ECAI 2023
K. Gal et al. (Eds.)

doi:10.3233/FAIA230257

Robust Federated Learning Method Against Data and Model Poisoning Attacks with Heterogeneous Data Distribution

Ebtisaam Alharbi[a,b;*], Leandro Soriano Marcolino[a], Antonios Gouglidis[a] and Qiang Ni[a]

[a]School of Computing and Communications, Lancaster University, United Kingdom
[b]Department of Computer Science, Umm Al-Qura University, Saudi Arabia
ORCiD ID: Ebtisaam Alharbi https://orcid.org/0000-0002-3969-3209, Leandro Soriano Marcolino https://orcid.org/0000-0002-3337-8611, Antonios Gouglidis https://orcid.org/0000-0002-4702-3942, Qiang Ni https://orcid.org/0000-0002-4593-1656

Abstract. Federated Learning (FL) is essential for building global models across distributed environments. However, it is significantly vulnerable to data and model poisoning attacks that can critically compromise the accuracy and reliability of the global model. These vulnerabilities become more pronounced in heterogeneous environments, where clients' data distributions vary broadly, creating a challenging setting for maintaining model integrity. Furthermore, malicious attacks can exploit this heterogeneity, manipulating the learning process to degrade the model or even induce it to learn incorrect patterns. In response to these challenges, we introduce RFCL, a novel Robust Federated aggregation method that leverages CLustering and cosine similarity to select similar cluster models, effectively defending against data and model poisoning attacks even amidst high data heterogeneity. Our experiments assess RFCL's performance against various attacker numbers and Non-IID degrees. The findings reveal that RFCL outperforms existing robust aggregation methods and demonstrates the capability to defend against multiple attack types.

1 Introduction

Federated learning (FL) [15] is a recent collaborative machine learning framework trained by widely distributed clients. In FL, clients train model updates based on local training data and the updated global model and then send these updates to the server. The server aggregates them to create a new global model, which is then sent back to the clients for the next training round. Since the training is distributed across several clients and conducted in parallel, FL provides efficiency and scalability [4]. FL allows for sharing learning models while preserving the privacy of the client's data [10].

Although FL can aggregate heterogeneous data across many clients to train a global model, it is a vulnerable structure. Data processing and local training procedures under client control may expose the global aggregate model to attacks. FL is vulnerable to malicious clients; simply one adversarial client may compromise the entire performance of the global model [2]. Specifically, untargeted poisoning attacks, like random noise [3] or sign-flipping attacks [2], aims to push the global model in the wrong direction from the outset of rounds. The result is a consistently high rate of test errors across all test sets, illustrating the damaging implications of the initial deviation in the model's learning direction.

Several robust FL aggregation methods are proposed in the literature [3, 8, 23, 7]. However, recent studies have revealed that some of the robust FL aggregation methods are susceptible to new attacks. For instance, A Little Is Enough (ALIE) attack can exploit the empirical variance between client updates to bypass Median [23] and Krum [3], provided that the variance is high enough [1]. Similarly, the Inner Product Manipulation (IPM) attack can significantly threaten Median [23] and Krum [3] by manipulating the inner product between the true gradient and the robust aggregated gradients to be negative [21]. Non-IID data can impact the robustness of the FL, particularly in the presence of adversarial attacks that exploit data heterogeneity. A small percentage of adversaries may be sufficient to launch a successful attack, making it even more critical to address Non-IID data in the context of potential attacks [1].

Existing defence methods try to distinguish between malicious and benign clients by analyzing the statistical differences in their model updates. However, these detection approaches demand many model updates to make reliable decisions. Consequently, malicious clients might have already poisoned the global model before being identified, reducing the efficacy of these defence strategies [23, 8].

Current defences against poisoning attacks in FL are developed to prevent the global model from being compromised by a small number of malicious clients. Even when trained with malicious clients, they ensure that the global model remains close to the one that would have been learned without them. Moreover, some aggregation methods such as FLTrust [6], LearnDefend [14], and Zeno++ [22] require the server to access part of or all the private data. This assumption contradicts the FL framework's zero server knowledge and privacy-preserving principle.

The server implementing robust aggregator methods faces difficulty distinguishing between benign and adversarial clients. This challenge is aggravated by high-dimensional gradients, a higher proportion of attackers, and significant heterogeneity (Non-IID).

We present RFCL, a novel Robust Federated aggregation CLustering technique to address security issues arising from data and model poisoning attacks in heterogeneous data settings. The RFCL frame-

[*] Corresponding Author. Emails: {e.alharbi, l.marcolino, a.gouglidis, q.ni}@lancaster.ac.uk

work employs clustering to group models from participating clients and establishes cluster centres. Subsequently, it identifies the most similar clusters by evaluating the cosine similarity between their respective cluster centres, ensuring the selection of high-quality models. A meta-learning phase is conducted, consolidating the models of clients within each chosen cluster to produce a global model tailored to the particular cluster of clients. To enhance security and mitigate the impact of adversarial clients on the final model, RFCL incorporates a personalization process. This process selectively sends updated models to clients associated with similar clusters, thereby bolstering security measures. We assess RFCL's performance across various attack scenarios considering varying numbers of attackers and non-IID data distributions. In summary, our contributions are three-fold: **(i)** We develop and implement RFCL, an innovative robust aggregation method that leverages clustering and cosine similarity to group the most similar clusters, providing a defence against data and model poisoning attacks in FL. **(ii)** We perform a comprehensive evaluation of RFCL's effectiveness across various attack scenarios, such as IPM [21], ALIE [1], sign-flipping, random noise, and label-flipping, considering different numbers of attackers and Non-IID data distributions. **(iii)** We compare RFCL's performance against recent robust methods [3, 23, 17, 9, 11] using several datasets (MNIST, CIFAR-10, and Fashion-MNIST). Our comprehensive experiments and analyses prove RFCL's superior performance in FL. RFCL exhibits remarkable efficacy in handling a large number of malicious clients and adapting to significant heterogeneity.

2 Related Works

Several defence strategies have been proposed in FL to mitigate the impact of poisoning attacks, with a particular emphasis on robustness aggregation methods. These methods aim to enhance the FL system's resilience against adversarial manipulation of data and models.

Krum is a robust aggregation method that employs distance-based outlier detection. It determines the similarity between client updates by calculating Euclidean distances. The clients are then sorted based on the total distances to their nearest clients. The client with the lowest average distance is selected to update the global model. However, in Non-IID distributions, selecting updates from only one client raises concerns regarding privacy and performance [3].

Trimmed mean is an aggregation method similar to FedAvg, which focuses on providing robustness against outliers by removing a proportion of the lowest and highest values per coordinate before calculating the mean [23]. Median-based algorithms, such as coordinate-wise median [23] and geometric median [18], are more resistant to outliers than mean-based algorithms, making them viable alternatives for FL aggregation. Although the trimmed mean and median exclude extreme values, it remains sensitive to outliers within the trimmed range. Furthermore, these aggregators lack flexibility and treat all client updates equally, disregarding differences in data quality and training conditions. This limitation can lead to suboptimal aggregation results, particularly in heterogeneous environments. Additionally, they do not support personalization by disregarding client-specific characteristics and assigning uniform weights, thus limiting their ability to adapt to individual clients' contributions.

Bulyan handled the issue in Krum of selecting one client to be a global model, which combines Krum and a trimmed mean [8]. Bulyan first iteratively selects local models using Krum, then aggregates the local models utilizing a form of the trimmed mean. However, the effectiveness of Bulyan heavily depends on the assumption that the majority of clients are benign and only a small fraction are Byzantine. Additionally, Bulyan's performance may degrade in scenarios with high levels of heterogeneity among clients.

Adaptive Federated Averaging (AFA) detects Byzantine clients. AFA utilizes the cosine similarity between client model updates and the global aggregated model to detect potentially malicious behaviour. It calculates the median, mean, and standard deviation for each model. Then the clients whose similarity falls outside a predetermined threshold are considered to have provided arbitrary updates and are subsequently blocked [17]. However, while AFA effectively blocks outlier malicious clients in IID settings, it may inadvertently block benign clients under Non-IID conditions due to the inherent data distribution differences.

FedMGDA+ is a robust aggregating method. It assigns a score to each client based on the similarity of their models, using a single-layer Neural Network (NN) to minimize the loss. The input to the NN is the difference between the client and global models, and the updated weights are returned and utilized for aggregation. The central server then seeks to learn the relevance of each client by considering their assigned significance scores, which are initially determined by the size of the client's dataset. Malicious clients are deemed insignificant and are only blocked once they surpass a specific threshold [9].

The Centered Clipping (CC) is an aggregation method that aims to improve the robustness of FL against malicious and Byzantine updates. The method operates by iteratively refining model updates by clipping them around a centre and updating it accordingly. By focusing on updates closer to the centre, CC mitigates the impact of potential outliers. The CC method's robustness has been proven effective when the variance of updates is bounded, and the proportion of malicious clients is ≤ 0.15 [11]. It relies on a fixed clipping threshold to limit the impact of extreme updates from malicious clients. However, this fixed threshold may not be optimal for all scenarios, as it fails to adapt to varying levels of adversarial behaviour or changing data distributions. Consequently, the CC method may not effectively handle sophisticated attacks or situations with significant data heterogeneity.

Group-Wise Robust Aggregation [24] is an alternative approach that utilizes clustering techniques to safeguard against untargeted attacks. Differing from this method, our approach integrates meta-learning principles and personalization by assembling clients with similar data distributions via clustering techniques. We then tailor the global model to these specific clusters and update the global model based on the most representative clusters, offering a more customized and secure aggregation strategy in federated learning environments.

3 Methodology

RFCL's method utilizes multi-centre, meta-learning, cosine similarity, and selective personalization to improve the robustness of the federated learning process against data and model poisoning attacks.

Problem Formulation. We consider a typical FL setting where multiple clients collaboratively train a model maintained on a central server [15]. At the beginning of the round, the server provides all clients with the initial global model parameter $\mathbf{M_0}$. Then, at round r, each client i gets the global model parameter $\mathbf{M_{r-1}}$ from the server, configures its local model parameters $\mathbf{M^i_{r-1}} = \mathbf{M_{r-1}}$, and performs local updates. Then, each client i sends its local model update $\triangle\mathbf{M^i_t} = \mathbf{M^i_r} - \mathbf{M_{r-1}}$ to the server. The server aggregates local model updates to provide a new global model for the next round $\mathbf{M_r} = \mathbf{M_{r-1}} + \sum_{\mathbf{i=1}}^{\mathbf{N}} \mathbf{p_i}\triangle\mathbf{M^i_r}$, where p_i is the aggregation weight of client i and $\triangle\mathbf{M^i_r}$ is the gradient of client i at round r.

We make the following assumptions concerning attacker capabilities, schemes, and objectives. First, the attacker controls a group of

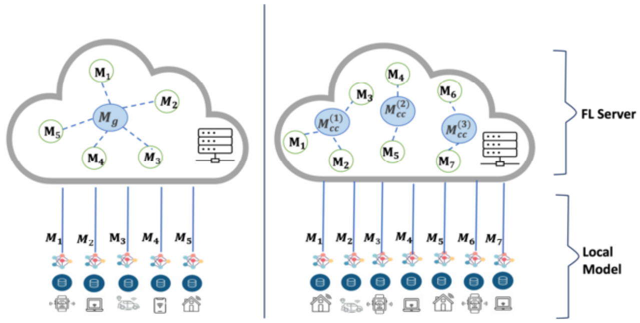


Figure 1: Comparison of typical FL single-centre aggregation (left) versus RFCL multi-centre internal aggregation (right).

clients. Second, the attacker can modify model updates before sending them to the central server. Third, the attacker cannot compromise the central server or influence other benign clients. Finally, the attacker is unaware of the aggregation method used by the server.

RFCL Process. The key steps and high-level view of the RFCL process, as outlined in Algorithm 1, are as follows: At the beginning of each round, the server checks if it is the first round ($r = 0$). If so, the server shares the initial model ($\mathbf{M_0}$) with all clients (N_i), and the clients perform local training. For subsequent rounds, the server shares the corresponding cluster centres models ($\mathbf{M_{cc}}$) with the clients associated with each cluster, and then the clients perform local training. After local training, the server collects all models from the clients ($\mathbf{M_i}$). The server then performs PCA and HDBSCAN clustering on the collected client models ($\mathbf{M_i}$) to generate cluster centres models ($\mathbf{M_{cc}}$). Based on cosine similarity, the server selects the most similar models among cluster centres models ($\mathbf{M}_{best}$). The server computes a concentrated model ($\mathbf{M_c}$) by aggregating the selected most similar models ($\mathbf{M_{best}}$). The cluster centres ($\mathbf{M_{cc}}$) of the selected most similar models ($\mathbf{M_{best}}$) are updated with the concentrated model ($\mathbf{M_c}$), and it is assigned $\mathbf{M_c}$ as a global model $\mathbf{M_g}$. Finally, the server tests the updated global model ($\mathbf{M_g}$) on the test dataset ($\mathbf{D_{test}}$) and records the error for the current round ($E[r]$).

Algorithm 1 RFCL Process

```
Require: $M_0, N, D_{test}$ return $E$ , $M_g$
for $r = 0$ to $R - 1$ do
    if $r == 0$ then
        Share $M_0$ with all $N_i$ and perform local training
    else
        Share $M_{cc}$ with associated $N_i$ and perform local training
    end if
    $M_i \leftarrow$ Collect all models
    $M_{cc} \leftarrow$ Perform clustering on $M_i$                      ▷ See Algorithm 2
    $M_{best} \leftarrow$ Select the most similar models in $M_{cc}$     ▷ See Algorithm 3
    $M_c \leftarrow$ External Aggregate $M_{best}$
    $M_{cc} \leftarrow$ Update the cluster centres of $M_{best}$ with $M_c$   ▷ See Algorithm 4
    $M_g \leftarrow M_c$
    $E[r] \leftarrow$ Evaluate $M_g$ on $test(D_{test})$
end for
return $E$ , $M_g$
```

Clustering Method: This technique groups models with similar weight vectors. The Algorithm 2 starts by extracting the weights from each model in the input list. These weights represent the learned parameters of each model. The next step is to subject the weights to Principal Component Analysis (PCA). This dimensionality reduction technique identifies the principal components of the data and projects the data onto these components. This helps reduce computational costs and improve clustering performance by mitigating the curse of dimensionality. The reduced-dimension data is then processed using Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN), an advanced clustering technique that identifies clusters of varying densities and can detect outliers. This is particularly useful for handling outlier models significantly different from the rest. Unlike many other clustering algorithms, HDBSCAN does not require the user to specify the number of clusters beforehand and can discover clusters of varying densities [5]. After applying HDBSCAN, the algorithm counts the number of clusters and initializes the cluster centres and size. It then iterates through the models, assigns them to their respective clusters, and computes the cluster centres by the ModiAFA method, which will be explained below. The resulting cluster centres provide valuable insights into the similarities and differences between models and can be used for further analysis.

Algorithm 2 Clustering

```
Require: $models$ return $M_{cc}$
$X \leftarrow$ extract_weights(models)            ▷ Extract weights from models
$X \leftarrow$ PCA($X$)              ▷ Apply PCA to weights for dimensionality reduction
cluster $\leftarrow$ HDBSCAN($X$)                 ▷ Apply HDBSCAN clustering
$cluster\_labels \leftarrow$ cluster.labels         ▷ Retrieve the cluster labels
$cluster\_count \leftarrow \max(cluster\_labels) + 1$        ▷ Count clusters
$indices \leftarrow$ [[] for _ in range ($cluster\_count$)]
for $i, l \in$ enumerate($cluster\_labels$) do
    if $l \neq -1$ then
        $cluster\_centres\_len[l] += 1$            ▷ Increment size of clusters
        append $i$ to $indices[l]$      ▷ Append model index to respective cluster
    end if
end for
$cluster\_centres\_len /= $ len($N$)                 ▷ Normalization
for $i, ins \in$ enumerate($indices$) do
    $M_{cc}[i] \leftarrow$ ModiAFA($ins, models$)
end for
return $M_{cc}$
```

Multi-Centres Internal Aggregation Method. RFCL leverages multiple centres of models aggregation method to address the challenges deriving from the heterogeneity in FL. Local models are clustered into numbers of clusters. Each covers a subset of local models. Figure 1 depicts an intuitive comparison between typical FL and multi-centre FL. In typical FL, only one centre global model exists, as illustrated in the left Figure 1. The multi-centre FL depicted on the right, on the other hand, has three centres, $\mathbf{M_{cc}^{(1)}}$, $\mathbf{M_{cc}^{(2)}}$, and $\mathbf{M_{cc}^{(3)}}$, and each centre represents a cluster of clients with similar data distributions and models. The Modified AFA (ModiAFA) is an enhanced method of the original AFA [17]. Unlike the original method, ModiAFA does not block clients; instead, it adjusts their contributions based on the quality of their updates. It detects outlier models in each cluster and generates new cluster centres. The process is initiated by computing similarities ($\mathbf{s_i}$) between the previous $\mathbf{M_{cc}}$ and the updated model $\mathbf{M_i}$ provided by every client in each cluster. Following this, three statistical measures - mean ($\hat{\mu_s}$), median ($\bar{\mu_s}$), and standard deviation (σ_s) - are calculated for these similarities. Outlier models are then identified by comparing the mean and median of the similarities of $\mathbf{M_{cc}}$ and $\mathbf{M_i}$. Two conditions are checked to determine if a client's model is an outlier: if $\hat{\mu_s} < \bar{\mu_s}$ and $s_i < \bar{\mu_s} - \xi\sigma_s$, or if $\hat{\mu_s} \geq \bar{\mu_s}$ and $\mathbf{s_i} < \bar{\mu_s} + \xi\sigma_s$. Here, ξ is a parameter that can be adjusted to regulate the sensitivity of the outlier detection process. If a client's model meets either of these conditions, it is considered to have sent an outlier update. Instead of blocking the client, its weight is set to 0, excluding its model from contributing to the next round of model aggregation. This approach enables the effective detection and management of outliers without blocking clients outright.

Similarity Analysis Method: It identifies models' highest degree of similarity. The algorithm selects the top K most similar cluster centres models $\mathbf{M_{cc}}$, where K is determined by the smaller of either a predefined hyperparameter or the count of non-outlier clusters. This methodology assists in filtering out any outliers models that may have evaded the cluster centres' detection process. Rather than choosing all cluster centres models, the algorithm favours the most similar models. A comprehensive explanation of this similarity analysis procedure is provided in Algorithm 3. 1) Checking for outliers: This step counts the number of not outliers clusters (i.e., labelled as -1). $unoutliers = \{\sum_{i=1}^{n} I[cluster\_labels_i \neq -1] \quad \text{if} -1 \in cluster\_labels.$ Where I is the indicator function.2) Number of clusters to select: Determine the number of $\mathbf{M_{cc}}$ to select num_select by taking the minimum between K and $unoutliers$. 3) Calculating cosine similarity: Generate cluster centres weights, $\mathbf{X}$, and compute the cosine similarity for every pair of weights. The similarity is stored in $\mathbf{sims}$. 4) Selection of Most Similar Clusters: Find the indices of each model's num_select largest similarity scores. If the sum of these selected similarities surpasses the current highest total similarity score, denoted as $best\_val$, update $best\_val$ and $\mathbf{indices_{best}}$. 5) Similarity Score Normalization: Normalize the similarity scores within the selected clusters to obtain the probabilities p_s of selecting each model. This can be expressed as: $p_{s_i} = \frac{s_i}{\sum_{j \in \mathbf{indices_{best}}} s_j}$. Here, s_i is the similarity score of the i model, and $\sum_{j \in \mathbf{indices_{best}}} s_j$ represents the sum of the similarity scores for the models in the selected clusters. 6) Probability Reconfiguration: Adjust the probabilities based on the size of each cluster and normalize them again. This can be expressed as: $p_{s_i} = \frac{p_{s_i} \cdot \text{len}(cluster\_centre_i)}{\sum_{j \in indices_{best}} p_{s_j} \cdot \text{len}(cluster\_centre_j)}$. At the end of this process, the method returns the selected models ($\mathbf{M_{best}}$), their corresponding probabilities (p_s), and their indices ($\mathbf{indices_{best}}$). By doing so, the method ensures that models from larger clusters and those that are more similar to other models within their own cluster are given higher importance.

Algorithm 3 Similarity Analysis

```
Require: $M_{cc}$ , $K$ return $M_{best}, p_s, indices_{best}$
if $\nexists i : cluster\_labels_i = -1$ then                ▷ Checking for outliers
    $unoutliers \leftarrow \text{len}(cluster\_labels)$
else
    $unoutliers \leftarrow \sum_{i=1}^{n} I[cluster\_labels_i \neq -1]$
end if
$num\_select \leftarrow \min(K, unoutliers)$
$X \leftarrow \_generate\_weights(M_{cc})$
for each $m1$ in $X$ do                                   ▷ Calculating cosine similarity
    for each $m2$ in $X$ do
        $sim \leftarrow cos(m1, m2)$
        Append $sim$ to $sims$
    end for
end for
for each $s$ in $sims$ do                                 ▷ Selecting the most similar models
    $indices \leftarrow$ indices of the $num\_select$ largest values in $s$
    $val \leftarrow \sum_{j \in \text{indices}} s[j]$
    if $val > best\_val$ then
        $best\_val \leftarrow val$
        $indices_{best} \leftarrow indices$
    end if
end for
$p_s \leftarrow \frac{p}{\sum_i sims[i]}$ for $p$ in $sims[i]$ for $i$ in $indices_{best}$      ▷ Normalization
$M_{best} \leftarrow M_{cc}[i]$ for $i$ in $indices_{best}$
$p_s \leftarrow p_s.mul(cluster\_centre\_len)$            ▷ Probabilities
$p_s \leftarrow \frac{p_s}{p_s.sum()}$
return $M_{best}, p_s, indices_{best}$
```

Meta-learning for External Aggregation Method. The method employs meta-learning concepts to utilize the best models ($\mathbf{M_{best}}$), their corresponding probabilities (p_s), and their indices ($\mathbf{indices_{best}}$) for external aggregation, yielding an updated model that encapsulates learning from diverse data distributions.

Firstly, the method extracts the probabilities associated with the currently considered best cluster centres and stores them in the list $\mathbf{conc\_p_s}$: $\mathbf{conc\_p_s} = [\mathbf{p_s}[\mathbf{i}]$ for $\mathbf{i}$ in $\mathbf{indices_{best}}]$.

Secondly, these probabilities are normalized to ensure that the relative importance of each cluster centre is preserved, essential in a FL context where each client's model update is typically weighted by the size of its data: $\mathbf{conc\_p_s} = [\mathbf{p} / \sum_{\mathbf{p} \in \mathbf{conc\_p_s}} \mathbf{p}$ for $\mathbf{p}$ in $\mathbf{conc\_p_s}]$.

Finally, the weighted average of the selected models is computed using these normalized probabilities. This aggregation is performed using an external aggregator like FedAvg, passing in a list of pairs $(\mathbf{conc\_p_s}, \mathbf{M_{best}})$, where $\mathbf{conc\_p_s}$ is the normalized probability associated with the cluster centre and $\mathbf{M_{best}}$ represents the selected models in that cluster centre. The external aggregation is represented as: $\mathbf{M_c} = \sum_{\mathbf{i} \in \mathbf{indices_{best}}} \mathbf{conc\_p_s}[\mathbf{i}] \cdot \mathbf{M_{best}}[\mathbf{i}]$. Here, $\mathbf{M_c}$ denotes the resulting aggregated model, capturing the shared learning across the selected clusters.

Personalized Model Sharing: RFCL focuses on each client's unique data distribution. Detailed in Algorithm 4, the process involves the selective distribution of the external aggregated model $\mathbf{M_c}$, not to all clients, but only to those associated with the selected cluster centres identified during the similarity analysis process. This ensures that each cluster acquires a model tailored to its specific data distribution, which can subsequently be further refined using local client data. The refined model, $\mathbf{M_c}$, is then utilized by the clients in the selected clusters and is updated with their local data. This strategy mirrors the principles of transfer learning, where $\mathbf{M_c}$ is adapted to better fit the specific data distribution of a cluster. In contrast, unselected cluster centres retain their unique models $\mathbf{M_{cc}}$, which are designed to isolate potential attacker updates and avoid sharing the model of $\mathbf{M_c}$ with them. This approach is instrumental in countering potential security threats and managing heterogeneity effectively. The cluster centres update can be expressed as follows:

$$\mathbf{M_{cc}}[\mathbf{i}] = \begin{cases} \mathbf{M_c} & \text{if } i \in \mathbf{indices_{best}} \\ \mathbf{M_{cc}}[\mathbf{i}] & \text{otherwise} \end{cases}$$

This indicates that for every i in the ($\mathbf{M_{cc}}$), if i is in the set of ($\mathbf{indices_{best}}$), then the corresponding cluster centre is replaced with the ($\mathbf{M_c}$) model. Otherwise, the cluster centre remains the same, ensuring that each cluster's unique characteristics are preserved while enhancing the shared knowledge across the selected clusters.

Algorithm 4 Personalization (Cluster-based Model Sharing)

```
for $i \in 0, 1, \ldots, \text{len}(M_{cc}) - 1$ do
    if $i \in indices_{best}$ then
        $M_{cc}[i] \leftarrow M_c$
    end if
end for
```

4 Experiments

We assess the effectiveness of the RFCL method in image classification tasks, employing a machine learning model across three public datasets under various scenarios. The efficiency of RFCL is compared with six existing aggregation methods. The RFCL implementation is publicly available on GitHub[1].

Datasets and Models. We conduct experiments on MNIST dataset [13], CIFAR-10 dataset [12], and Fashion-MNIST dataset [20] for image classification. We utilise a client-server structure consisting of a central server and multiple clients to conduct our experiments until convergence. For all experiment methods, i.e., all clients are selected to provide model updates at each round. The error rate is computed on the test set to evaluate the performance of each aggregation method. Default experimental settings for MNIST, Fashion-MNIST, and CIFAR-10 datasets are in Table 1.

[1] https://github.com/EbtisaamCS/RFCL

MNIST and Fashion-MNIST Model Architecture
DNN (784 × 512 × 256 × 10), with 2 Hidden layers Activation functions: Leaky ReLU Batch size: 64, Loss function: Cross-Entropy Optimizer: SGD (learning rate = 0.1) Dropout: $p = 0.5$
CIFAR-10 Model Architecture
DNN (3072 × 256 × 128 × 10), with 2 Hidden layers Activation functions: Leaky ReLU Batch size: 128, Loss function: Cross-Entropy Optimizer: SGD (learning rate = 0.5)

Table 1: Models and parameters in the experiments.

Attack Methods. Several attack methods are considered to evaluate the robustness of the RFCL framework.

Inner Product Manipulation Attack. The IPM attack aims to evade detection by disguising the attack within the original gradient direction while maintaining the same norm as the original gradient, thus avoiding detection. This results in a compromised model with reduced accuracy. The attack manipulates the gradients of the clients by computing a malicious gradient update $\mathbf{\Delta g^i_t}$ according to the equation: $\mathbf{\Delta g^i_t} = \epsilon \cdot \sum_{\mathbf{j=1}}^{\mathbf{d}} \mathbf{g^i_{t,j}} \cdot \mathbf{w_j}$. Where $\mathbf{d}$ is the dimension of the model parameters, $\mathbf{w}_j$ is the $\mathbf{j}$-th element of the model parameters, and ϵ is a scalar multiplier chosen by the attacker to maximize the impact on the final model parameters while avoiding detection [21]. The epsilon (ϵ) is a scalar multiplier used to control the magnitude of the perturbations applied to the gradients.

A Little is Enough Attack. The ALIE attack aims to manipulate the noise in an undetected way while still deceiving the aggregation rules. It assumes that the benign updates are typically distributed. The attackers exploit the high empirical variance between the updates of clients and upload a noise in a range without being detected. To achieve this, for each coordinate $\mathbf{i} \in [\mathbf{d}]$, the attacker calculates the mean ($\mu_\mathbf{i}$) and standard deviation ($\delta_\mathbf{i}$) over benign updates. The attacker then sets the corrupted updates to values within the range ($\mu_\mathbf{i} - \mathbf{z}_{\max}\delta_\mathbf{i}, \mu_\mathbf{i} + \mathbf{z}_{\max}\delta_\mathbf{i}$), where $\mathbf{z}_{\max}$ ranges from 0 to 1 and is obtained from the Cumulative Standard Normal Function [1].

Sign Flipping (SF) Attack. SF attack does not require access to model updates from other clients like IPM and ALIE attacks [11]. Instead, the SF flips the signs of the gradient. This strategy aims to maximize the loss via gradient ascent rather than gradient descent.

Random Noise (RN) Attack. RN introduces random noise to the model parameters during training by generating a perturbation based on a Gaussian distribution. Byzantine clients added these perturbations to the model parameters to mislead the training process and degrade the model's performance [17].

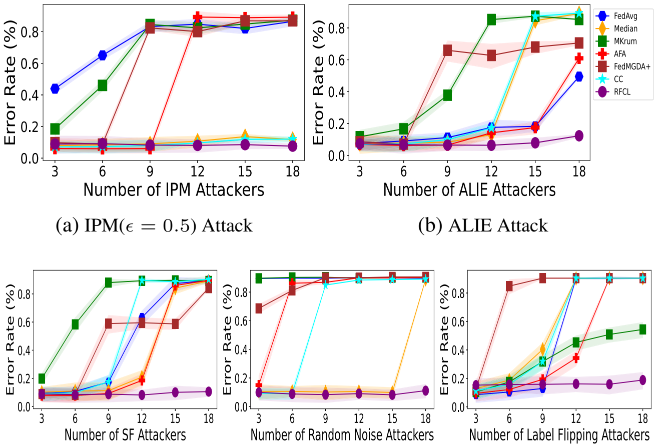
Label Flipping (LF) Attack. LF is a data poisoning attack in which malicious clients modify their dataset to conduct targeted attacks on the model [19]. The attack involves changing the class of each instance in the dataset to be the target of a misclassification attack.

Non-IID Degree. A Dirichlet distribution simulates a Non-IID by splitting training images into 30 clients. A high value of the α parameter results in a low variance of both class and quantity, resulting in low deviation splitting amongst clients. In contrast, a low value of the α parameter increases the variance of the clients, resulting in a significantly Non-IID data split. Non-IID data distributions with two degrees are simulated: a slightly Non-IID ($\alpha = 0.5$) and an extremely Non-IID ($\alpha = 0.1$).

Number of Attackers. In our training process, we allocated a certain number of clients each round, with a portion of these clients programmed to initiate attacks. We put our RFCL method to the test under six distinct attacker scenarios, with the count of attackers varying across scenarios, specifically being 3, 6, 9, 12, 15, and 18 clients.

Comparison Methods. We compared our method performance, RFCL, to the baseline robust aggregation rules that depend on distance-based outlier detection as MKrum [3], Median [23], AFA [17], FedMGDA+ [9], and CC [11].

Experiment Results. On the MNIST dataset, we run FedAvg and other robust methods MKrum, Median, AFA, FedMGDA+, CC, and RFCL, to train the federated model. Then we measure the performance of these methods in the presence of IPM, ALIE, SF, RN, and LF attackers under Non-IID distribution. We repeat all experiments re-sampling the dataset five times and evaluate the average results. In all plots, the error bars show the confidence interval $\rho = \mathbf{0.01}$. When we say that a result is significantly better than another, we mean with statistical significance considering $\rho \leq \mathbf{0.01}$.



(a) IPM($\epsilon = 0.5$) Attack

(b) ALIE Attack

(c) SF Attack

(d) RN Attack

(e) LF Attack

Figure 2: Performance comparison of FedAvg, Median, MKrum, AFA, FedMGDA+, CC, and RFCL methods on MNIST dataset under a Non-IID ($\alpha = 0.5$) scenario. The methods are evaluated against various numbers of IPM, ALIE, SF, RN, and LF attackers.

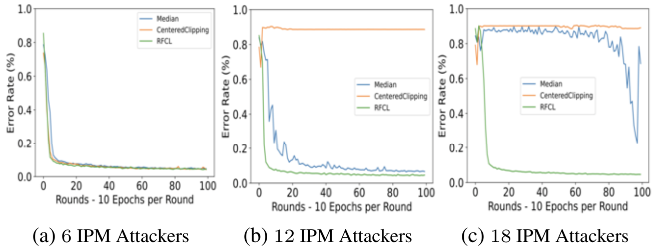
Figure 2 shows the performance of aggregation methods and RFCL method under the Non-IID degree $\alpha = 0.5$ with an increasing number of attackers: RFCL outperforms the other methods.

As shown in Figure 2(a) when the number of IPM attackers is 3 and 6, RFCL, Median, AFA, FedMGDA+, and CC can defend against these attacks. Whereas the accuracy significantly drops for FedAvg and MKrum methods. Furthermore, when the number of attackers increases, the RFCL, Median, and CC are still robust.

For further analysis, we evaluated Median, CC, and RFCL under IPM attacks with increased perturbation magnitude.Figure 3 demonstrates the effect on the performance of various aggregation methods when the value of ϵ is increased, considering different numbers of attackers. When the value of ϵ increases from 0.5 to 100.0, the magnitude of the perturbations applied to the gradients also increases. This means that the gradients are changed more drastically, potentially leading to larger changes in the model parameters during the update step. This can cause the learning process to deviate more significantly from the desired direction, which may negatively affect the performance of the aggregation methods.

Figure 2(b) shows the performance of aggregation rules under ALIE attacks. When the number of attackers is 3 and 6, all methods perform approximately equivalent to a low average error rate. However, the robust methods record a high average error rate when the number of ALIE attackers increases. It shows that, even at this level of Non-IID, RFCL can eliminate ALIE gradients from the global model aggregation. This is because RFCL used clustering with a personalization model-sharing method.

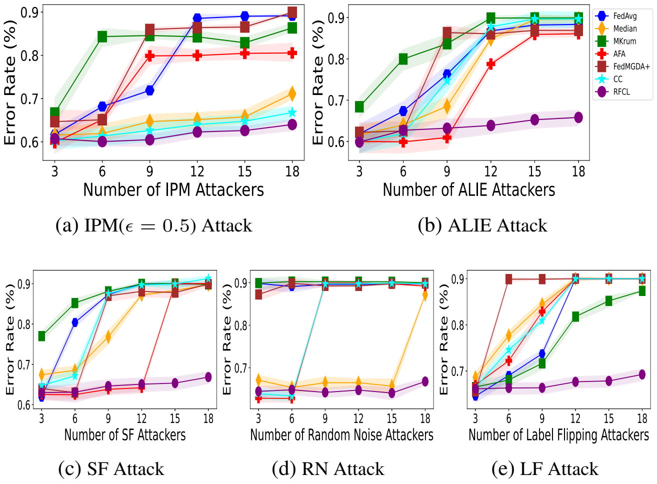
Figure 2(c), (d), and (e) shows the performance of aggregation rules under SF, RN, and LF attacks. RFCL outperform the other robust aggregation rules with the lowest testing error rate.



(a) 6 IPM Attackers (b) 12 IPM Attackers (c) 18 IPM Attackers

Figure 3: Comparison of each round performance of Median, CC, and RFCL Methods on MNIST under Non-IID ($\alpha = 0.5$) scenario against the different number of IPM ($\epsilon = 100.0$) attacks.

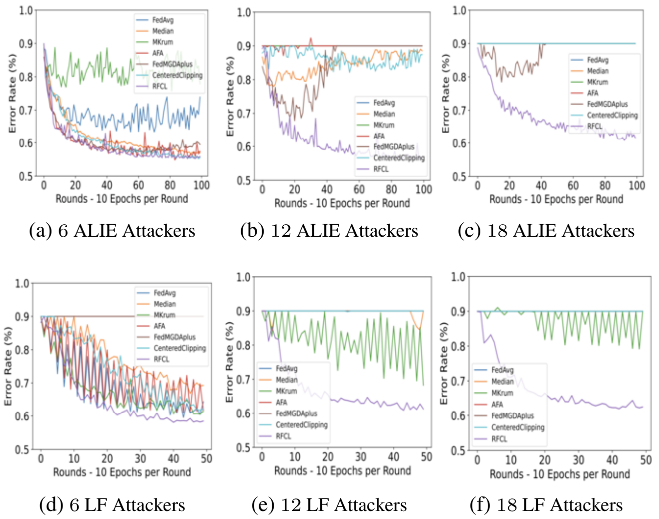
To further validate the RFCL method, we evaluate robust aggregation schemes on CIFAR-10. Figure 4 shows RFCL achieves the highest performance when compared to other methods in this situation with non-ID degrees of $\alpha = 0.5$ and an increasing number of attackers. RFCL, CC, and Median performance have the lowest testing error rate under various IPM ($\epsilon = 0.5$) attackers, while the other methods are entirely disabled, as shown in Figure 4(a). Figure 4(b), (c), (d), and (e) depicts the impact of ALIE, SF, RN, and LF attacks on the aggregation methods, highlighting that RFCL achieves the lowest error rate among them. RFCL's combination of multi-centre, clustering, similarity analysis, and personalized model sharing enables it to outperform other robust methods, particularly when the number of attackers exceeds half the total number of clients. This is because it effectively isolates attackers within outlier clusters and focuses on the most similar models for aggregation, reducing the impact of adversaries on the FL process.



(a) IPM($\epsilon = 0.5$) Attack (b) ALIE Attack

(c) SF Attack (d) RN Attack (e) LF Attack

Figure 4: Performance comparison of FedAvg, Median, MKrum, AFA, FedMGDA+, CC, and RFCL methods on CIFAR-10 dataset under a Non-IID ($\alpha = 0.5$) scenario. The methods are evaluated against various numbers of IPM, ALIE, SF, RN, and LF attackers.

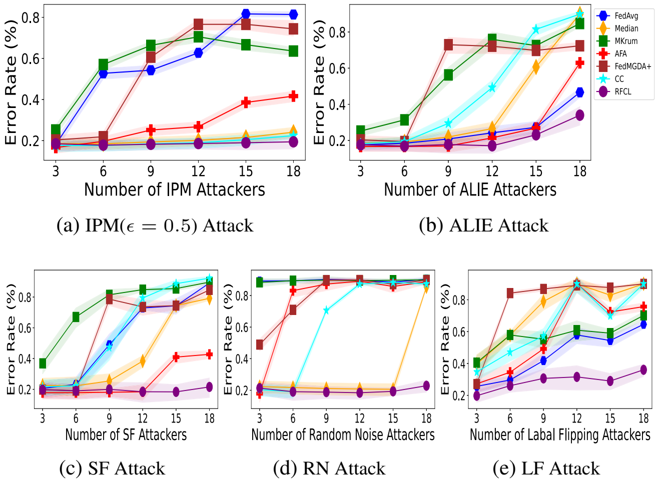


(a) 6 ALIE Attackers (b) 12 ALIE Attackers (c) 18 ALIE Attackers

(d) 6 LF Attackers (e) 12 LF Attackers (f) 18 LF Attackers

Figure 5: Comparison each round performance of FedAvg, Median, MKrum, AFA, FedMGDA+, CC, and RFCL Methods on CIFAR-10 under Non-IID ($\alpha = 0.5$) scenario against the different number of ALIE and LF attacks.

Figure 5 demonstrates the error rate for each round when the methods encounter 6, 12, or 18 ALIE and LF attackers. RFCL performance has the lowest testing error rate. The combination of HDBSCAN clustering and PCA helps identify and group similar models. This method is robust against high-dimensional attacks and helps isolate the effect of malicious clients, even when they form a majority. RFCL selects the most similar clusters using cosine similarity, likely to contain benign models. This helps identify and exclude outlier models not detected in the clustering phase. When many attackers are present as 12 and 18, this method ensures that the most similar models are selected for aggregation.

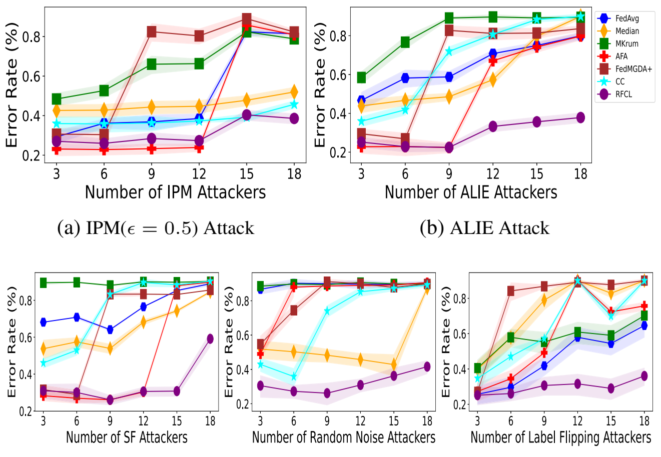


(a) IPM($\epsilon = 0.5$) Attack (b) ALIE Attack

(c) SF Attack (d) RN Attack (e) LF Attack

Figure 6: Performance comparison of FedAvg, Median, MKrum, AFA, FedMGDA+, CC, and RFCL methods on Fashion-MNIST dataset under Non-IID ($\alpha = 0.5$). The methods are evaluated against various numbers of IPM, ALIE, SF, RN, and LF attackers.

To further validate our method, we evaluate it on Fashion-MNIST. Figure 6 shows RFCL achieves the highest performance compared to other methods in this situation with non-ID degrees of $\alpha = 0.5$ and an increasing number of attackers.



(a) IPM($\epsilon = 0.5$) Attack (b) ALIE Attack

(c) SF Attack (d) RN Attack (e) LF Attack

Figure 7: Performance comparison of methods on Fashion-MNIST dataset under Non-IID ($\alpha = 0.1$).

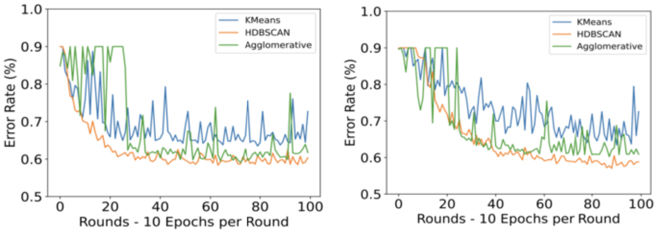
A comprehensive analysis of all methods and RFCL method in a highly heterogeneous scenario. Figure 7 shows that the RFCL method performs well in extremely heterogeneous data distributions. RFCL's clustering with the personalization approach ensures that each client receives a model more aligned with its data distribution. This benefits non-IID settings, allowing clients to learn more effectively from their local data. However, other methods share the final global model with all clients. To conclude, RFCL's methods of multi-centre meta-learning, selective personalization, cosine similarity-based selection, and advanced clustering techniques make it highly effective against data and model poisoning attacks, even when the number of attackers is increased and in non-IID settings.

Different Clustering Methods. In the RFCL method, we further explore its robustness using different clustering methods, such as Agglomerative, K-Means, and HDBSCAN. The comparative analysis is illustrated in Figure 8, depicting the performance of these methods under the IPM and ALIE attack on the CIFAR-10 dataset. HDBSCAN seems to provide the best results.

K-Means, a widely used clustering technique, however, requires a predefined number of clusters. While this might limit its flexibility in certain scenarios, its ability to adapt to different data distributions can be advantageous in specific use cases. In our experiments, we adjust the number of K-means clusters (C) to be five more than the number of attackers (M), i.e., C = M + 5. This adjustment ensures comprehensive data analysis using the K-means algorithm.

Agglomerative clustering operates differently. It initially treats each data point as a separate cluster, gradually merging them bottom-up based on their similarities [16]. This hierarchical technique can help visualize the data and decide the number of clusters. However, the computational complexity of Agglomerative clustering can make it less suited for larger datasets.

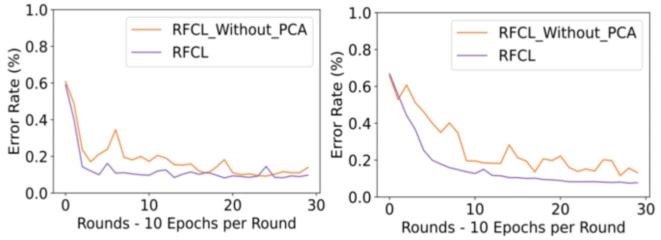
HDBSCAN, a density-based clustering method, excels in identifying clusters of various densities and does not necessitate specifying the number of clusters beforehand [5]. This can provide a distinct advantage in identifying dense regions of similar models while isolating outliers in the RFCL method. Setting appropriate parameters for $min\_cluster\_size$ and $min\_samples$ in the HDBSCAN algorithm relies on domain knowledge and empirical experimentation. Specifically, the $min\_cluster\_size$ parameter could be established between $[24 - 18]$, anticipating varying attacks smaller than 12 attackers. Concurrently, $min\_samples$ can be between $[6-12]$. In scenarios where the number of attackers escalates to a higher proportion, such as 15 or 18, would involve adjusting the $min\_cluster\_size$ between [12-9] while setting $min\_samples$ between [18-21]. This configuration is based on a domain-driven approach that considers the expected number of attackers and empirical observations derived from the performance of the clustering algorithm under various parameter settings. It's important to note that these parameter settings should be continually evaluated and adjusted to optimize the model's ability to detect and handle outliers effectively.



(a) 18 IPM Attackers (b) 18 ALIE Attackers

Figure 8: Impact of Kmeans, Agglomerative, and HDBSCAN clustering methods on CIFAR-10

Ablation study. We conduct ablation studies to evaluate the impact of each component of our method. Figure 9 demonstrates the performance of RFCL when PCA is removed from the process. The RFCL's performance exhibits minimal noise without PCA when subjected to RN and LF attacks. This indicates that the RFCL can maintain a degree of robustness even without the PCA step. However, it is worth noting that the inclusion of PCA does enhance the overall performance slightly, suggesting that while not critical, PCA contributes positively to the resilience and effectiveness of the RFCL method.



(a) Random Noise Attack (b) Label Flipping Attack

Figure 9: Ablation Study concerning using PCA

5 Conclusion

RFCL presents a novel, robust federated learning approach that addresses the significant security concerns of data and model poisoning attacks in heterogeneous data settings. The method employs a unique clustering strategy, focusing on grouping similar models from participating clients, consequently enhancing the selection of high-quality models. RFCL's innovative meta-learning phase and the subsequent personalization process significantly enhance the performance and security of the federated learning system.

Acknowledgements

This work was funded by the Saudi Arabian Ministry of Education, the Saudi Arabian Cultural Bureau in London, and the Umm Al-Qura University in Makkah. We extend our sincere gratitude to the High-End Computing facility of Lancaster University for providing valuable computing resources that greatly supported this research.

References

[1] Gilad Baruch, Moran Baruch, and Yoav Goldberg, 'A little is enough: Circumventing defenses for distributed learning', *Advances in Neural Information Processing Systems*, **32**, (2019).

[2] Arjun Nitin Bhagoji, 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).

[3] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer, 'Machine learning with adversaries: Byzantine tolerant gradient descent', *Advances in Neural Information Processing Systems*, **30**, (2017).

[4] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konečnỳ, Stefano Mazzocchi, Brendan McMahan, et al., 'Towards federated learning at scale: System design', *Proceedings of Machine Learning and Systems*, **1**, 374–388, (2019).

[5] Ricardo JGB Campello, Davoud Moulavi, Arthur Zimek, and Jörg Sander, 'Hierarchical density estimates for data clustering, visualization, and outlier detection', *ACM Transactions on Knowledge Discovery from Data (TKDD)*, **10**(1), 1–51, (2015).

[6] Xiaoyu Cao, Minghong Fang, Jia Liu, and Neil Zhenqiang Gong, 'Fltrust: Byzantine-robust federated learning via trust bootstrapping', *arXiv preprint arXiv:2012.13995*, (2020).

[7] Clement Fung, Chris JM Yoon, and Ivan Beschastnikh, 'Mitigating sybils in federated learning poisoning', *arXiv preprint arXiv:1808.04866*, (2018).

[8] Rachid Guerraoui, Sébastien Rouault, et al., 'The hidden vulnerability of distributed learning in byzantium', in *International Conference on Machine Learning*, pp. 3521–3530. PMLR, (2018).

[9] Zeou Hu, Kiarash Shaloudegi, Guojun Zhang, and Yaoliang Yu, 'Federated learning meets multi-objective optimization', *IEEE Transactions on Network Science and Engineering*, (2022).

[10] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al., 'Advances and open problems in federated learning', *Foundations and Trends® in Machine Learning*, **14**(1–2), 1–210, (2021).

[11] Sai Praneeth Karimireddy, Lie He, and Martin Jaggi, 'Learning from history for byzantine robust optimization', in *International Conference on Machine Learning*, pp. 5311–5319. PMLR, (2021).

[12] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton, 'The cifar-10 dataset'.

[13] Yann LeCun, 'The mnist database of handwritten digits', *http://yann. lecun. com/exdb/mnist/*, (1998).

[14] Suyi Li, Yong Cheng, Wei Wang, Yang Liu, and Tianjian Chen, 'Learning to detect malicious clients for robust federated learning', *arXiv preprint arXiv:2002.00211*, (2020).

[15] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas, 'Communication-efficient learning of deep networks from decentralized data', in *Artificial intelligence and statistics*, pp. 1273–1282. PMLR, (2017).

[16] Daniel Müllner, 'Modern hierarchical, agglomerative clustering algorithms', *arXiv preprint arXiv:1109.2378*, (2011).

[17] 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).

[18] Krishna Pillutla, Sham M Kakade, and Zaid Harchaoui, 'Robust aggregation for federated learning', *arXiv preprint arXiv:1912.13445*, (2019).

[19] Vale Tolpegin, Stacey Truex, Mehmet Emre Gursoy, and Ling Liu, 'Data poisoning attacks against federated learning systems', in *European Symposium on Research in Computer Security*, pp. 480–501. Springer, (2020).

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

[21] Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta, 'Fall of empires: Breaking byzantine-tolerant sgd by inner product manipulation', in *Uncertainty in Artificial Intelligence*, pp. 261–270. PMLR, (2020).

[22] Cong Xie, Sanmi Koyejo, and Indranil Gupta, 'Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance', in *International Conference on Machine Learning*, pp. 6893–6901. PMLR, (2019).

[23] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett, 'Byzantine-robust distributed learning: Towards optimal statistical rates', in *International Conference on Machine Learning*, pp. 5650–5659. PMLR, (2018).

[24] Lei Yu and Lingfei Wu, 'Towards byzantine-resilient federated learning via group-wise robust aggregation', in *Federated Learning*, 81–92, Springer, (2020).