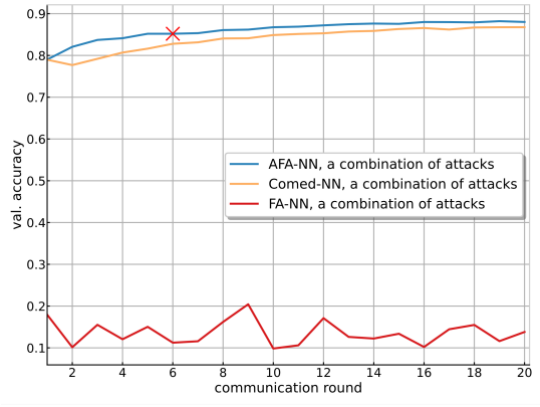


(b) validation accuracy

Figure 18: Validation performance of the CNN model on FMNIST against 6 model stealthy attackers

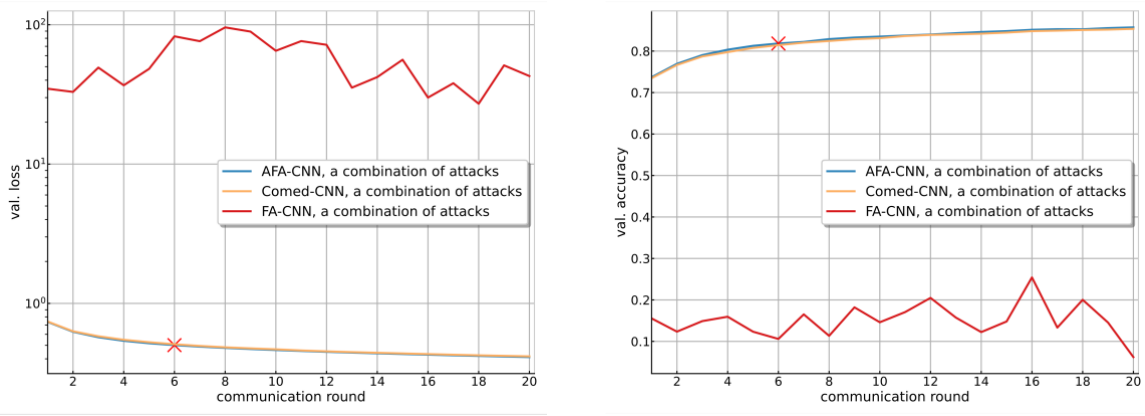


(a) validation loss



(b) validation accuracy

Figure 19: Validation performance of the NN model on FMNIST against a combination of all attackers



(a) validation loss **(b) validation accuracy**

Figure 20: Validation performance of the CNN model on FMNIST against a combination of all attackers

Overall, as we demonstrated in this section, our robust defences can mitigate the effect of multiple colluding or groups of colluding attackers for all models and datasets. In this section, we presented the results only for Neural Networks, but similar results are expected for other supervised linear parametric models supported by the MMLL library, such as Support Vector Machine (SVM), Logistic Classifier (LC), Linear Regression (LR), as we also showed in deliverable D5.4. In this sense, the scenarios considered here with the two neural network architectures are more challenging for the defender, compared to the case of linear classifiers, as the number of parameters is significantly higher, which gives more opportunities to the adversary to craft more successful attacks, for example, by using the stealthy poisoning attack. Furthermore, our robust API is flexible and can also be used in POMs 4 and 5 in those cases where the central aggregator receives unencrypted model updates.

3.5 Assessment of POMs 2 and 3

POMs 2 and 3 require the aggregator to compute all the operations in the encrypted domain. Thus, as explained before, robust aggregation schemes are not available as they cannot be applied as, some of the operations required to implement these algorithms are not supported in the encrypted domain with homomorphic encryption. However, as described in Table 1, in the MMLL library we have implemented alternative mechanisms to defend against poisoning attacks in these situations (see deliverable D5.4 for more details): outlier detection and label sanitisation. The evaluation of the robustness of these techniques is presented in Section 5.

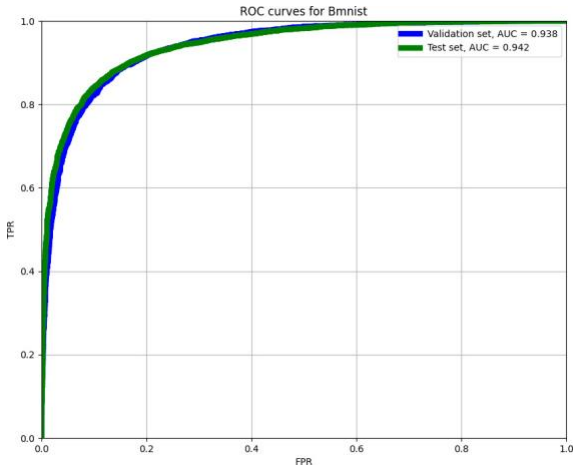
3.6 Assessment of POMs 4 and 5

In POM 4, the aggregator node uses an additively homomorphic cryptosystem to protect the confidentiality of the data and uses the support of a crypto node, which operates on blinded data for unsupported operations (D4.7, p11). The aggregator (called MN in D4.7) is assumed not to be able to collude with the crypto node. To our understanding, this mode of operation does not make POM 4 suitable for the application of aggregation-based defences. In particular, several operations necessary for aggregation-based defences such as comparisons are not supported by the homomorphic cryptosystem. In contrast, in POM 5, it is the client (referred to as WN in D4.7) which carries out the operations needed by the aggregator using the homomorphic cryptosystem and the aggregator has access to the model (D4.7, p12). We have therefore focussed our evaluation on POM 5 and report the results below. Although use of the *RobustAPI* was not implemented for this algorithm, we have manually adapted the implementation of the algorithm to be able to carry out the evaluation. If, under POM 4 the aggregator is given sufficient access to the model updates to enable the use of our robust aggregation techniques, we expect the results to be similar.

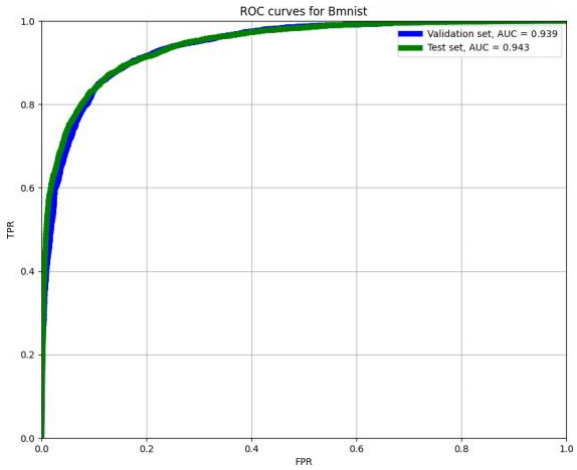
We evaluated the robustness of Logistic Classifier (LC) trained on BMNIST dataset under normal operation and against label-flipping and Byzantine attackers.

Our experimental parameters are as follows: We trained the Logistic Classifier (LC) model for 50 communication rounds with 5 participants. We stopped the federated learning when the relative difference between the model parameters between communication rounds was less than 0.01. We have considered two types of attack scenarios: 1) one participant is a label-flipping attacker; 2) one participant is a Byzantine attacker with strength $s = 5.0$.

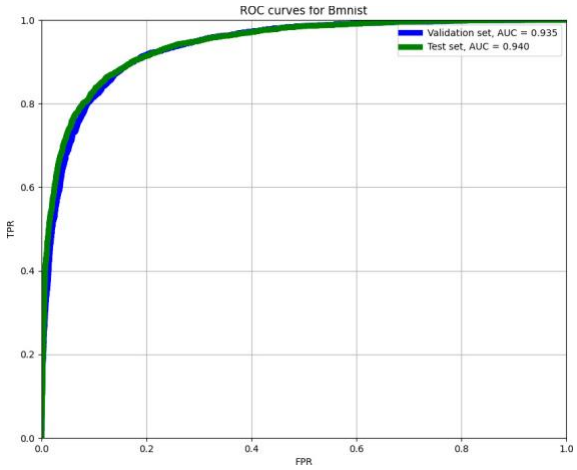
In Figure 21: The ROC curves of the logistic classifier trained with all benign participants on BMNIST using FA (a), COMED (b), AFA (c), respectively we present the results of training the logistic classifier in POM5 federated settings when all participants are benign. As we can see, the ROC curves of our robust defences in Figure 21: The ROC curves of the logistic classifier trained with all benign participants on BMNIST using FA (a), COMED (b), AFA (c), respectively are similar to the ROC curve of the model trained with standard federated averaging in Figure 21: The ROC curves of the logistic classifier trained with all benign participants on BMNIST using FA (a), COMED (b), AFA (c), respectively. It means that our defence does not negatively affect the performance of the final model. Next, we report our results for two attack scenarios with label flipping and byzantine attackers.



(a)



(b)

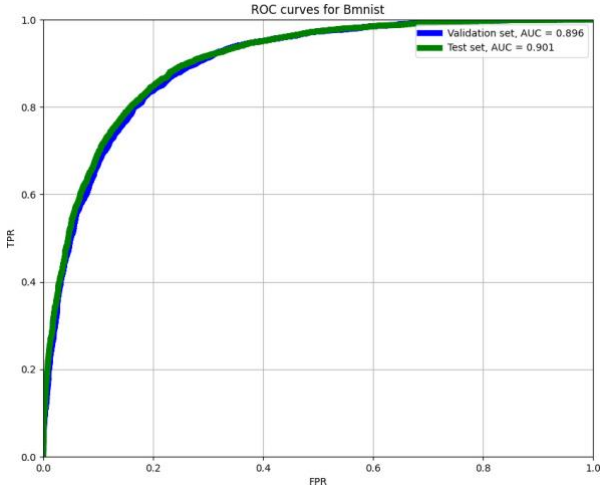


(c)

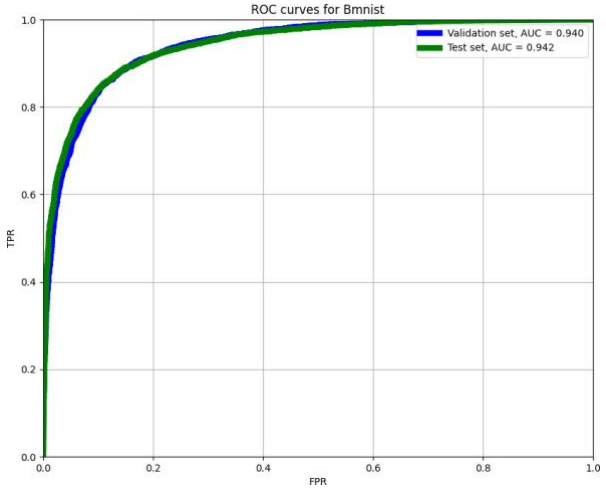
Figure 21: The ROC curves of the logistic classifier trained with all benign participants on BMNIST using FA (a), COMED (b), AFA (c), respectively

In Figure 21: The ROC curves of the logistic classifier trained with all benign participants on BMNIST using FA (a), COMED (b), AFA (c), respectively and Figure 23, we present the results of training the logistic classifier in POM5 federated settings when one participant is label-

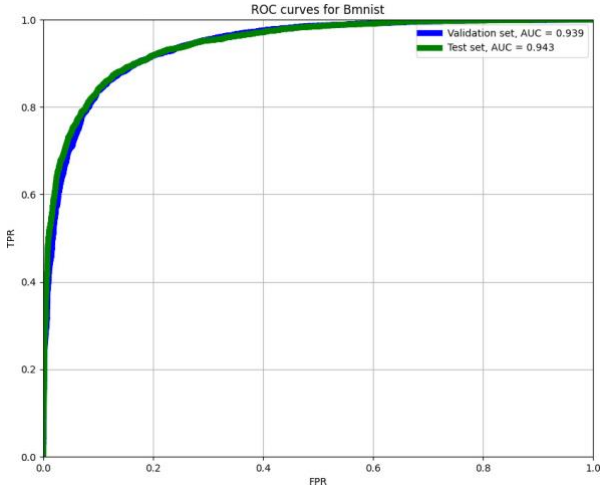
flipping or byzantine attacker, respectively. Unsurprisingly, standard gradient averaging is not robust to label-flipping and byzantine attackers. Label-flipping and byzantine attackers decrease AUC on the test set from 0.942 to 0.901 and 0.564, respectively. The byzantine attack is much stronger since it can directly manipulate the averaged model, while the label-flipping attack indirectly influences the aggregated model. Against a single byzantine attacker in Figure 23, the final model trained with a gradient averaging has AUC on the test set similar to the coin flip. On the other hand, our defences are able to mitigate both label-flipping and byzantine attackers, which can be seen from the respective ROC curves in Figure 21: The ROC curves of the logistic classifier trained with all benign participants on BMNIST using FA (a), COMED (b), AFA (c), respectively and Figure 23. In this section, we evaluated the Logistic Classifier (LC) model, but our assessment results are expected to be similar to other parametric models supported by the MMLL library.



(a)

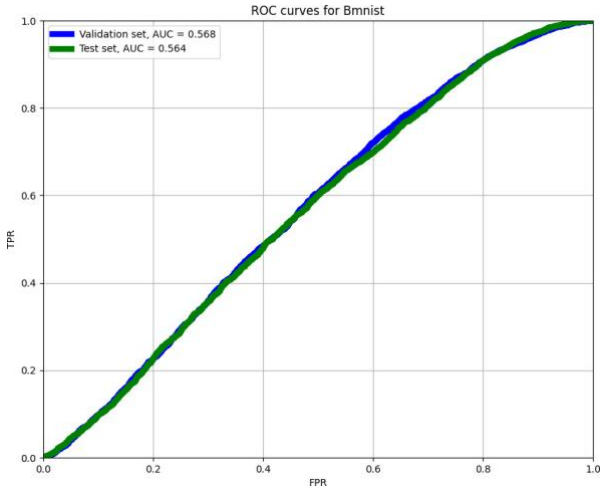


(b)

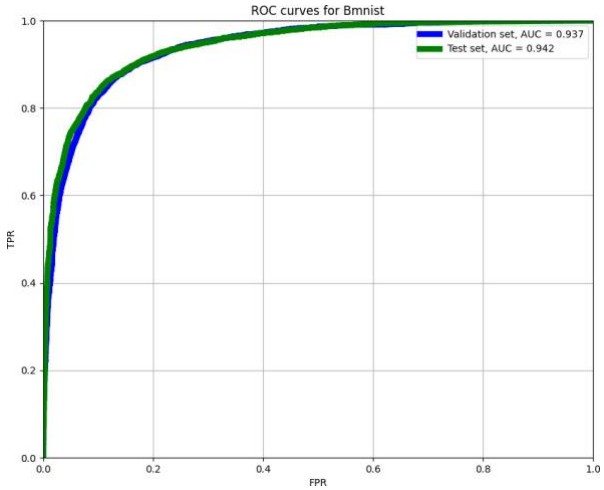


(c)

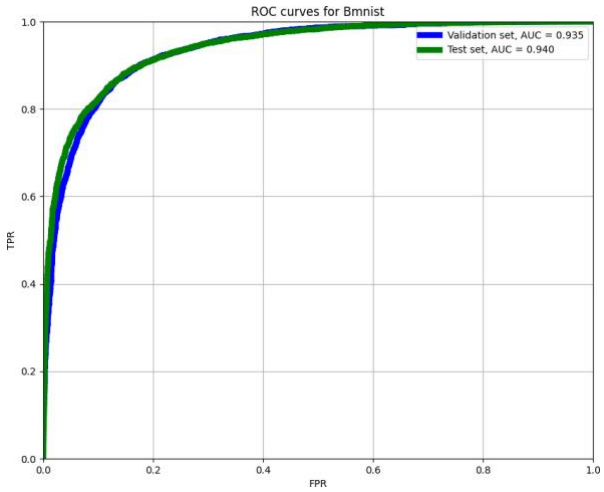
Figure 22: The ROC curves of the logistic classifier trained with one label-flipping attacker on BMNIST using FA (a), COMED (b), AFA (c), respectively



(a)



(b)



(c)

Figure 23: The ROC curves of the logistic classifier trained with one Byzantine attacker on BMNIST using FA (a), COMED (b), AFA (c), respectively

3.7 Assessment of POM 6

As mentioned in Section 3.3 above, POM 6 operates in multiple ways. As specified in D4.7 p. 13, *"POM6 is not a general procedure, it requires that every algorithm is implemented from scratch, and it is not guaranteed that any algorithm can be implemented under POM6."* It is therefore impossible to assess the robustness of POM 6 algorithms in the general case and every single algorithm implemented needs to make use of the defences according to its mode of operation. When a round-robin approach is being used or when the information revealed to the aggregator does not consist in gradients, aggregation methods cannot be employed. Only, the Logistic Classifier and Multiclass Logistic Classifier models present under POM 6 in the MMLL library are gradient based. Unfortunately, Version 2.2.0 of MMLL library on November 5 does not have the API implementation for *RobustAPI* for these algorithms. We have therefore not been able to include these algorithms in the evaluation for supervised ML models in POM 6. Note that in all cases data pre-filtering techniques such as outlier detection and label sanitisation can be employed to provide some level of protection. The evaluation of the protection offered by such techniques is detailed in Section 5.

4 Robustness of Unsupervised Learning Algorithms against Poisoning Attacks

In this section we evaluate the robustness of the unsupervised learning algorithms in the MMLL against poisoning attacks across the different POMs. Similar to D5.6 and D5.7, we analyse the performance of the robust K-Means clustering algorithm (see deliverable D5.4) implemented for the MMLL library against poisoning attacks with colluding adversaries, which represent a more dangerous threat for the system. For the experiments, we consider a scenario where 30 clients are participating in training a clustering model for 20 communication rounds and each client updates the centroids using their local training data before sending the updated centroids back to the aggregator. The aggregator, after receiving centroids from all clients, computes the aggregated centroids using the corresponding aggregation rule. Total number of Byzantine or malicious clients participating in the training are 6 which corresponds to 20% of the total clients participating in the training. We consider MNIST, F-MNIST and 2D-synthetic datasets for evaluation. For MNIST and F-MNIST, K-means clustering algorithm is run to estimate 20 centroids, and for 2D-syntehtic dataset it estimates 6 centroids. Furthermore, we use the MMLL library and *pycloudmessenger* to train the model in this federated learning setting. To reduce the computational power and time required for training the model on a desktop machine emulating 30 participating clients and a server, we use a reduced training datasets for both MNIST and F-MNIST (10,000 random training samples instead of 60,000 training samples of complete training dataset) and split them equally among all clients. On the other hand, because of this reduced training dataset size, defending against attacks is a more challenging task as the ML algorithm can only use a smaller subset of training data to train the model at each client.

4.1 Summary of the Attacks used for the Assessment.

Here we summarise the attacks used for the evaluation (the detailed description can be found in D5.2). For creating the different scenarios, we consider two different types of attacks for the evaluation of the proposed robust clustering scheme. These attacks were already used for previous deliverables in WP5 (see deliverables D5.2, D5.4, D5.6 and D5.7).

Indiscriminate data poisoning attack

In indiscriminate data poisoning attacks, as the goal of the attacker is to degrade the performance of the trained model for a large set of inputs. A very simple (yet effective) indiscriminate attack, previously described in deliverable D5.2, to compromise K-Means consists on manipulating the value of the training data of the malicious participants by using

the following transformation: for every input sample i of the malicious participant k, we compute $x_i'^k = clip(1 - x_i^k, 0, 1)$, where the $clip$ function keeps the sample value in valid input range [0, 1]. This completely shifts the distribution of the data points for the malicious participants compared to the benign ones.

Random model updates

In this type of poisoning attack, the malicious participant k sends a uniform random noise i.e., $c_t'^k = \mathcal{U}_{N_c \times p}[0, 1]$ as the model centroids update to the central node, where $\mathcal{U}_{N_c \times p}[a, b]$ denotes the uniform distribution on $[a, b]^{N_c \times p}$ and N_c denotes the number of cluster centroids and p denotes the centroid dimension. In colluding attacks, the set of malicious participants send malicious centroids following the same noise distribution.

4.2 Summary of Defences Available

Here we summarise the techniques available to defend against poisoning attacks in K-Means clustering algorithm (the detailed description can be found in D5.4). First, we discuss the implementation details of the robust clustering scheme that has been proposed in D5.4, designed to mitigate poisoning attacks. We consider a local outlier threshold of 0.05 during the robust centroids' initialization process at the participants with 5 repetitions for the centroids initializations, choosing the centroid initialization with minimum distortion for each client. This local outlier detector aims to mitigate the effect of outliers and poisoning points introduced in the local training sets of the different participants. At the aggregator, we consider an aggregator outlier threshold of 0.1 and we take a minimum threshold of 10 participants (in the federated learning tasks used in the experiments there are a total of 30 participants) for supporting a centroid which equals to one third of total participants in the training. This mechanism is necessary to mitigate the effect of poisoning attacks aiming at creating fake clusters supported by a set of colluding attackers. Furthermore, we consider that the aggregator runs K-Means clustering algorithm for 10 steps to estimate the aggregated centroids from all received centroids. For comparison, we use the standard (non-robust) implementation of K-means clustering algorithm in the platform. In POMs where we cannot employ robust clustering scheme, we can use our data prefiltering schemes to provide robustness against indiscriminate data poisoning attacks.

On the other side, in those scenarios where robust K-Means is not available (see below), we have used outlier detection as an alternative to mitigate poisoning attacks by filtering out the malicious training data points that could be present in the local datasets provided by the participants.

4.3 Applicability of the Defences for each POM

In the tables below we show the applicability of the different defensive mechanisms implemented in the MMLL library to defend against poisoning attacks across the different POMs.

Table 2. Applicability of defences for each POM in unsupervised learning tasks.

POM	Outlier Detection	Robust clustering
POM 1	✔	✔
POM 2	✔	N/A
POM 3	✔	N/A
POM 4	✔	✔*
POM 5	✔	✔*
POM 6	✔	✔

The outlier detector can be used across all the POMs in the library. For K-Means the outlier detector can be applied by labelling all the data points with the same class label, as it were a classification problem with a single class.

In POMs 2 and 3 the use of the robust clustering algorithm is not possible, as it requires to perform mathematical operations in the aggregator that are not available in the encrypted domain. On the other hand, for POMs 4 and 5, if the aggregator can get unencrypted centroids through the *cryptonode* in POM 4 or by decrypting themselves in POM 5, then robust K-Means clustering algorithm can be applied in both POMs.

For POM 6, Robust K-means algorithm can be applied by integrating the *RobustAPI* in POM 6 in a manner similar to POM 1. An integration of the *RobustAPI* into the algorithms under POM 6 was not available in time for this evaluation so we have conducted an evaluation using a previous version of the MMLL library as detailed below.

4.4 Assessment of POM 1

Here we report the experimental evaluation for POM 1. We consider three different types of colluding attack scenarios, similar to those scenarios used in D5.6 and D5.7:

1. All malicious participants perform the indiscriminate data poisoning attack.
2. All malicious participants send random model updates back to the aggregator using the same noise distribution.
3. Two sets of colluding attackers, where 50% of malicious participants perform an indiscriminate data poisoning attack and the remaining 50% of malicious participants send random model updates to the aggregator using the same noise distribution.

For analysing the effectiveness of the proposed defences, we will resort to visual inspection of the centroids learnt by the aggregator. We noticed that, given the dimensionality and characteristics of the benchmarks considered, traditional metrics, such as the distortion, are not very informative to diagnose and reflect the impact of the attack on the resulting model. However, the visual inspection of the centroids clearly shows the significance and impact of the attack on the resulting models.

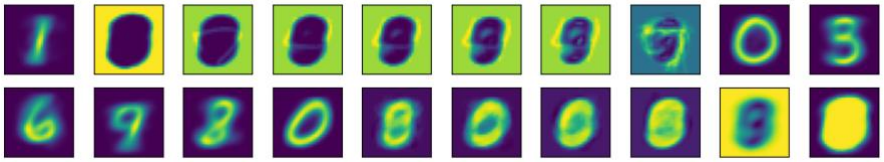
4.4.1 MNIST Dataset

Figure 24 shows the results for standard (non-robust) K-means clustering algorithm performance in the presence of colluding adversaries. Figure 24 (a) shows the centroids learned when colluding attackers perform an indiscriminate data poisoning attack. It can be seen that 7 of the resulting centroids are completely dominated by the flipped pixel values introduced by the colluding attackers and do not contain any of the valid digit class representation. On the other hand, 2 centroids (8^{th} in 1^{st} row and 10^{th} in 2^{nd} row) are not dominated by the flipped pixel values (introduced by the colluding attackers) but, nonetheless, they do not contain any useful data representation.

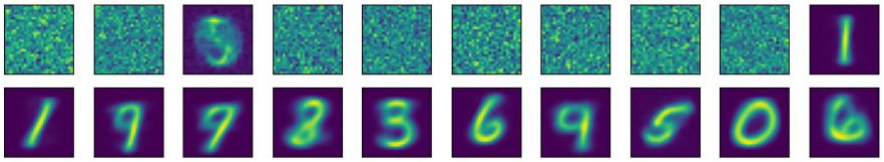
Similar results can be observed for Figure 24 (b) which shows that learnt centroids for the case when colluding malicious clients perform a random model update attack. Here, it can be observed that 8 centroids contain just random noise and 1 centroid (3^{rd} in 1^{st} row) contains a digit that is distorted beyond recognition and does not contain any useful representation.

Finally, Figure 24 (c) shows the resulting centroids for the case when there are multiple groups of colluding attackers in the federated learning setup. These groups have different malicious objectives i.e., one group of colluding attackers performs indiscriminate data poisoning attack, whereas the other group of attackers' crafts random model updates. By inspecting the centroids in Figure 24 (c), we find that both attacks have been successful in compromising the centroids learned according to their malicious objective at the same time. For example, the 3^{rd} and the 5^{th} centroid in the 1^{st} row represent centroids successfully attacked by a random

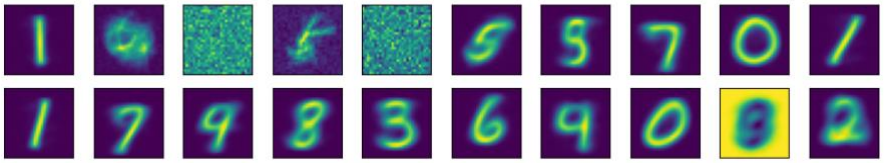
model update attack whereas 9th centroid in the 2nd row contains a centroid that is inserted by the indiscriminate data poisoning attackers. We also observe that since in this case, there are two different group of attackers present in the training and each has attack category has only 50% of the total attackers, the overall number of centroids compromised in this setting (Figure 24 (c)) are less compared to when only one group of attackers is present (Figure 24 (a) and (b)).



(a) Scenario 1: All malicious clients employ indiscriminate data poisoning attack



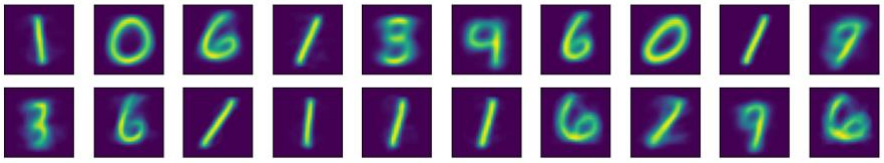
(b) Scenario 2: All malicious clients send random model updates as learnt centroids



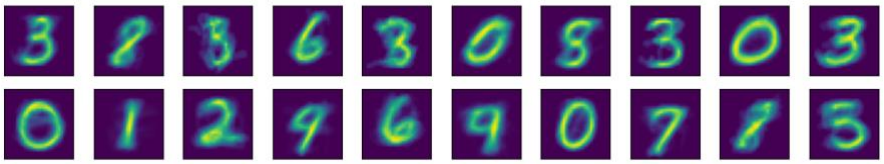
(c) Scenario 3: 50% of malicious clients employ indiscriminate data poisoning attack and the remaining 50% of malicious participants send random model updates as centroids.

Figure 24: Learnt centroids for MNIST dataset using standard (non-robust) K-Means clustering algorithm

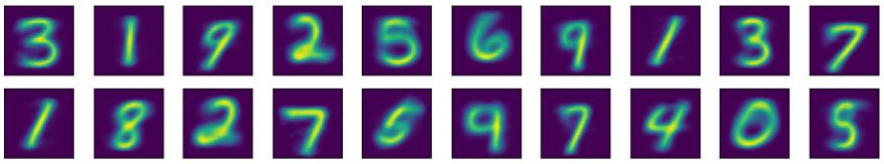
Figure 25 shows the results for the three scenarios when proposed robust clustering has been applied. We observe that in all three scenarios, the learnt centroids from proposed robust K-means clustering algorithm do not contain any centroid compromised from either the indiscriminate data poisoning attack or random model updates attack.



(a) Scenario 1: All malicious clients employ indiscriminate data poisoning attack



(b) Scenario 2: All malicious clients send random model updates as learnt centroids



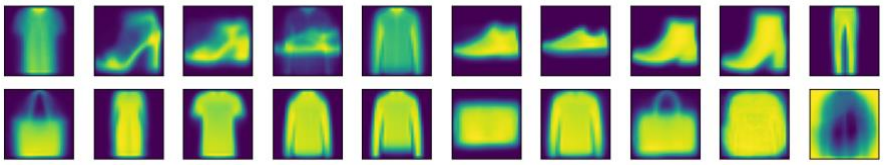
(c) Scenario 3: 50% of malicious clients employ indiscriminate data poisoning attack and the remaining 50% of malicious participants send random model updates as centroids.

Figure 25: Learnt centroids for MNIST dataset using proposed robust K-Means clustering algorithm

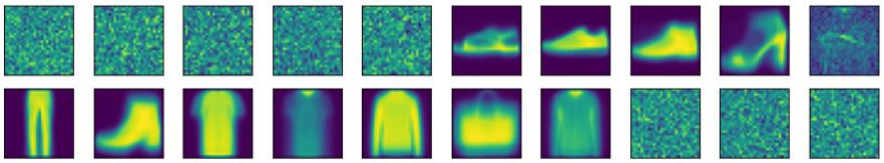
4.4.2 F-MNIST Dataset

Figure 26 shows the centroids learned in the three attack scenarios described before. From Figure 26 (a) we observe that only one centroid (10^{th} centroid in the 2^{nd} row) is completely dominated by the indiscriminate data poisoning attack and majority of the other centroids are valid images corresponding to different F-MNIST dataset classes. On the other hand, the attack with random model updates is more successful against F-MNIST dataset, as 9 learnt centroids in Figure 26 (b) only contain noise. Finally, Figure 26 (c) shows the learnt centroids for the scenario when two groups of attackers are present in the training task: one group of attackers is performing an indiscriminate data poisoning attack and the other group of attackers is sending random model updates. It can be seen that both types of attacks are able to compromise the training process, as 4 centroids show random noise (corresponding to the random model updates attack), and 1 centroid shows flipped pixel values (corresponding to the indiscriminate data poisoning attack). We also observe the similar trend as in MNIST dataset that the total number of centroids corrupted by the random model updates attack is

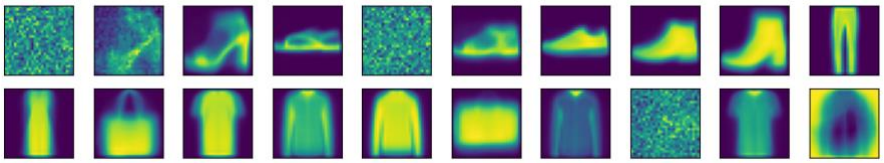
smaller in Figure 26 (c) compared to the case when only a random model update attack is performed during training i.e., 4 vs 9 corrupted centroids.



(a) Scenario 1: All malicious clients employ indiscriminate data poisoning attack



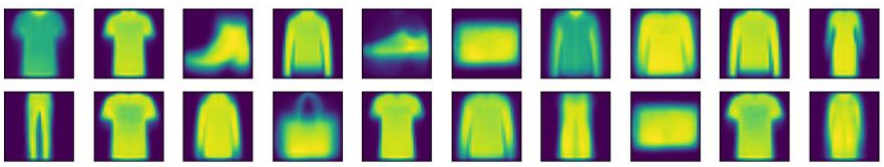
(b) Scenario 2: All malicious clients send random model updates as learnt centroids



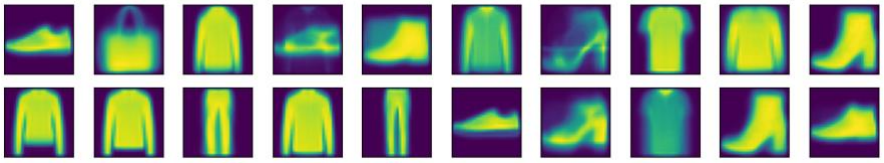
(c) Scenario 3: 50% of malicious clients employ indiscriminate data poisoning attack and the remaining 50% of malicious participants send random model updates as centroids.

Figure 26: Learnt centroids for F-MNIST dataset using standard (non-robust) K-Means clustering algorithm

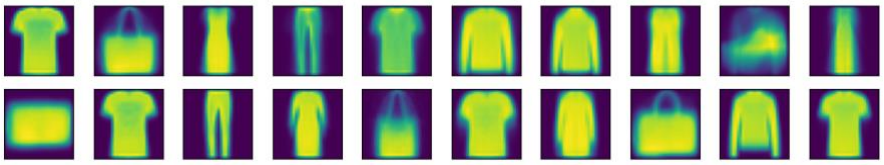
Figure 27 shows the learnt centroids when using robust K-means. It can be seen that all learnt centroids in three scenarios correspond to valid images from F-MNIST dataset and none of the indiscriminate data poisoning attack or random model update attack has been able to compromise any centroid.



(a) Scenario 1: All malicious clients employ indiscriminate data poisoning attack



(b) Scenario 2: All malicious clients send random model updates as learnt centroids



(c) Scenario 3: 50% of malicious clients employ indiscriminate data poisoning attack and the remaining 50% of malicious participants send random model updates as centroids.

Figure 27: Learnt centroids for F-MNIST dataset using proposed robust K-Means clustering algorithm

The results for MNIST and F-MNIST datasets show that standard (non-robust) K-means clustering algorithm is very vulnerable to different types of attackers and multiple groups of colluding attackers can easily infiltrate and poison the training process. As a result, the integrity of trained model is compromised and, in many cases, it will not learn a very useful representation of the data. On the other hand, proposed robust K-means clustering algorithm successfully defends against these different attackers and the learnt representations of the data are very similar to the case when all participating clients are benign. On the other side, the results in Figure 25 and Figure 27 show that the quality of the centroids learned with robust K-Mean is comparable to the quality of the non-compromised centroids learned by the standard implementation. In other words, the use of K-Means does not seem to affect the quality of the centroids learned.

4.5 Assessment of POMs 2 and 3

POMs 2 and 3 require the aggregator to compute its operations in the encrypted domain, where some basic operations, such as comparisons, are not available. Thus, it is not possible to implement our robust clustering algorithm there. However, in this case, as shown in Table 2, outlier detection can be applied to filter-out suspicious points from the training sets provided by the participants. The results on how outlier detection helps to mitigate attacks is analysed in Section 5.1.

4.6 Assessment of POMs 4 and 5

In POMs 4 and 5, the use of robust clustering is limited to those cases where the aggregator works with unencrypted data. Otherwise, as in POM 2 and 3, outlier detection (see Section 5.1) must be used to defend against poisoning attacks.

The algorithms available in the MMLL library under these POMs up to 5th November 2021 did not implement the *RobustAPI*. However, for those cases where robust K-Means can be applied, the training of the robust clustering algorithm is analogous to POM 1. Thus, in terms of performance, we would expect to obtain the same results under POMs 4 and 5 as under POM 1 (see Section 4.4).

4.7 Assessment of POM 6

Here we report the experimental evaluation for POM 6. This experiment has been performed in MMLL library version 0.5.0 (the same as was used for deliverable D5.4) as the K-means clustering algorithm available did not use the *RobustAPI*. However, the API designed for robust clustering is available for integration in the MMLL library in subsequent versions. The API has been used in the assessment of POM 1 in this deliverable and can be directly integrated in POM 6 when the implementation of the clustering algorithms follows the same rules.

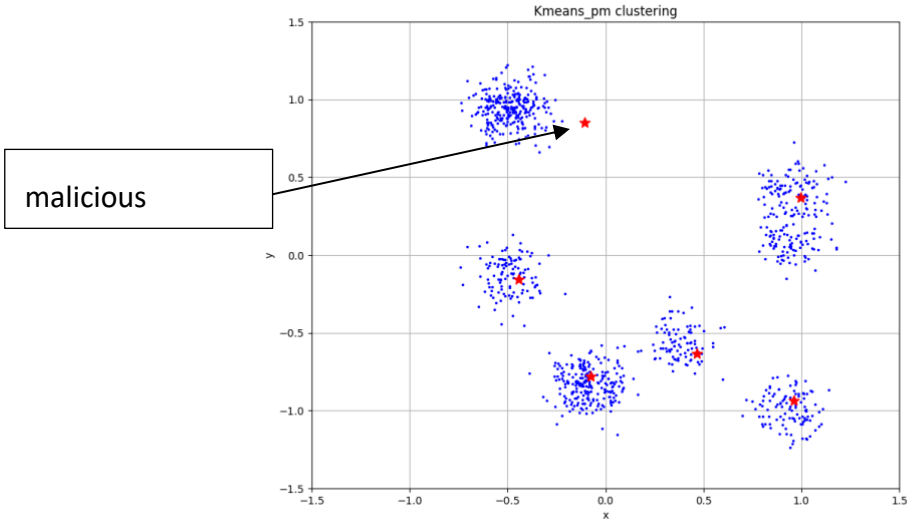
As previously described in the assessment of POM 1 (Section 4.4), we consider three different types of colluding attack scenarios:

1. All malicious participants perform the indiscriminate data poisoning attack.
2. All malicious participants send random model updates back to the aggregator using the same noise distribution.
3. Two sets of colluding attackers, where 50% of malicious participants perform an indiscriminate data poisoning attack and the remaining 50% of malicious participants send random model updates to the aggregator using the same noise distribution.

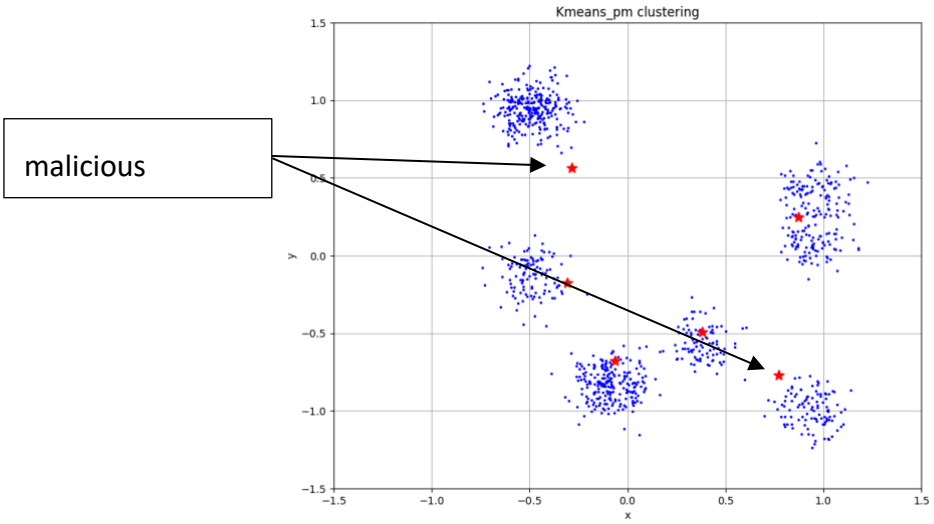
We consider the evaluation of the K-means clustering algorithm on 2D-synthetic dataset that allows us to get a better visualization perspective on how different attacks compromise the centroids estimated in standard (non-robust) K-means clustering algorithm in MMLL library and the performance of our proposed robust K-means clustering algorithm which mitigates different poisoning attacks.

Figure 28 (a-c) shows the learnt centroids (red circles) on top of the clean training data (blue circles) for standard (non-robust) K-Means clustering algorithm for different colluding attack scenarios. It can be seen that the colluding attackers in all three scenarios can successfully manipulate the centroids. However, centroids are not severely compromised especially in case of indiscriminate data poisoning attack where only 1 out of 6 centroids is not supported by the distribution of data. This is because the low dimensionality of the dataset and the fact that the data is spread across a broad region of the space of valid data points. In the case of scenario 2 (random model updates attack) and scenario 3 (both random model update and indiscriminate data poisoning attack), 2 out of 6 centroids are not supported by the data distribution. As mentioned before, the apparent lower success of the attacks (compared to MNIST and F-MNIST datasets) can be attributed to low dimensional and (slightly) symmetrical nature of training data, and data range in [-1, 1], as the learnt centroids are not severely compromised by the indiscriminate poisoning attack which flips and clip the sample value in range [0, 1] or the random model updates which send centroids from uniform distribution $\mathcal{U}_{N_c \times p}[-1.5, 1.5]$. But, in all cases, the attack is still able to compromise the integrity of trained model.

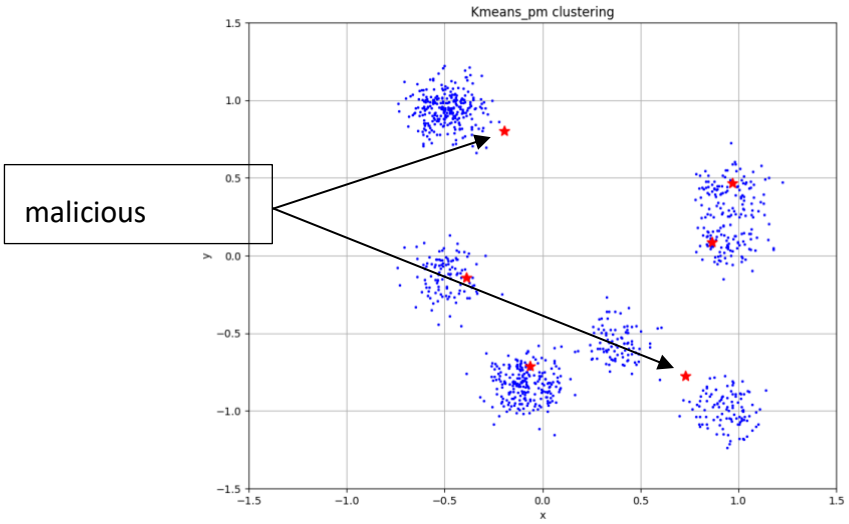
On the other hand, Figure 29 shows that centroids estimated by the proposed robust federated K-Means algorithm are in the support of the distribution of benign data points and are mostly unaffected by the different colluding attackers across the three scenarios, validating the effectiveness of this technique to defend against poisoning attacks in unsupervised learning settings.



(a) Scenario 1: All malicious clients perform an indiscriminate data poisoning attack

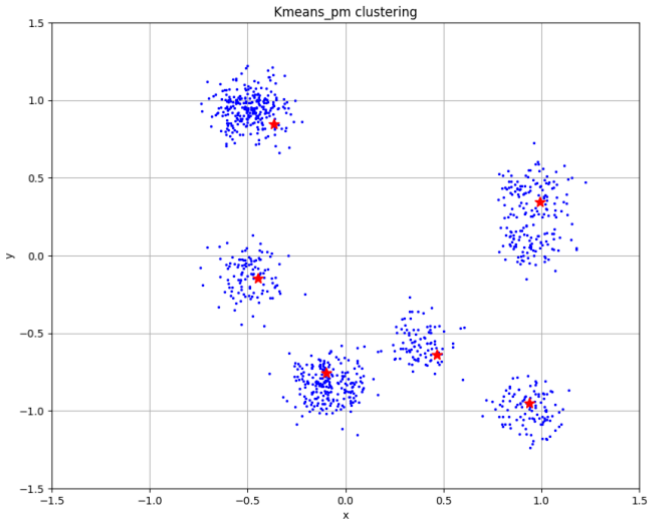


(b) Scenario 2: All malicious clients send random model updates as learnt centroids

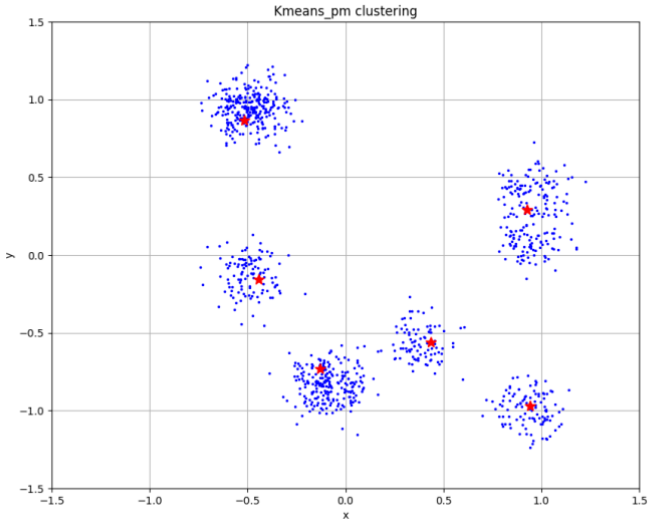


(c) Scenario 3: 50% of malicious clients perform an indiscriminate data poisoning attack and the remaining 50% of malicious participants send random model updates as centroids.

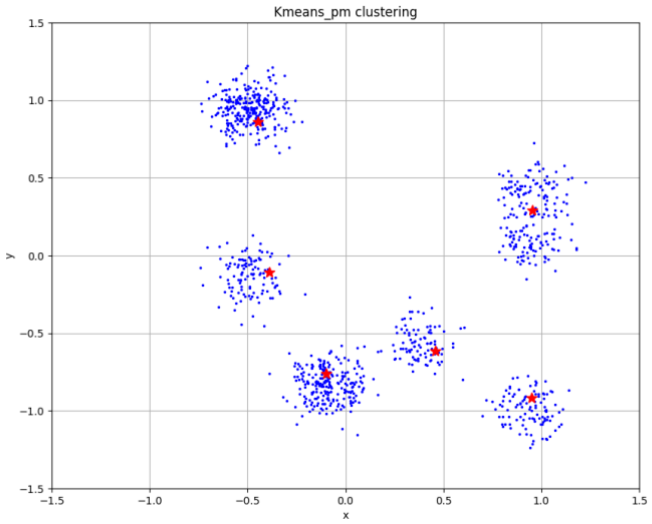
Figure 28: Learnt centroids for 2D-Synthetic dataset using standard (non-robust) K-Means clustering algorithm



(a) Scenario 1: All malicious clients perform an indiscriminate data poisoning attack



(b) Scenario 2: All malicious clients send random model updates as learnt centroids



(c) Scenario 3: 50% of malicious clients perform an indiscriminate data poisoning attack and the remaining 50% of malicious participants send random model updates as centroids.

Figure 29: Learnt centroids for 2D-Synthetic dataset using proposed robust K-Means clustering algorithm

5 Robustness of the Data Pre-processors

5.1 Outlier Detection

In this section, we evaluate the performance of the outlier detection scheme implemented in deliverable D5.4 for the MMLL library. The outlier detection scheme is used as a pre-processing step before the start of the training and is applicable for all POMs and different learning algorithms and helps to mitigate data poisoning attacks. As shown in Table 1 and Table 2, outlier detection can be applied for all POMs in both supervised and unsupervised learning tasks, being a useful mitigation for those POMs where robust aggregation cannot be applied.

We consider the federating training setup of a logistic classifier model similar to the one described in deliverable D5.4. We use Pima dataset and consider POM 5 for our evaluation. This training setup considers a total of 5 participants for the training of the model. One of these participants has poisoned the local training dataset which can undermine the performance of the trained model when using non-robust aggregation schemes, like FA. We consider the federated training for 20 communication rounds and the training stops when the change in the model parameters is less than 0.01. For the outlier detector, we consider the

normal samples threshold to be 0.90 and we consider 20 nearest neighbours for a sample's outlier score estimation. Furthermore, for the poisoned data at the participant, we consider that sign of 50% of the training data at the participant has been flipped i.e., $x'_i = -x'_i$.

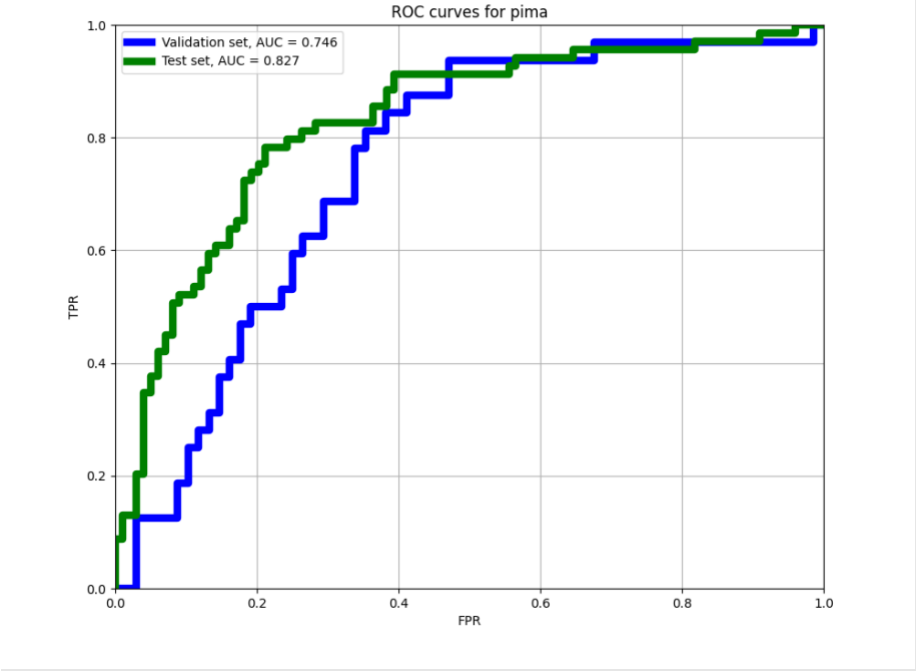


Figure 30: Logistic classifier performance when all participants are benign

Figure 30 shows the ROC curves when all participants are benign and proposed outlier detection scheme has not been used as a pre-processing step at the clients. It can be seen that AUC evaluated on the test set is 0.827. Figure 31 shows the performance of logistic classifier for the scenario when one of the participants has poisoned data. We observe that the performance of the trained classifier drops significantly to a smaller AUC of 0.738. This shows that standard implementations of algorithms in MMLL are very vulnerable and an attacker, who just crafts a very simple data poisoning attack that reverses the sign of only a proportion of training data at a participant, can decrease the accuracy of the federated learning model in the absence of any defensive strategy.

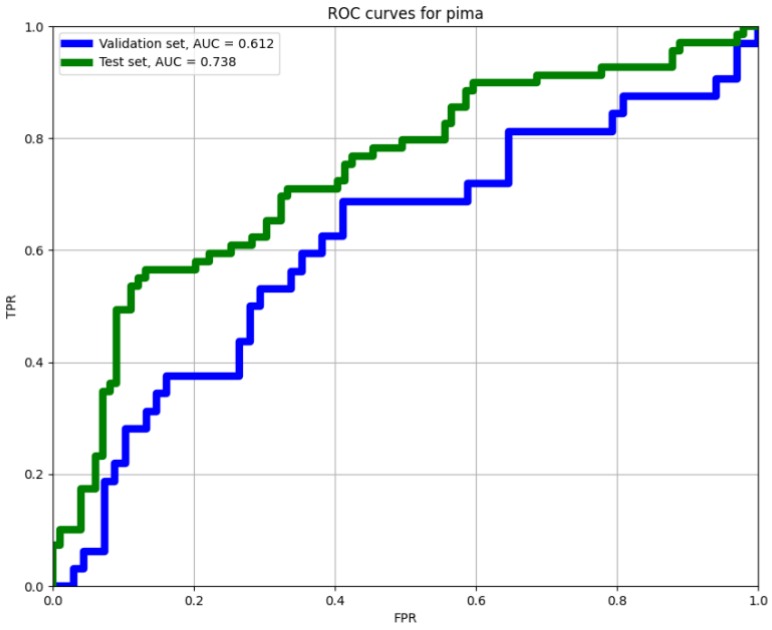


Figure 31: Logistic classifier performance when one participant has poisoned data

On the other hand, Figure 32 shows the performance of logistic classifier when we apply the proposed outlier detector as a pre-processor on all clients' training data to filter-out the poisoning points. We observe that the outlier detector can successfully minimize the influence of poison data on trained model and the ROC curve for the resulting model is very similar to the case when all the participants are benign with AUC 0.817 as shown in Figure 32. The small drop in performance is due to the fact that the scenario with poisoned datasets, has smaller number of effective (benign) training data points compared to the case where there is no attack.

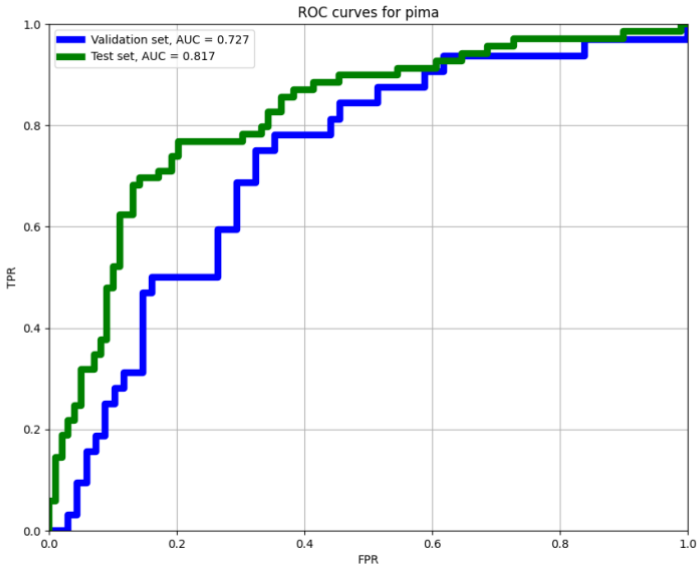


Figure 32: Logistic classifier performance when one participant has poisoned data and proposed outlier detector is used as a pre-processor

Finally, we also consider the federated training setup of logistic classifier when we use the proposed outlier detector as a pre-processor at the clients, even when all participating clients are benign. This is to analyse the effect of outlier detector as a data pre-filtering step at the clients to remove outliers (low quality data points that are not necessarily malicious) in the training datasets of the participants. The result in Figure 33 shows an improved AUC of 0.859 compared to 0.827 observed in Figure 30 when all participants are benign during the training, but the proposed outlier detector has not been used as a pre-processor step. This shows that even in the absence of any adversary, outlier detection can be used to filter out low quality training data at the clients and can improve the performance of the trained model.

The effect of outlier detection is the same regardless of the POM used, as it is applied before the training of the federated model starts. Thus, although in the experiments in this section we focused on POM 5, the results would be equivalent across the different POMs when using the same machine learning algorithm and settings.

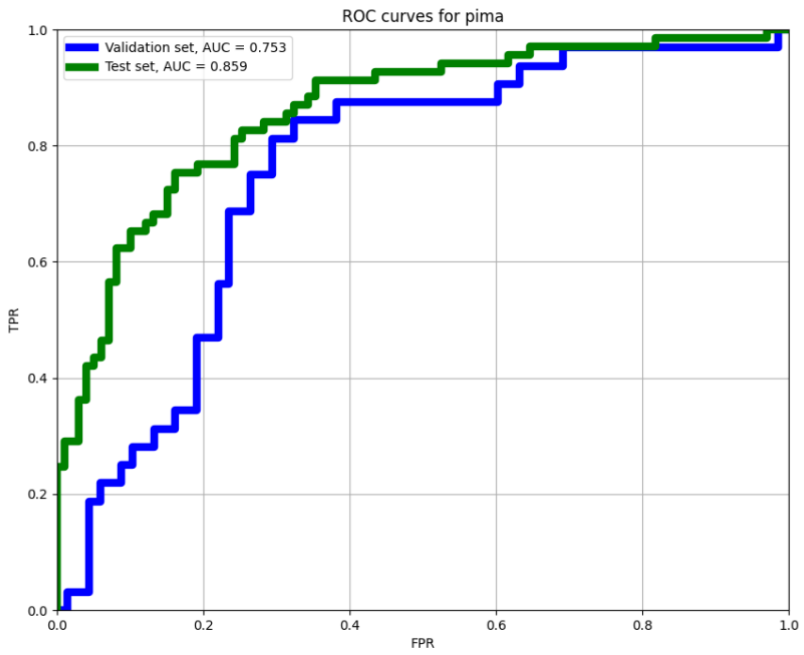


Figure 33: Logistic classifier performance when all participants are benign and proposed outlier detector is used as a pre-processor

5.2 Label Sanitisation

In supervised learning, we can mitigate the effect of label flipping attacks by using the data sanitisation strategy proposed in [Paudice et al. 2018] and described in deliverable D5.4. As shown in Table 1, label sanitisation can be applied for all supervised learning tasks across the different POMs in the MMLL library and a viable alternative for those POMs where robust aggregation cannot be applied.

In the experiments for this section, we consider the federating training of a logistic classifier (LC) model. We use Pima dataset and consider POM 6 for our evaluation. The total number of participants is five. One of these participants is a label-flipping attacker, which can undermine the performance of the trained model when using non-robust aggregation schemes, like FA. We consider the federated training for 20 communication rounds. We stop the training when the change in the model parameters is less than 0.01. For the label sanitisation, we consider the normal samples threshold to be 0.90. For the KNN Classifier, we consider 10 nearest neighbours to estimate the sample's label. PIMA is a binary classification task, so the label flipping attacker simply flips the labels of the training data points.

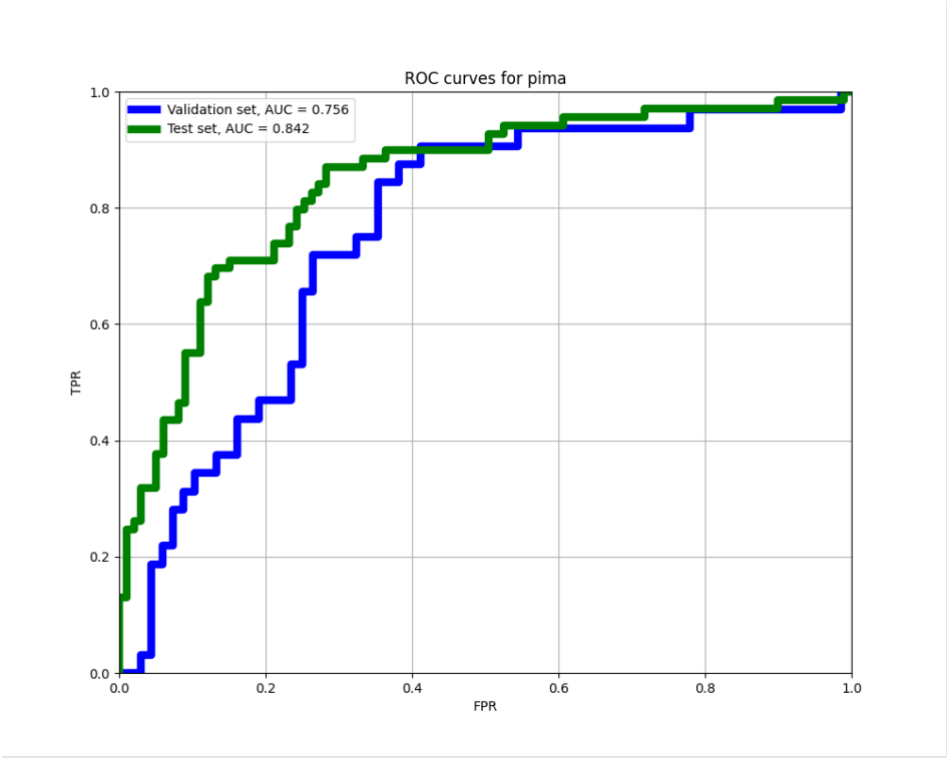


Figure 34: Logistic classifier performance when all participants are benign

Figure 34: Logistic classifier performance when all participants are benign shows the ROC curves when all participants are benign without any data pre-processing. AUC on the test set is 0.842. Figure 35: Logistic classifier performance when one participant is label flipping attacker shows the performance of the logistic classifier for the scenario when one of the participants is a label-flipping attacker (participant #3). We observe that AUC on the test set drops to 0.188. Predictably, standard algorithms in the MMLL library are extremely vulnerable to even a single malicious participant. Figure 36 shows the performance of the logistic classifier against one label-flipping attacker when we apply the proposed label sanitisation technique as a pre-processor on all clients' training data to flip the most suspicious labels in their local training data to the closest class in the provided curated dataset. In this case, AUC on the test set is 0.787, which is comparable to the training logistic classifier in federated settings with all benign participants. Overall, we observe that label sanitisation successfully minimises the influence of label flipping attackers. The final ROC curve for the trained model is similar to that of the tested model when all the participants are benign.

The effect of label sanitisation is the same regardless of the POM used, as it is applied before the federated learning starts. Thus, the results in this section are generalisable to different POMs when using the same machine learning algorithm and settings.

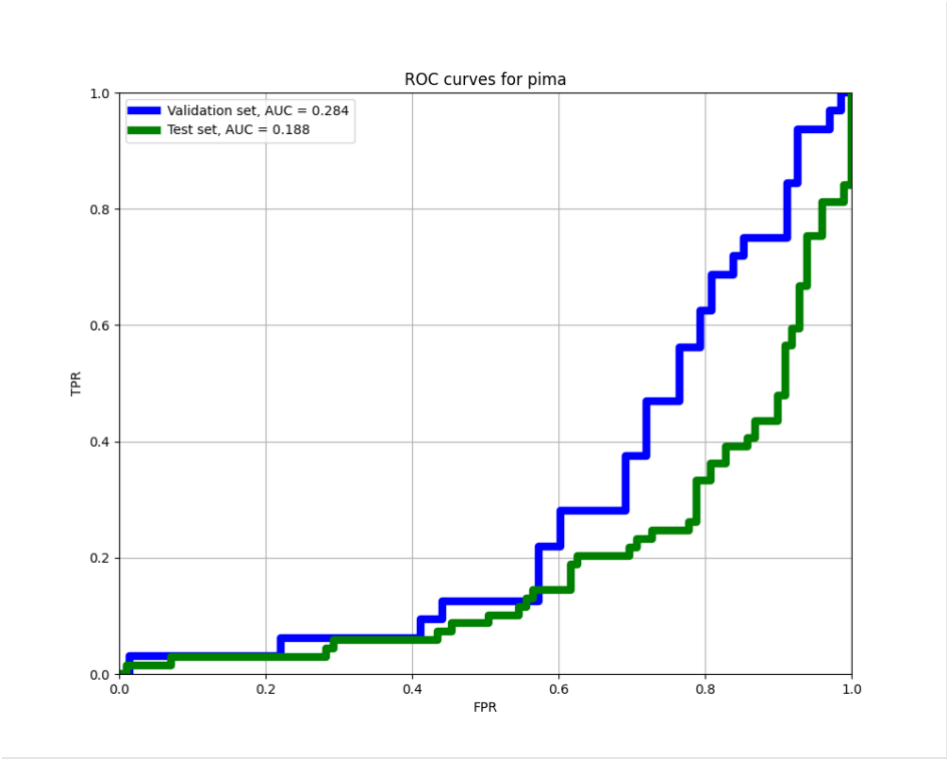


Figure 35: Logistic classifier performance when one participant is label flipping attacker

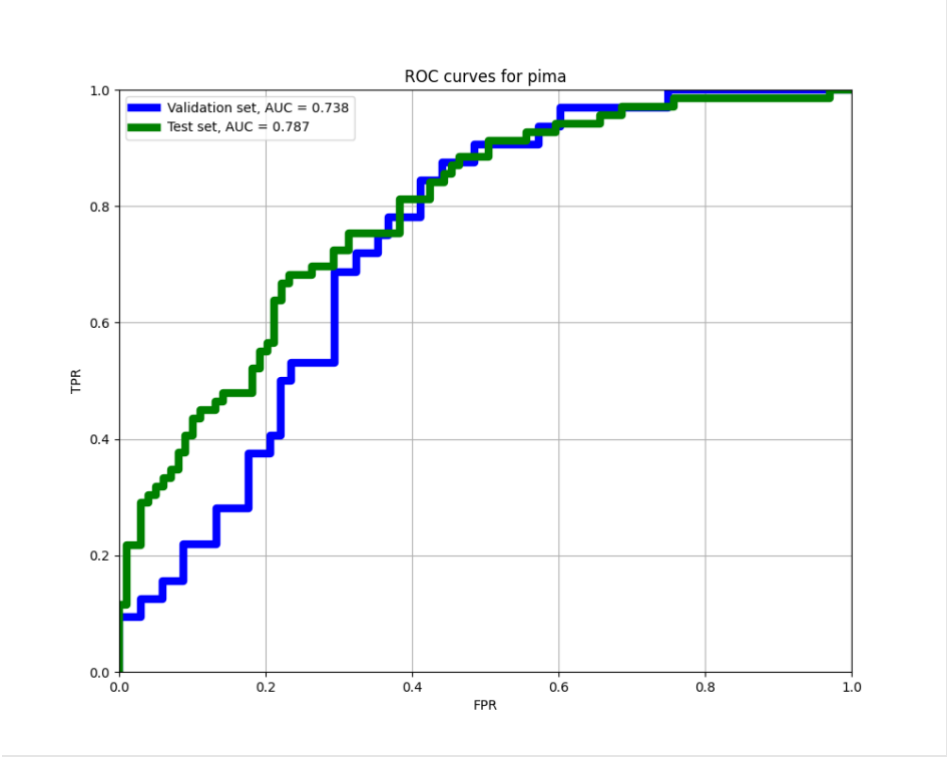


Figure 36: Logistic classifier performance when one participant is label flipping attacker and proposed label sanitisation is used as a pre-processor

6 Robustness of Supervised Learning Algorithms against Evasion Attacks

For this evaluation we test the MMLL library against evasion attacks when defensive steps are taken by the clients. This builds on the work presented in deliverables D5.3 and D5.5 which evaluated POM 1 on an older version of the library. Here we extend the evaluation to also include POM 2 and POM 3. For the evaluation of the attacks and defences we use v2.2.0 of the MMLL library.

The structure of this evaluation is as follows: we give relevant background into evasion attacks in 6.1, introduce the attack we use in Section 6.2, the defence employed in Section 6.3, motivations for the POMs selected in 6.4 and finally show the results in Section 6.5 and 6.6.

6.1 Evasion Attacks

Evasion attacks are a well-known phenomenon in machine learning. In the classical setup, an attacker begins with a datapoint x and optimises to find a perturbation δ such that a classifier will output different predictions for $f(x)$ compared to $f(x+\delta)$. The perturbation δ is constrained to be small, in the image domain this is typically done using a L_p norm. The difficulty from a defender's perspective is that δ can be so small such that it is challenging to detect, and furthermore adaptive attackers can tailor their optimisation to evade many proposed defences [Athalye et al, 2018]. An example of how small the perturbations can be, and yet have a significant impact, is shown in Figure 37: A adversarial perturbation is added, such that the image on the right is indistinguishable to the human eye compared to the original, and yet changes the prediction of the neural network.

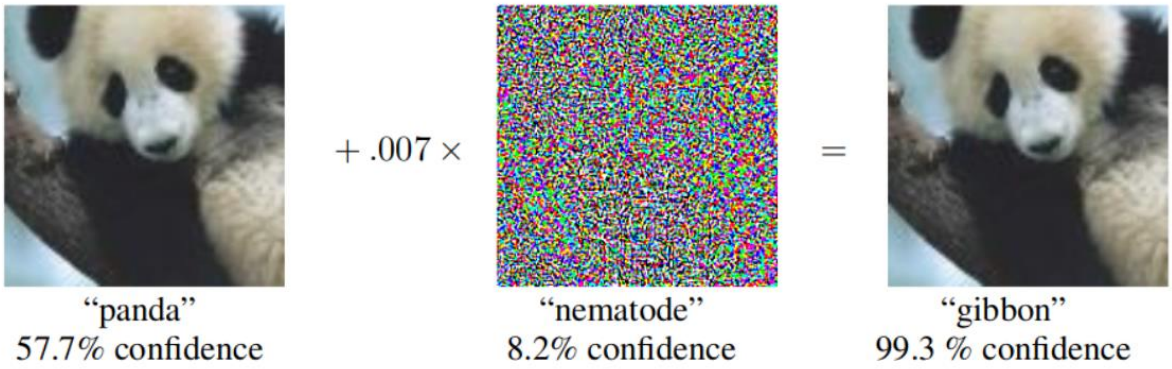


Figure 37: A adversarial perturbation is added, such that the image on the right is indistinguishable to the human eye compared to the original, and yet changes the prediction of the neural network.

6.2 Summary of the Attacks used for the Assessment

Test time evasion attacks against FL trained models function in an equivalent manner to modes trained in a centralised setting. There are therefore many different attacks developed by the community to simple one step methods [Goodfellow et al. 2014] to complex optimisation schemes [Carlini, N. and Wagner, D. 2017]. Here we evaluate the models using Projected Gradient Descent (PGD) as it is one of the most widely used benchmark attacks [Madry et al. 2018].

The parameters we use for the attack are a L_∞ bound of 0.3, with 40 gradient iterations each using a step size of 0.01. We use the Adversarial Robustness Toolbox to run the attacks [Nicolae et al, 2018] providing a well-tested and opensource implementation of the PGD attack.

6.3 Summary of Defences Available

In a similar manner to the wide range of attacks there are many defences to choose from. We evaluate the models when they conduct adversarial training [Madry et al. 2018], as it is a well-established defensive method which offers strong performance and is a widely used benchmark for both attacks and defences. Given (x, y) data label pairs the defence aims to find a set of parameters θ to solve the following optimisation problem:

$$\min_{\theta} \rho(\theta), \text{ where } \rho(\theta) = \mathbb{E}_{(x,y)\sim D}\left\{\max_{\delta \in \mathcal{S}} \mathcal{L}(x+\delta, y; \theta)\right\}.$$

where $\mathcal{L}$ is the loss function of the neural network, and δ is the adversarial perturbation. The adversarial perturbation is limited to a L_p ball to which the defender will try and protect up to. For the MNIST dataset we examine $L_\infty = 0.3$ which is a commonly used bound. We additionally use random starting of x while training where each datapoint is projected to a L_∞ ball of maximum 0.3 when the adversarial example construction process begins to avoid overfitting.

6.4 Applicability of the Defences for each POM

We consider neural network-based architectures as being applicable for this assessment. Adversarial examples can be made for other machine learning models, however state-of-the-art defences against adversarial examples have been designed and developed for protecting neural networks. Hence for this section POMs 1 – 3 will be in theory applicable. However, in practice only POM 1 is of practical relevance due to the computational overheads incurred with the encryption schemes employed in POM 2 and 3. More precisely, adversarial training requires large neural networks to handle the difficulty of the underlying task. However, for a network with just 8,000 parameters the extra time overhead is between 5 – 10 min for POM 2 and POM 3 per round. When scaling to neural networks of realistic size with millions of parameters [Madry et al. 2018] the time overhead runs into 12+ hours making it impractical for use.

For this assessment we are considering neural-network-based models and so do not adversarially train models in POMs 4 - 6. This is because to be effective classical adversarial training requires models with large capacities following results in prior literature works [Madry et al. 2018]. Therefore, shallow models such as logistic regression are ineffective at handling adversarial training on complex data like images. Hence, without adversarial training we did not run adversarial evaluations as it will not show anything which is not already well established by the machine learning community: that ML models are vulnerable to evasion attacks. Without defences, regardless of centralised or federated training, the resulting models are known to be vulnerable to adversarial examples e.g. SVMs [Papernot et al, 2016] or regression [Mode, G.R, and Khaza, A.H 2020] [Zizzo et al, 2020].

6.5 Assessment of POM 1

For POM 1 we use the MNIST neural network in [Madry et al. 2018] and we run MMLL for 80 communication rounds with 3 clients, each of which has a random partition of the MNIST dataset.

The final performance of the model when trained adversarially is 98.24% on normal data and 93.22% on adversarial examples. Conversely, the performance when trained normally without adversarial training is 0.01% for adversarial examples and 98.78% on normal data.

The security curve showing the relationship between the epsilon budget and the accuracy of the model is shown in Figure 38.

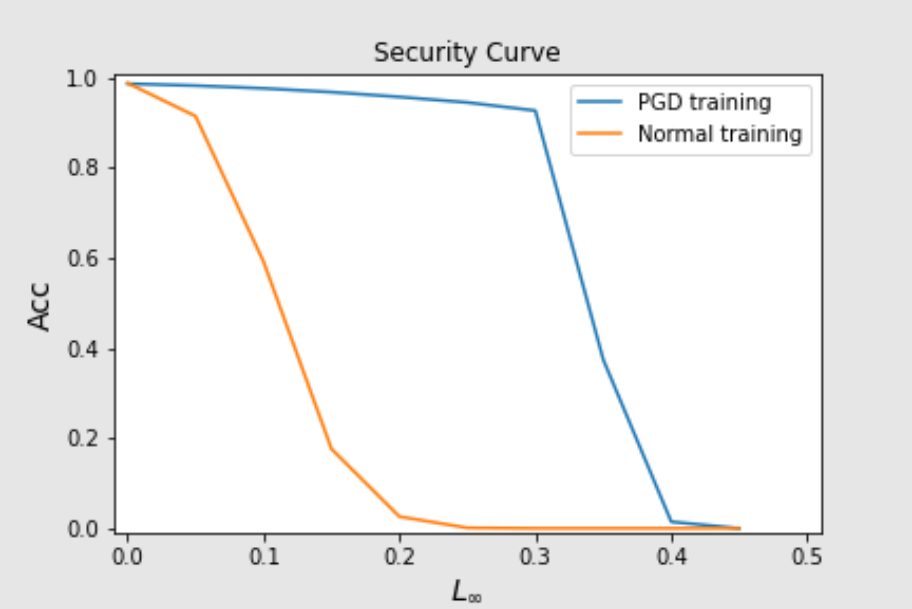


Figure 38: Performance of the models with varying attack budget

6.6 Assessment of POMs 2 and 3

Here, we used a much smaller neural network comprising of 2 convolutional layers and a final dense layer with slightly less than 8,000 parameters. Due to the high computational costs with running the encryption algorithms we run MMLL for 20 communication rounds with 3 clients, each of which has a random partition of the MNIST dataset.

With a network of such a small size adversarial training predictably fails. The neural network does not learn the underlying task and outputs the prediction “1” for essentially all inputs and its accuracy on both adversarial and normal data is ~10%. This occurred on both POM 2 and 3. Note that this is not due to federated learning: centralised adversarial training still results in a model outputting the same class for all inputs.

If for POM 2 using FL we train the model normally (i.e., non-adversarially) then it functions well on normal data with 97.21% accuracy, however as it is undefended against evasion attacks its performance on adversarial data is 0.0%. Likewise, for POM 3 the normal data

performance is 97.94% and the adversarial performance is 0.0%. The corresponding security curve is shown in Figure 39.

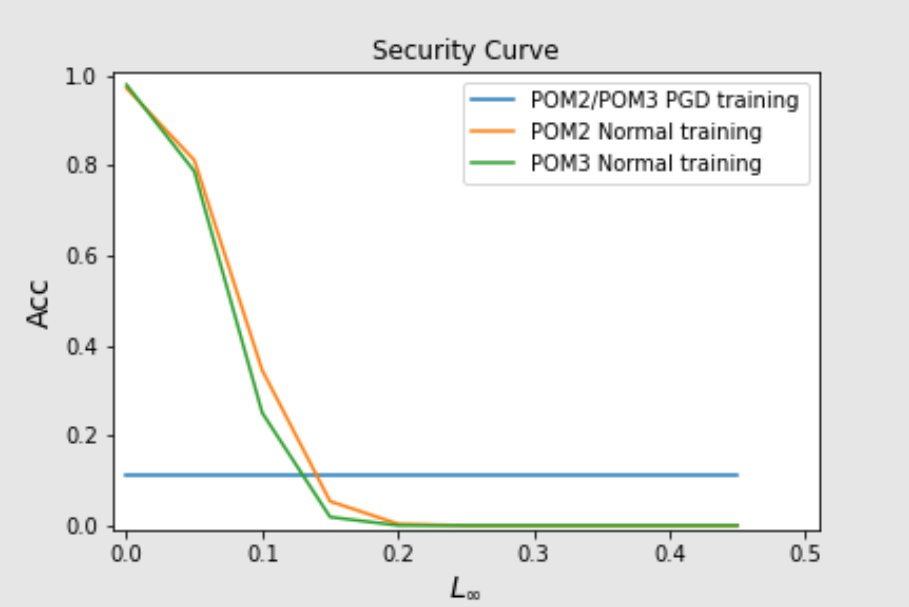


Figure 39: Performance of the smaller model used for POM2/3. As when PGD training is attempted the model essentially always outputs the same prediction the accuracy for the PGD model is constant.

7 Conclusion

In this deliverable we have presented a comprehensive evaluation of the robustness of the algorithms developed and implemented in the MMLL library for MUSKETEER. We have considered scenarios with poisoning attacks (at training time) and evasion attacks (at test time), both in supervised and unsupervised learning settings.

As we already shown in previous deliverables in WP5 (see for example D5.2 and D5.3), standard federated learning algorithms can be very brittle in the presence of an adversary, who can compromise the performance of the resulting model both at training and at test time. On the other side, our evaluation has shown that the defensive techniques developed and implemented in the library are effective to mitigate these threats, achieving the KPIs described in the consortium agreement and offering significantly improved robustness (e.g., being robust to attack scenarios with 20% of malicious participants). The defences implemented in MUSKETEER have also been put to the test in the 2nd Hackathon, which was entirely designed around the participants attempting to break the defences.

Our assessment has comprehensively included the analysis of the robustness of the federated algorithms across the different POMs implemented in the platform. Two main types of

defences have been implemented and evaluated: defences based on data pre-filtering and defences based on robust data aggregation. Defences based on pre-filtering can be applied across all the POMs and in all the modes and help protect the model from outsider attacks. Defences based on robust data aggregation help protect the learning against insider attacks, even when the malicious users collude but not if the malicious users are too numerous. Our aim was to achieve robustness for 20% malicious users. The results above demonstrate that we achieve this and exceed it, in some cases by a significant margin. In some cases, our defences go beyond simply mitigating the attacks as they also enable to identify the malicious users. However, defences based on data aggregation can only be used when the aggregator can perform the operations required. This is not the case when the aggregator operates on encrypted data or has to rely solely on the operations supported by homomorphic encryption. This is also the case for adversarial training. In this sense, there is a trade-off between privacy and robustness to attacks. Restricting the aggregator's access to the updates also restricts its ability to defend against malicious clients. Users of the MUSKETEER platform are therefore advised to balance carefully data privacy risks and model integrity risks in their given context of application.

Our assessment justifies the need of having mechanism to defend against training and test-time attacks and endorses the usefulness and the efficacy of the techniques implemented in the MMLL library to mitigate such attacks.

8 References

[Arthur & Vassilvitskii 2007] Arthur, D., Vassilvitskii, S. "*K-means++: the advantages of careful seeding.*" In Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms (SODA '07). Society for Industrial and Applied Mathematics, USA, 1027–1035.

[Athalye et al, 2018] Athalye, Anish, Nicholas Carlini, and David Wagner. "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples." International conference on machine learning. PMLR, 2018.

[Bhagoji et al. 2019] Bhagoji, Arjun Nitin, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. "*Analyzing Federated Learning through an Adversarial Lens.*" In International Conference on Machine Learning, pp. 634-643. PMLR, 2019.

[Biggio et al. 2013] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, Fabio Roli, "*Evasion Attacks against Machine Learning at Test Time.*" European Machine Learning and Data Mining Conference (ECML/PKDD), 2013.

[Blanchard et al. 2017] Blanchard, Peva, Rachid Guerraoui, and Julien Stainer. *"Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent."* In Advances in Neural Information Processing Systems, pp. 119-129. 2017.

[Carlini, N. and Wagner, D. 2017] Carlini, Nicholas, and David Wagner. "Towards evaluating the robustness of neural networks." 2017 IEEE Symposium on Security and Privacy (S&P). IEEE, 2017.

[Goodfellow et al. 2014] Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples." arXiv preprint arXiv:1412.6572, 2014.

[LeCun et al. 2010] LeCun Y, Cortes C, Burges CJ. "*MNIST handwritten digit database.*" URL: http://yann. lecun. com/exdb/mnist, 2010.

[Madry et al. 2018] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, Adrian Vladu, "*Towards Deep Learning Models Resistant to Adversarial Attacks*." International Conference on Representation Learning (ICLR), 2018.

[McMahan et al. 2017] "*Communication-efficient learning of deep networks from decentralized data.*" Artificial intelligence and statistics. PMLR, 2017.

[Mode, G.R, and Khaza, A.H 2020] Mode, Gautam Raj, and Khaza Anuarul Hoque. "Adversarial examples in deep learning for multivariate time series regression." 2020 IEEE Applied Imagery Pattern Recognition Workshop (AIPR). IEEE, 2020.

[Muñoz-González et al. 2019] Luis Muñoz-González, Kenneth T. Co, and Emil C. Lupu. *"Byzantine-Robust Federated Machine Learning through Adaptive Model Averaging."* arXiv preprint arXiv:1909.05125, 2019.

[Nicolae et al, 2018] Nicolae, Maria-Irina, et al. "Adversarial Robustness Toolbox v1. 0.0." arXiv preprint arXiv:1807.01069, 2018.

[Papernot et al, 2016] Papernot, Nicolas, Patrick McDaniel, and Ian Goodfellow. "Transferability in machine learning: from phenomena to black-box attacks using adversarial samples." arXiv preprint arXiv:1605.07277, 2016.

[Szegedy et al. 2013] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, Rob Fergus. "*Intriguing properties of neural networks*." arXiv pre-print arXiv:1312.6199, 2013.

[Xiao et al. 2017] Xiao, Han, Kashif Rasul, and Roland Vollgraf. *"Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms."* arXiv preprint arXiv:1708.07747, 2017.

[Yin et al. 2018] Yin, D., Chen, Y., Ramchandran, K., and Bartlett, P. "*Byzantine-Robust Distributed Learning: Towards Optimal Statistical Rates*." in International Conference on Machine Learning (ICML), pp. 5636-4545, 2018.

[Zizzo et al, 2020] Zizzo, Giulio, et al. "Adversarial Attacks on Time-Series Intrusion Detection for Industrial Control Systems." 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom). IEEE, 2020.