Preprints.org

Article

Not peer-reviewed version

A Robust Federated Learning Against Data Poisoning Attacks: Prevention and Detection of Attacked Nodes

Pretom Roy Ovi * and Aryya Gangopadhyay

Posted Date: 26 June 2025

doi: 10.20944/preprints202506.2218.v1

Keywords: federated learning; data poisoning attack; label flipping attack; privacy preservation

Preprints.org is a free multidisciplinary platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This open access article is published under a Creative Commons CC BY 4.0 license, which permit the free download, distribution, and reuse, provided that the author and preprint are cited in any reuse.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2025 doi:10.20944/preprints202506.2218.v1

Disclaimer/Publisher's Note: The statements, opinions, and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions, or products referred to in the content.

Article

A Robust Federated Learning against Data Poisoning Attacks: Prevention and Detection of Attacked Nodes

Pretom Roy Ovi [1,*] and Aryya Gangopadhyay [2]

[1] University of North Texas, USA
[2] University of Maryland, Baltimore County, USA
* Correspondence: pretomroy.ovi@unt.edu

Abstract

Federated Learning (FL) enables collaborative model building among a large number of participants without sharing the sensitive data to the central server. Because of its distributed nature, FL has limited control over the local data and corresponding training process. Therefore, it is susceptible to data poisoning attacks where malicious workers use malicious training data to train the model. Furthermore, attackers on the worker side can easily manipulate local data by swapping the labels of training instances, adding noise to training instances, adding out-of-distribution training instances in the local data to initiate data poisoning attacks. And local workers under such attacks carry incorrect information to the server, poison the global model, and cause misclassifications. So, prevention and detection of such data poisoning attacks is crucial to build a robust federated training framework. To address it, we propose a prevention strategy in federated learning, namely Confident Federated Learning to prevent the workers from such data poisoning attacks. Our proposed prevention strategy at first validates the label quality of local training samples by characterizing and identifying label errors in the local training data and then exclude the detected mislabeled samples from the local training. To this aim, we experiment with our proposed approach on MNIST, Fashion-MNIST, and CIFAR-10 dataset and experimental results validated the robustness of our proposed Confident Federated Learning in preventing the data poisoning attacks. Our proposed method can successfully detect the mislabeled training samples with above 85% accuracy and exclude those detected samples from the training set to prevent the data poisoning attacks on the local workers. However, our prevention strategy can successfully prevent the attack locally in the presence of certain percentage of poisonous samples. Beyond that percentage, the prevention strategy may not be effective in preventing attacks. In such cases, detection of the attacked workers is needed. So, in addition to the prevention strategy, we propose a novel detection strategy in federated learning framework to detect the malicious workers under attacks. We propose to create a class-wise cluster representation for every participating worker by utilizing the neurons' activation maps of local model and analyze the resulting clusters to filter out the workers under attacks before model aggregation. We experimentally demonstrated the efficacy of our proposed detection strategy in detecting workers affected by data poisoning attacks, along with the attack types, e.g., label-flipping or dirty labeling. In addition, experimental results suggested that the global model couldn't converge even after a large number of training rounds in the presence of malicious workers, whereas after detecting the malicious workers with our proposed detection method and discarding them from model aggregation, the global model ensured convergence within a very few training rounds. Furthermore, our proposed approach stays robust under different data distributions and model sizes and does not require prior knowledge about the number of attackers in the system.

Keywords: federated learning; data poisoning attack; label flipping attack; privacy preservation

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2025 doi:10.20944/preprints202506.2218.v1

1. Introduction

Distributed machine learning has been a topic of interest for many years. Distributed machine learning offers the advantage of processing large amounts of data by distributing the workload among multiple machines [1,2]. It can train models faster and more efficiently, mainly when dealing with large-scale data sets. However, it raises data privacy concerns and requires additional resources for data management and coordination. So, maintaining use privacy and data security is always a critical research problem to solve, and research in this area has been accelerated significantly with the advent of federated learning [3–5]. With the rapid increase in computational capacity of edge devices and advancements in efficient communication technologies with low power consumption, FL is being applied nowadays in many fields [6,7].

The concept of federated machine learning involves collaborative machine learning model building across multiple workers while ensuring data privacy, where the raw data remains locally across numerous local devices. Despite recent progress in federated learning on privacy preservation, FL is still vulnerable to poisoning attacks. Based on the attacker's goal, it has been possible to categorize the attacks into targeted poisoning attack [8,9] and an untargeted poisoning attack [10,11]. A targeted attack is a deliberate attempt by the client to manipulate models' behavior by altering classification decisions or data. Targeted attacks affect only the subset of classes those are under attacks, while no impact on the remaining classes. In data poisoning attacks, a malicious FL participant alters their training data by adding poison to the instances or changing existing instances in an adversarial manner. Moreover, attackers can dominate a certain number of participants and inject poisonous data directly into the model's training data to degrade the model's performance. Also, due to the server's limited authority over the activities and local data of each client involved in the process, any of them can deflect from normal behavior and poison the global model.

Existing prevention methods of data poisoning attacks are primarily designed for centralized data collection scenarios, and their effectiveness in federated learning settings needs further investigation. Recent research on poisoning attacks and their defense strategies in FL often relies on unfeasible assumptions. In particular, these assumptions encompass aspects such as the percentage of compromised clients, the proportion of poisoned samples, the number of manipulated labels, prior knowledge about the number of adversaries, and the type of the FL system. Therefore, there is a need for further research to make federated learning more resistant to poisoning attacks, specifically to data poisoning attacks. Authors [12] pointed out that dirty-label data poisoning attacks tend to cause a lot of misclassifications, up to 90%, when an attacker adds a small number of dirty-label samples to the training dataset. And authors [13] pointed out the data poisoning attacks in FL as an issue that needs immediate addressing. This severe issue can compromise the accuracy and reliability of the machine learning model.

To address these issues mentioned above, we first focus on the vulnerability of FL systems to malicious participants who aim to poison the global model. We attempt to analyze the effect of data poisoning attacks in federated training. In data poisoning attacks, it is assumed that attackers/malicious participants manipulate the local training data on their local devices. The label-flipping attack is the most attractive among all data poisoning attacks due to its simplicity and significant impact. It allows even non-expert attackers to carry out data poisoning attacks without knowing the model type, parameters, or FL process. Label-flipping attacks are already shown to be effective against traditional, centralized ML models [14]. This paper focuses on the targeted label-flipping attack, where the adversaries maliciously alter the true labels of training samples with the chosen class labels.

In this paper, we first focus on developing a prevention strategy utilizing the concept of confident learning [15] to validate the quality of the data label and propose a *Confident Federated Learning (CFL)* framework to prevent label-flipped data poisoning attacks. We also demonstrate the potential of our proposed approach in preventing workers from label-flipped data poisoning attacks. Secondly, we found that our prevention strategy can prevent the attack when the percentage of label-flipped samples is under certain threshold. In certain cases, the prevention strategy may not be effective. So in addition to prevention strategy, we also propose a clustering-based detection strategy to detect

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2025 doi:10.20944/preprints202506.2218.v1

poisonous workers attacked. So, we divided the paper into two parts, the first part is on developing a prevention strategy against attacks (Section 2) and the second part is on developing a detection strategy (Section 3).

2. Prevention Strategy of Data Poisoning Attacks

We first conduct experiments to showcase the effect of label-flipped data poisoning attacks in federated learning setup. Our findings reveal that the effectiveness of the attack, measured by the decrease in model utility, depends on the percentage of malicious users involved. Even when only a small percentage of users are malicious, the attack can still be effective. Additionally, we found that such attacks are targeted, which means that attacks have a significantly negative impact on a subset of classes which are under attack while having little to no impact on the remaining classes. This is advantageous for adversaries who want to poison only a specific subset of classes while avoiding the detection of a completely corrupted global model. Finally, we utilize the concept of confident learning [15] to validate the label quality of the data and propose a *Confident Federated Learning (CFL)* framework to prevent label-flipped data poisoning attacks. We also demonstrate the potential of our proposed approach in preventing the workers from label-flipped data poisoning attacks. Our approach typically involves estimating label noise probabilities and a confidence threshold to identify potentially mislabeled instances and then excludes them from the local training on the worker sides.

The major contributions we have included in this section are:

- *Effect of Compromised workers on Performance:* In the federated training setup, we first showed how compromised clients under data poisoning attacks affect the performance of the global model. Experimental results suggested that data poisoning attacks could be targeted, and even a small number of malicious clients cause the global model to misclassify instances belonging to targeted classes while other classes remain relatively intact.
- *Proposed Prevention Approach against Data Poisoning Attacks:* We propose a confident federated learning (CFL) framework to prevent data poisoning attacks during local training on the worker sides. Along with ensuring data privacy and confidentiality, our proposed approach can successfully detect mislabeled samples, discard them from local training, and ensure prevention against data poisoning attacks.

2.1. Background Literature

By nature, deep learning models necessitate a substantial amount of data to make good predictions. This specific requirement gives rise to a potential problem of the data breach. Moreover, numerous adversarial attacks have been launched to target machine learning and deep learning models. However, most research has focused on poisoning the models in the traditional setting, where a centralized party collects the training data. So, many of the attacks and defenses designed for traditional machine learning do not apply to FL because the server only observes local updates from FL participants, not their instances.

The growing usage of federated learning has prompted research into different types of attacks, such as backdoor attacks [16,17], gradient leakage attacks [18–20], poisoning attacks [8,21,22] and membership inference attacks [23]. The type of attack that is relevant to this chapter is poisoning attacks, which can be classified into two categories: model poisoning and data poisoning. Model poisoning can be effective in specific scenarios, whereas data poisoning may be preferable due to the fact that it does not require malicious manipulation of the model learning software on participant devices, making it efficient and accessible for non-expert poisoning participants. Our research explicitly investigates data poisoning attacks in the context of federated learning. Authors [12] launched targeted backdoor attacks using data poisoning and pointed out that using only a small number of poisonous samples achieves an attack success rate of above 90%. A backdoor poisoning attack was proposed in [16,24], in which the attacker injects backdoored inputs into local data to modify specific features of training data and implant backdoors in the global model. An aggregation-agnostic attack, which

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2025 doi:10.20944/preprints202506.2218.v1

continuously adds noises to the model parameters to introduce backdoors and restrict the global model from converging, was developed in [25]. Data poisoning attacks can cause substantial drops in classification [8]. In data poisoning attacks [8], the adversary can introduce a number of data samples it wishes to miss-classify with the desired target label into the training set. This data-centric poisoning can be two types- label-flipping [26] and dirty labeling. An attacker can easily manipulate its local data by directly swapping the labels of honest training instances of one class (the source class) to a specific target class while keeping the features of the training data unchanged. It is known as the label-flipping attacks [27], which can cause significant drops in the performance of the global model even with a small percentage of malicious clients [8]. And in dirty labeling, the attacker aims to add some out-of-box samples (irrelevant samples for training) to the training dataset and mislabels them. For example, the server wants the global model to be trained on hand written digits, but the attacker adds some samples on the training set which are not even the digit (e.g., animal images but label them as digit class). This type of labeling is known as dirty labeling.

Several methods have been suggested to address poisoning attacks in FL [28–31]. These defense strategies are designed to identify malicious updates that deviate from benign updates. For instance, Auror [28] attempts to cluster model updates into two classes, in which the smaller class will be identified as the malicious class and filtered out. Furthermore, the author introduces Byzantine-robust FL methods: Krum [29], in which the client with the smallest sum of distances to other clients will be selected as the global model. However, in the presence of a large number of malicious participants, these approaches were found to be ineffective in defending against the attacks. Author in [8] presented a defense mechanism that leverages parameter extraction combined with PCA for dimensionality reduction to differentiate malicious and legitimate participants in model updates.

Author introduced an adaptive federated aggregation technique [32], which involves comparing gradients against medians and adjusting weights accordingly. This method is particularly effective in detecting and mitigating malicious gradients. Their experiments, focusing on label-flipping and backdoor attacks with a network of 51 clients, demonstrated efficacy especially in scenarios where less than 50% of participants were malicious. Author advocated the defense mechanism CONTRA in FL [33]. This system employs cosine similarity, an adaptive learning rate, and a reputation-based scheme. It works by assessing the credibility of clients, adjusting local learning rates, and evaluating client contributions based on their historical contribution, irrespective of the data distribution being IID or non-IID. In [34], author introduced a novel defense approach which involves extracting and clustering the parameter gradients from the output layer's neurons of local updates. This method enables the identification and removal of undesirable updates within the clusters. In [35], author proposed an enhanced defense mechanism, building upon previous work [8]. Their approach uses an improved variant of the dimensionality reduction algorithm, PCA, and subsequently applied k-means clustering on IID data. Author [36] evaluated the impact of unreliable clients on the system by deriving a convergence upper bound on the loss function based on the gradient descent updates.

Moreover, these defenses are designed for untargeted model poisoning attacks. In contrast, in respect to data poisoning attacks, a very few research [12,22] pointed out the effect of data poisoning on the performance of the global model. However, the research gap lies in the developing prevention strategy of data poisoning attacks that can effectively prevent such data manipulation on the training set.

2.2. Methodology: Proposed Prevention Strategy

In this section, we describe the detailed architecture and work flow of our proposed federated training. Our proposed framework is built on top of the FedAvg[3] algorithm where some of the notations used in this description are $\mathcal{N} = \{1, \ldots, N\}$ signify the set of N clients, each of which has its own dataset $D_{k \in \mathcal{N}}$. Each client trains a local model on its local dataset and only shares the weight updates of local model with the server. Then, the global model, $\mathbf{w}_G$ formation takes place with all the local model updates which can denoted by $\mathbf{w} = \cup_{k \in \mathcal{N}} \mathbf{w}_k$. The proposed federated framework is depicted in Figure 1. The process based on the workload of server and client is described below:

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2025 doi:10.20944/preprints202506.2218.v1

1. **Executed in server**
   - *Weight initialization:* The server determines the type of application and how the user will be trained. Based on the application, the global model is built in the server. The server then distributes the global model $\mathbf{w}_G^0$ to selected clients.
   - *Aggregation and global update:* The server aggregates the local model updates from the participants and then sends the updated global model $\mathbf{w}_G^{t+1}$ back to the clients. The server wants to minimize the global loss function [13] $L(\mathbf{w}_G^t)$, i.e.

$$L(\mathbf{w}_G^t) = \frac{1}{N}\sum_{k=1}^{N} L(\mathbf{w}_k^t) \tag{1}$$

   This process is repeated until the global loss function converges or a desirable training accuracy is achieved. The Global Updater function runs on the SGD [37] formula for weight update. The formal equation of global loss minimization formula by the averaging aggregation at the t^{th} iteration is given below:

$$\mathbf{w}_G^t = \frac{1}{\sum_{k\in\mathcal{N}} D_k}\sum_{k=1}^{N} D_k \mathbf{w}_i^t \tag{2}$$

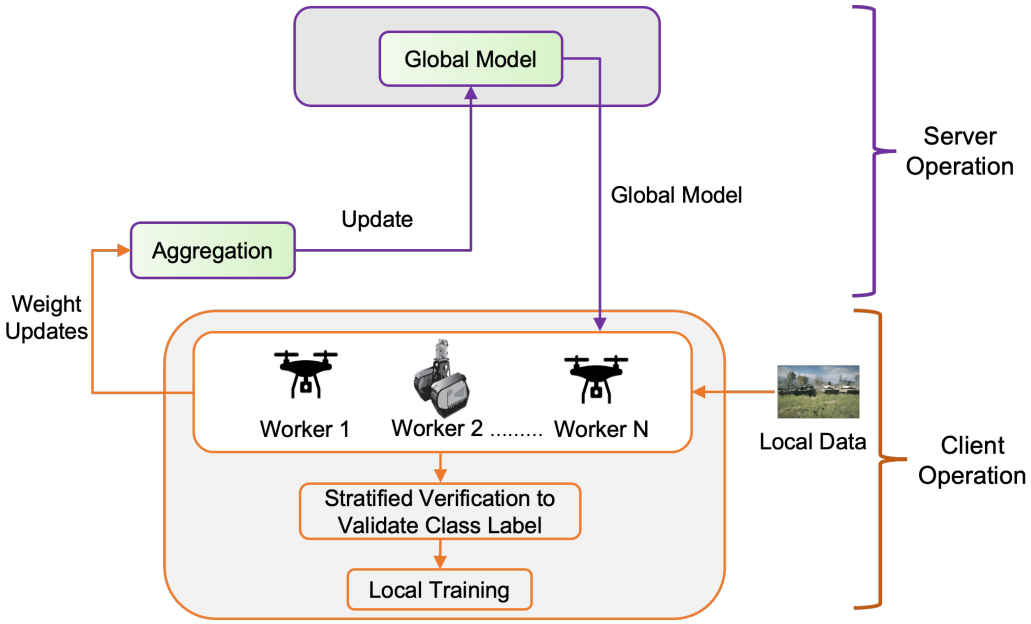


Figure 1. Proposed Confident Federated Learning.

2. **Executed at the client level**
   - *Validate label quality:* Each client will first validate the label quality of local dataset after receiving the global model $\mathbf{w}_G^0$ from the server. To detect the mislabeled samples in local dataset, every participant needs to compute the out-of-sample predictions for every data point of its local dataset. Then out-of-sample prediction and given labels will be utilized to estimate the joint distribution of given and latent true labels, finally estimate the label errors for each pair of true and noisy classes, details in Section 2.3. After validating the class labels of every datapoint in local dataset, overall health score for local dataset can be calculated to track label quality.
   - *Local training:* Each client will utilize $\mathbf{w}_G^t$ from the server, where t stands for each iteration index, start local training on verified local training samples where detected mislabeled

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2025 doi:10.20944/preprints202506.2218.v1

samples are discarded from the local training. The client tries to minimize the loss function [13] $L(\mathbf{w}_k^t)$ and searches for optimal parameters $\mathbf{w}_k^t$.

$$\mathbf{w}_k^{t*} = \arg\min_{\mathbf{w}_k^t} L(\mathbf{w}_k^t) \tag{3}$$

- *Local updates transmission:* After each round of training, updated local model weights are sent to the server afterwards. In addition, each client may send the health score of local data so that the server may monitor the label validation process by tracking the health score.

2.3. Stratified Training to Validate Label Quality

To identify label errors across the local dataset, out-of-sample predicted probabilities for each data point needs to be computed first with K-fold cross-validation. Out-of-sample predicted probabilities refer to the model's probabilistic predictions made only on data points not shown to the model during training. For example, in a traditional train-test split of the data, predicted probabilities generated for the test set can thus be considered out-of-sample. K-fold cross-validation can be used to generate out-of-sample predicted probabilities for every data point in the training set.

The diagram in Figure 2 depicts K-fold cross-validation with K = 5. K independent copies of our model are trained, where for each model copy, one fold of the data is held out from its training (the data in this fold may be viewed as a validation set for this copy of the model). Each copy of the model has a different validation set for which we can obtain out-of-sample predicted probabilities from this copy of the model. Since each data point is held-out from one copy of the model, this process allows us to get out-of-sample predicted probabilities for every data point. We recommend to apply stratified cross-validation, which tries to ensure the proportions of data from each class match across different folds. Then the out-of-sample predicted probabilities and noisy (given) labels will be used to estimate the joint distribution of noisy (given) and true labels, and find the incorrectly labeled samples in the local dataset.

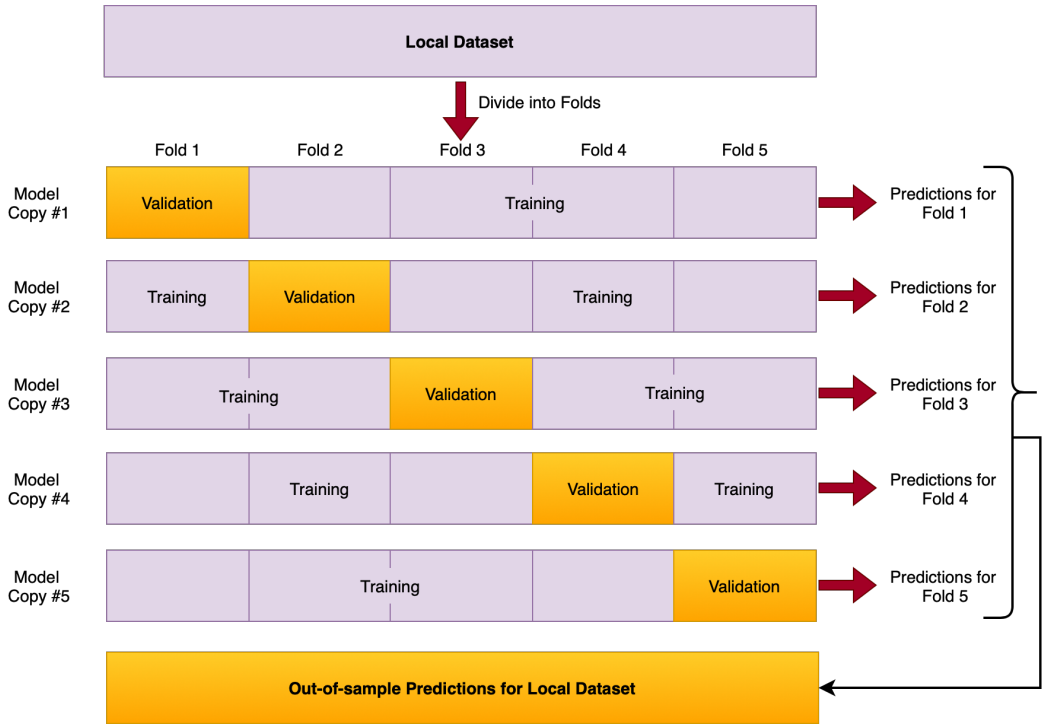


Figure 2. Out-of-sample predictions for local dataset with K-fold cross validation.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2025 doi:10.20944/preprints202506.2218.v1

$$t_j = \frac{1}{|X_{\tilde{y}=j}|} \sum_{x \in X_{\tilde{y}=j}} \hat{p}(\tilde{y} = j; x, \theta) \tag{4}$$

$$\hat{X}_{\tilde{y}=i,y^*=j} = \{x \in X_{\tilde{y}=i} : \hat{p}(\tilde{y} = j; x, \theta) \geq t_j\} \tag{5}$$

Here, $\tilde{y}$ is discrete random variable takes a given noisy label, y^* is discrete random variable takes the unknown, true latent label, the threshold t_j is the expected (average) self-confidence for each class, $\hat{X}_{\tilde{y}=i,y^*=j}$ is the set of examples x labeled $\tilde{y} = i$ with large enough $\hat{p}(\tilde{y} = j; x, \theta)$ to likely belong to class $y^* = j$, determined by a per-class threshold t_j. And $C_{\tilde{y},y^*}$ is the confident joint formally defined as eqn 6. The confident joint $C_{\tilde{y},y^*}$ estimates $X_{\tilde{y}=i,y^*=j}$, the set of examples with noisy label i that actually should have true label j, by partitioning X into estimate bins $\hat{X}_{\tilde{y}=i,y^*=j}$. When $\hat{X}_{\tilde{y}=i,y^*=j} = X_{\tilde{y}=i,y^*=j}$, then $C_{\tilde{y},y^*}$ exactly finds label errors.

$$\begin{aligned} C_{\tilde{y},y^*}[i][j] &:= \left|\hat{X}_{\tilde{y}=i,y^*=j}\right|, \quad \text{where} \\ \hat{X}_{\tilde{y}=i,y^*=j} &:= \{x \in X_{\tilde{y}=i} : \hat{p}(\tilde{y} = j; x, \theta) \geq t_j, \\ &\qquad j = \arg \max_{l \in [m],\, \hat{p}(\tilde{y}=l;x,\theta) \geq t_l} \hat{p}(\tilde{y} = l; x, \theta) \Big\} \end{aligned} \tag{6}$$

2.4. Experiment Setup and Result Analysis

This section will step through the detailed experiment setup and result analysis.

Federated Training Setup: Experiments were conducted on a system with an Intel(R) Core(TM) i9-11900K CPU (8 cores) and an NVIDIA GeForce RTX 3090 GPU. The federated learning framework was implemented using Keras with a TensorFlow backend. The server controlled the training pace, including the number of epochs per round and the overall number of rounds. The server sets the pace of the training, determines the number of epochs per round, and how many rounds of overall training are to be conducted. We utilized the LeNet-5 CNN architecture as a global model and simulated FL training with 15 clients. For each round, 1 epoch of local training is conducted on the client side. We used SGD optimizer and set the learning rate η to 0.01 during training. And the datasets we used in this experiment are pre-divided into training and testing subset. The test set is kept on the sever and used for model evaluation only and is therefore not included in any participants' local train data.

Datasets: To demostrate the attack effect and evaluate the performance of our proposed approach, we conducted experiments on MNIST, Fashion-MNIST, and CIFAR-10 datasets.

MNIST: The dataset comprises handwritten digit images showing individual digits (0 to 9). The training set has 60,000 samples, and the test set has 10,000 samples, with images size of 28×28.

Fashion-MNIST: The dataset includes grayscale images of ten clothing categories, with 60,000 training set samples and 10,000 test set samples, all with an image size of 28×28.

CIFAR-10: The dataset encompasses color images of ten classes, each with 50,000 training set samples (5,000 images per class) and 10,000 test set samples (1,000 images per class) at pixels resolution of 32×32.

Label Flipping Process: We randomly choose N x m% of the participants as malicious to explore the impact of label flipping attack on federated learning system containing N participants, where a particular percentage (m%) of them are malicious. The remaining participants are considered honest. To account for the impact of the random selection of malicious participants, we repeat the experiment multiple times and report the average results. We explore label-flipping attack scenarios for the MNIST, Fashion-MNIST, and CIFAR-10 datasets by flipping the label of the source class to a specific target class denoted as source class $\rightarrow$ target class pairing. For MNIST, we experiment with the following pairings: 0: digit_0 $\rightarrow$ 2: digit_2, 1: digit_1 $\rightarrow$ 5: digit_5, and for Fashion-MNIST, we evaluated the pairing on 1: trouser $\rightarrow$ 3: dress, 0: t-shirt/top $\rightarrow$ 4: coat. For CIFAR-10, we experiment with the following pairings: 6: frog $\rightarrow$ 7: horse, 1: automobile $\rightarrow$ 3: cat.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2025 doi:10.20944/preprints202506.2218.v1

Attack Effects Analysis: Label flipping denoted by src → target class indicates that ground truth labels of source class samples' have been flipped with the target class label. If we train a machine learning model on a dataset in which the ground-truth labels of some examples from the source class have been flipped with those of the target class, some samples from the source class will be predicted as belonging to the target class during testing. This implies that the source class will have some false negatives, which will ultimately impact its Recall. On the other hand, the target class will have some false positives, which will affect its Precision. Label-flipping attacks are targeted attacks, meaning they have a significant negative impact on a subset of classes that are under attack while having no impact on the remaining classes. So, to evaluate the attack effects with varying percentages of malicious participants, we utilize Recall for the source class and Precision for the target class as evaluation metrics.

The Figure 3 illustrates the degradation of the recall of the source class and precision of the target class when the percentage of malicious participants (m) varies from 0% to 40%. The value 0% signifies that none of the participants are malicious, and therefore, there is no poisoning (NP) in the considered scenario. The findings reveal that with the increase of malicious participants, the global model's potential to predict the source and target classes declines. Even with a small percentage of malicious participants, both the source class recall and the target class precision deteriorate in comparison to a non-poisoned scenario (NP) for both MNIST and Fashion-MNIST, depicted in Figure 3a,b respectively. For example, when malicious participants reach 40%, recall drops from 0.99 to 0.73, and precision drops from 0.98 to 0.78 for MNIST. A similar trend is observed for the Fashion-MNIST dataset as well.

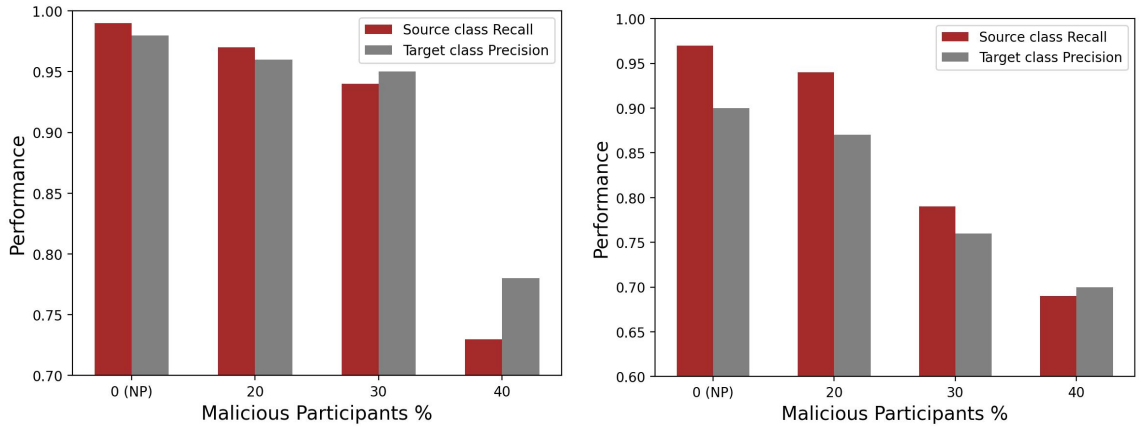


Figure 3. Impact of label-flipping attacks on targeted classes on (a) MNIST data and (b) Fashion-MNIST data.

Furthermore, in Figure 4, we depicted the global model's performance drop (accuracy, source class recall, and target class precision) for CIFAR-10 data with the increasing percentage of malicious participants. It is clear from the figure that the higher the percentage of malicious participants, the more the performance drop can be. In the experiments, once the malicious percentage reaches 50%, the precision and recall of the targeted classes drop around 0.24 and 0.65, respectively, while the overall accuracy of the global model declines only 0.06. This result indicates the targeted nature of label-flipping attacks where the attack degrades the global model's potential in predicting the instances belonging to source and target classes while performance for other classes remains relatively the same. And so, the overall accuracy of the global model deteriorates less in comparison with the drop in precision and recall of target and source class, respectively. Therefore, it is evident that an attacker who controls even a small proportion of the total participants can significantly affect the global model's utility.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2025 doi:10.20944/preprints202506.2218.v1

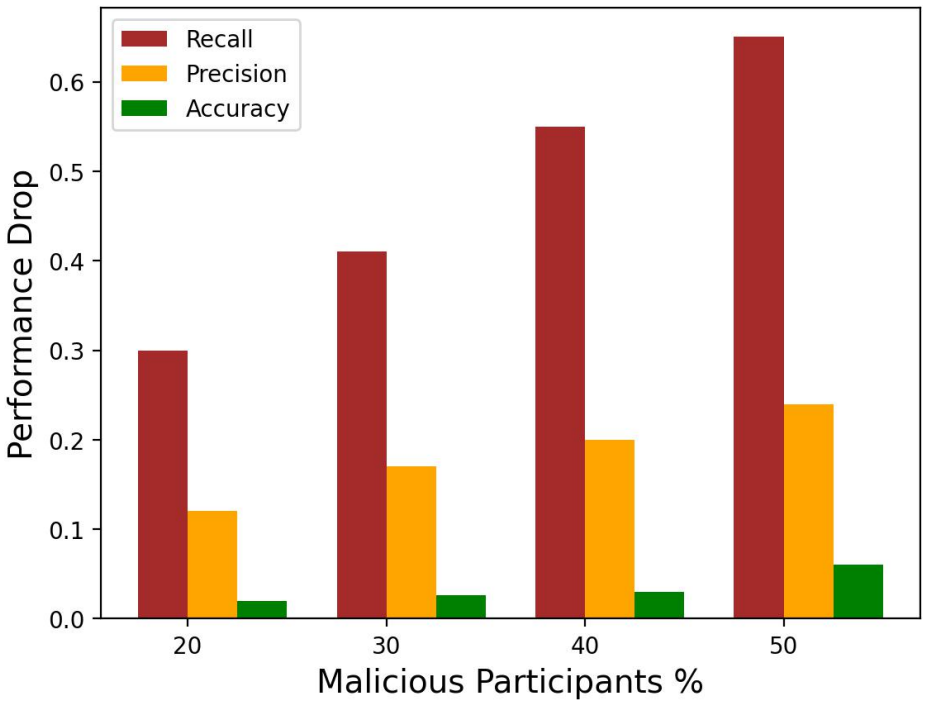
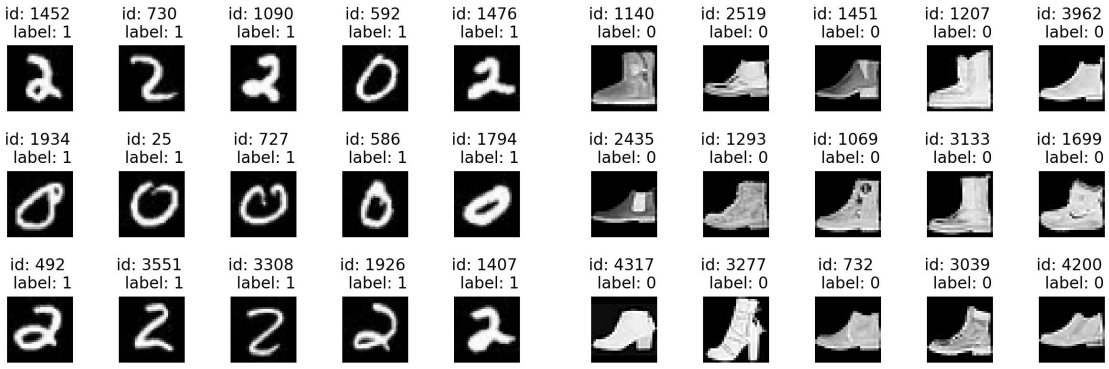


Figure 4. Impact of attacks on the performance for CIFAR-10 data.

Performance Evaluation of Proposed Approach: Based on the analysis presented above, it is clear that preventing data poisoning attacks is crucial in the context of federated training. Our proposed FL framework involves validating the local dataset of each client to identify label-flipped samples prior to the commencement of local training. To evaluate the effectiveness of our approach for detecting mislabeled training samples and preventing them from the local training process, we conducted experiments wherein we deliberately manipulated the local dataset of a few clients to initiate data poisoning attacks. Specifically, we altered the class labels of a few samples in the MNIST dataset (digits 0 and digit 2 images were changed to class label 1). Similarly, in Fashion-MNIST, t-shirt/top images are assigned to class label 0, and ankle boot images are assigned to class label 9. We intentionally altered the label of some ankle boot samples to class label 0. Experimental results using both datasets demonstrate that our proposed approach can successfully detect the poisoned samples of local workers, shown in Figure 5, with an accuracy over 88% for MNIST and 86% for Fashion-MNIST, albeit with a small number of false positives (approximately 5%).



(a) Detected flipped samples on altered MNIST **(b)** Detected flipped samples on altered Fashion-MNIST

Figure 5. Detection of label flipped samples.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2025 doi:10.20944/preprints202506.2218.v1

3. Detection Strategy of Data Poisoning Attacks

The proposed prevention strategy can prevent the attack when the percentage of label-flipped samples is under certain threshold (around 35%). So, for any worker, if incorrectly labeled samples are higher in percentage than the correctly labeled samples and/or the training set of a worker is entirely poisoned, prevention strategy fails to prevent attacks. So, in addition to prevention, we propose a class-wise cluster-based detection strategy to detect the workers affected by data poisoning attacks.

This data poisoning can be of two types- label-flipping and dirty labeling. Label flipping attack is swapping the source label with the target label while keeping the features of the training data unchanged. On the other hand, in dirty labeling, the attackers/poisoned workers add out-of-distribution samples as training instances and mislabel them. Numerous strategies have been developed to counteract poisoning attacks, particularly label flipping, yet they often fall short because they are either impractical or having strong assumption regarding the distribution of local training data. For example, research study [38] assumes the server has some data examples representing the distribution of the workers' data, which is not always a realistic assumption in FL. Some of the defense approaches assume data to be independent and identically distributed (iid) among workers, which cause degradation in performance when the data are non-iid [33]. Some research studies [27,33] detect untrustworthy workers by auditing the model behavior e.g., local gradient/weight updates. However, existing approaches of auditing model behavior may be effective to detect the worker affected by model poisoning attacks but cannot guarantee to detect worker affected by label-flipped data poisoning attacks because malicious workers under label flipped attacks can have similar local updates as trustworthy local workers. Additionally, methods like multi-Krum (MKrum) [29] and trimmed mean (TMean) [39] require prior knowledge about the ratio of attackers in the system, which is impractical. Beyond data distribution or behavioral assumptions, model dimensionality plays a crucial role in these defense mechanisms. High-dimensional models are particularly susceptible to poisoning, as attackers can make minor yet impactful alterations in their local updates that go undetected [40].

Here, we introduce an innovative method to identify attacked workers by examining the neurons' activation of second last layer, which is linked to the neurons of the source and target classes in the SoftMax output layer. Specifically, we find that the discrepancies between the honest workers and compromised workers due to data poisoning attacks are reflected in the neurons' activation of second last layer which is connected to the final output layer. And this activation serves as a more effective discriminative feature for attack detection. And so, our approach involves a cluster-based representation technique to create a distinct cluster representation for each worker that leverages the activation maps of the second last layers for each local training samples and their associated class labels. This clustering-based strategy enables determining whether a worker has been attacked with data poisoning attacks. So, we propose to create a class-wise cluster representation for every worker by utilizing the neurons' activation maps of local model and analyze the resulting clusters to filter out the workers under attacks before model aggregation.

3.1. Related Literatue

The defenses proposed in the current literature to counter poisoning attacks (Label Flipping attacks in particular) against FL are based on one of the following types:

- **Representative Dataset based:** Approaches within this category exclude or penalize a local update if it has a negative impact on the evaluation metrics of the validation dataset of the global model, e.g. accuracy metrics. Specifically, Studies [38,41] use a validation data set on the server to compute the loss on a designated metric caused by local updates of each worker. Subsequently, updates that detrimentally influence this metric are omitted from the aggregation process of the global model. However, the efficacy of this method hinges on the availability of realistic validation data, which implies a necessity for the server to have insights into the distribution of workers' data. This requirement poses a fundamental conflict with the principle of Federated Learning (FL), where the server typically may not have the representative dataset alike workers.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2025 doi:10.20944/preprints202506.2218.v1

- **Clustering Based:** Methods categorized under this approach involve clustering updates into two groups, with the smaller subset being identified as potentially malicious and consequently excluded from the model's learning process. Notably, techniques such as Auror [28] and multi-Krum (MKrum) [29] operate under the assumption that data across peers are independent and identically distributed (iid). This assumption, however, leads to increased rates of both false positives and negatives in scenarios where data are not IID [33]. Additionally, these methods necessitate prior knowledge regarding either the characteristics of the training data distribution [28] or an estimation of the anticipated number of attackers within the system [29].
- **Model behavior Monitoring Based:** Methods under this category operates under the assumption that malicious workers tend to exhibit similar behaviors, which means that their updates will be more similar to each other than to those of honest workers. Consequently, updates are subjected to penalties based on their degree of similarity. For example, FoolsGold (FGold) [27] and CONTRA [33] limit the contribution of potential attackers with similar updates by reducing their learning rates or preventing them from being selected. However, it is worth noting that these approaches can inadvertently penalize valuable updates that exhibit similarities, ultimately resulting in substantial reductions in model performance [42,43].
- **Update aggregation:** This approach employs resilient update aggregation techniques that are sensitive to outliers at the coordinate level. Such techniques encompass the use of statistical measures like the median [39], the trimmed mean (Tmean) [39], or the repeated median (RMedian) [44]. By adopting these methods, bad updates will have little to no influence on the global model after aggregation. It is worth noting, however, that while these methods yield commendable performance when dealing with updates originating from independent and identically distributed (iid) data, their effectiveness diminishes when handling updates from non-iid data sources. This is primarily due to their tendency to discard a significant portion of information during the model aggregation process. Additionally, the estimation error associated with these techniques grows proportionally to the square root of the model's size [40]. Furthermore, the utilization of RMedian [44] entails substantial computational overhead due to the regression procedure it performs, while Tmean [39] demands explicit knowledge regarding the fraction of malicious workers within the system.

In contrast, our proposed cluster based detection technique stays robust under different data distributions and model sizes, and does not require prior knowledge about the number of attackers in the system.

3.2. Methodology: Proposed Cluster-Based Detection Strategy

Federated learning is designed to protect the privacy of data across different workers. And our proposed method ensure that it does not compromise this privacy. Here's a step-by-step breakdown of how it works:

Layer Selection: Identify the layer in the neural network that effectively captures the characteristics of each class is crucial. Typically, layers closer to the output are more class specific. We find that the contradiction between the honest workers and compromised workers due to data poisoning attacks are reflected in the neurons' activation of second last layer. Specifically, we find the neurons' activation of second last layer which is connected to the final output layer as better discriminative feature for attack detection as it can effectively captures the characteristics of each class.

Feature Extraction: For each data point in a worker's local dataset, we extract the feature vector from the selected layer. These feature vectors represent the data in a high-dimensional space where similar items (from the same class) are expected to be closer together. As feature maps are high dimensional, each client will perform dimension reduction technique, e.g., Principal Component Analysis (PCA), on its activation value to convert it to lower dimension. Then each worker will share converted lower dimensional feature maps for each local data point and its associated label with the server. Crucially, as this process involves sharing only the transformed (lower dimensional) activation

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2025 doi:10.20944/preprints202506.2218.v1

maps from a single selected layer, it does not reveal any information about the private training data and so, preserves the privacy of the data. This approach does not violate the privacy-preserving principles of federated learning protocols.

Class-wise cluster representation: Upon receiving the activation maps and class labels for every local data point from participating workers, the server employs a cluster-based representation for each worker. And the clustering is done separately for each class and training samples from the same class label is also expected to be in the same feature space in cluster-based representation. This results in distinct clusters for each class within each worker's dataset.

Cluster Comparison and Attack Detection: Finally, the server compares the class-wise cluster position among the workers to detect the workers affected by data level poisoning. Look for workers whose clusters for a particular class significantly deviate from the clusters of the same class from other workers. We can use statistical tests apart from only visualizing it to determine if the deviations are beyond threshold. Workers with anomalous class-wise clusters are flagged for potential data poisoning attack. This detection process can be considered as a periodic check during the federated learning cycles. If a worker is flagged, it will be considered as malicious workers affected by data poisoning attacks and its contributions will be discarded and excluded from the federated training.

3.3. Result Analysis of Proposed Detection Approach

We conducted federated training on MNIST handwritten digit dataset where a number of workers participated in the training process to build a model. We simulated a federated training set where 2 workers were affected by data poisoning attacks. Among the 2 poisoned workers, 1 worker is attacked with label flipping and other one is attacked with dirty labeling. To launch the label flipped data poisoning attack, we intentionally trained one worker (worker_3) on MNIST data but with label flipped samples (flipping the label of digit 0 and digit 1). So, worker_3 manipulates its local data by swapping the labels of training instances of class label 0 and class label 1 while keeping the training instances unchanged. And another worker (worker_4) is trained on dirty labeled data where images of public transport e.g. bus, metro, airport, and battlefield vehicles e.g. helicopter, tank have been labeled as digit classes. So, the worker_4 utilizes out-of-distribution samples (irrelevant samples), mislabels them (label them as digit classes) and is trained on such manipulated data to launch the dirty label data poisoning attack.

Here, we employed a cluster-based representation for each worker and compared the class-wise cluster among the workers to detect the data level poisoning. The assumption is that a worker trained on poisoned data produce clusters that significantly differ from the majority, indicating potential data poisoning attacks. For simplicity and easy visualization, we visualized only a few digit classes instead of all digit classes to visually inspect the clusters. Figure 6 demonstrates that worker_1 and worker_2 are honest workers and trained on their local MNIST digit data. And they are honest which is evidenced by the nearly identical positions of class-wise clusters in a 3D feature space for majority workers. However, in the same Figure 6 (lower left), a notable difference is observed for worker_3. Here, the blue and orange clusters, which represent digit classes 0 and 1, have swapped positions in the feature space compared to the honest workers. This cluster swap is a result of a label flipping attack, where the labels for digit classes 0 and 1 have been flipped, leading to this altered cluster positions for worker_3. It's important to note that the cluster positions for other digit classes in worker_3 remain largely unchanged and are almost identical to those of the honest workers due to accurate labeling for these other digit classes.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2025 doi:10.20944/preprints202506.2218.v1

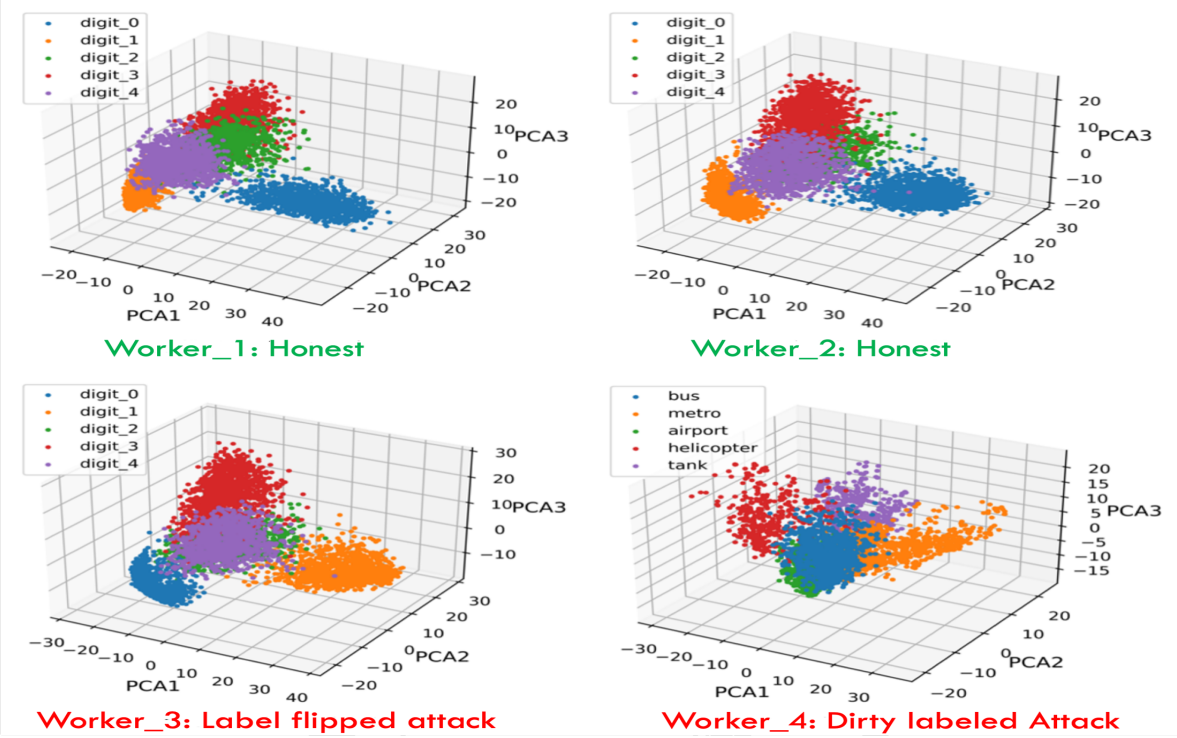


Figure 6. Clustering based approach: (a) honest worker, (b) honest worker, (c) worker attacked with label flipping among digit-0 and digit-1 class, (d) worker attacked with dirty labeling.

As depicted in Figure 6 (lower right), for worker_4, the class-wise cluster positions in the feature space for all classes are completely different from those of the honest workers. This discrepancy arises because worker_4 was trained on dirty labeled data, where images of buses, metros, airports, helicopters, and tanks were labeled as digit classes. Consequently, our cluster-based approach successfully identified both workers who were impacted by label-flipping and dirty-labeling data poisoning attacks.

3.4. Effect of Attack on Model's Convergence

In our experiment, we simulated a federated learning (FL) environment with 25 local workers, among which 5 were intentionally compromised with data poisoning attacks. The presence of these malicious workers hindered the convergence of the global model. As shown in Figure 7a, the global model fails to converge even after 75 communication rounds when trained in the presence of such adversarial participants.

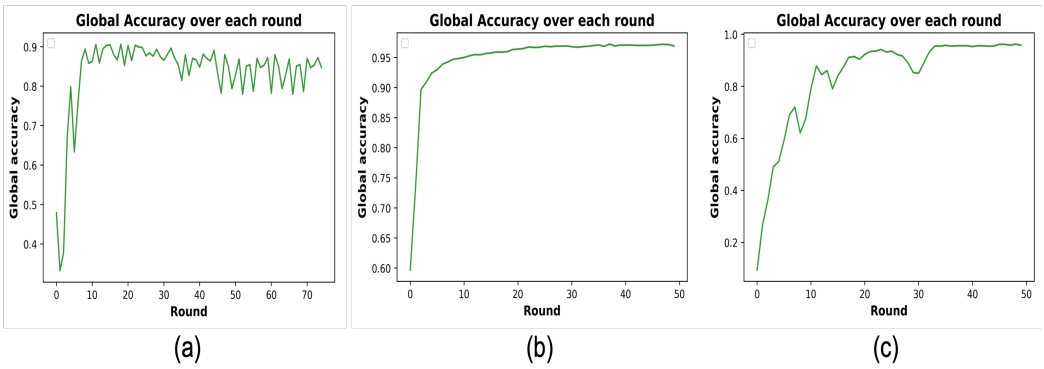


Figure 7. Effect on Convergence. (a) in presence of malicious workers, (b) all honest workers in IID setup, and (c) all honest workers in non-IID setup.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2025 doi:10.20944/preprints202506.2218.v1

After detecting the malicious workers with our proposed detection method and discarding them from training, we evaluated the convergence behavior of the global model under two data distribution settings: independent and identically distributed (IID) and non-independent and identically distributed (non-IID). In the IID setting, each worker received a relatively balanced class distribution of training samples. In contrast, the non-IID setting was configured by assigning imbalanced class distributions to different workers. As in Figure 7, convergence in the IID and non-IID scenario required only 25 and 40 rounds respectively. So, by detecting the malicious workers and discarding them in model building, we found that the global model achieved convergence in significantly fewer training rounds. These findings highlight the detrimental impact of data poisoning on convergence and demonstrate that once malicious workers are effectively detected and discarded from training, convergence of the global model becomes stable.

4. Conclusions

In this paper, we first investigated the feasibility and impact of data poisoning attacks in federated learning, demonstrating that as the number of malicious participants increases, the performance of the global model significantly deteriorates, particularly in terms of precision, recall, and overall accuracy. To counteract this, we evaluated the proposed method for detecting and excluding poisonous samples from local training to prevent label-flipped data poisoning attacks. Our strategy capitalizes on the local nature of FL, where workers retain access to their own data but not others'. By leveraging correctly labeled local data, our approach can identify mislabeled samples within a worker's dataset. However, when a local dataset is entirely or predominantly poisoned, this strategy becomes ineffective, underscoring the need for broader detection mechanisms to identify compromised participants.

We then proposed a robust detection mechanism that constructs class-wise cluster representations based on neuron activation maps from local models. These clusters are analyzed to identify and filter out malicious workers prior to model aggregation. Our experiments confirm the effectiveness of this method not only in detecting compromised workers but also in distinguishing between different types of attacks, such as label-flipping and dirty-labeling. Importantly, our approach remains resilient across varying data distributions and model complexities. Overall, our findings establish the robustness and adaptability of the proposed federated learning framework in mitigating data poisoning threats.

References

1. Dean, J.; Corrado, G.; Monga, R.; Chen, K.; Devin, M.; Mao, M.; Ranzato, M.; Senior, A.; Tucker, P.; Yang, K.; et al. Large scale distributed deep networks. *Advances in neural information processing systems* **2012**, *25*.
2. Duan, M.; Li, K.; Liao, X.; Li, K. A parallel multiclassification algorithm for big data using an extreme learning machine. *IEEE transactions on neural networks and learning systems* **2017**, *29*, 2337–2351.
3. McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; y Arcas, B.A. Communication-efficient learning of deep networks from decentralized data. In Proceedings of the Artificial intelligence and statistics. PMLR, 2017, pp. 1273–1282.
4. Konečnỳ, J.; McMahan, B.; Ramage, D. Federated optimization: Distributed optimization beyond the datacenter. *arXiv preprint arXiv:1511.03575* **2015**.
5. Bonawitz, K.; Eichner, H.; Grieskamp, W.; Huba, D.; Ingerman, A.; Ivanov, V.; Kiddon, C.; Konečnỳ, J.; Mazzocchi, S.; McMahan, B.; et al. Towards federated learning at scale: System design. *Proceedings of Machine Learning and Systems* **2019**, *1*, 374–388.
6. Ovi, P.R.; Dey, E.; Roy, N.; Gangopadhyay, A.; Erbacher, R.F. Towards developing a data security aware federated training framework in multi-modal contested environments. In Proceedings of the Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications IV. SPIE, 2022, Vol. 12113, pp. 189–198.
7. Cho, H.; Mathur, A.; Kawsar, F. FLAME: Federated Learning Across Multi-device Environments. *arXiv preprint arXiv:2202.08922* **2022**.
8. Tolpegin, V.; Truex, S.; Gursoy, M.E.; Liu, L. Data poisoning attacks against federated learning systems. In Proceedings of the European Symposium on Research in Computer Security. Springer, 2020, pp. 480–501.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2025 doi:10.20944/preprints202506.2218.v1

9. Bhagoji, A.N.; Chakraborty, S.; Mittal, P.; Calo, S. Analyzing federated learning through an adversarial lens. In Proceedings of the International Conference on Machine Learning. PMLR, 2019, pp. 634–643.
10. Wu, Z.; Ling, Q.; Chen, T.; Giannakis, G.B. Federated variance-reduced stochastic gradient descent with robustness to byzantine attacks. *IEEE Transactions on Signal Processing* **2020**, *68*, 4583–4596.
11. Fang, M.; Cao, X.; Jia, J.; Gong, N. Local model poisoning attacks to {Byzantine-Robust} federated learning. In Proceedings of the 29th USENIX Security Symposium (USENIX Security 20), 2020, pp. 1605–1622.
12. Chen, X.; Liu, C.; Li, B.; Lu, K.; Song, D. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526* **2017**.
13. Lim, W.Y.B.; Luong, N.C.; Hoang, D.T.; Jiao, Y.; Liang, Y.C.; Yang, Q.; Niyato, D.; Miao, C. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials* **2020**, *22*, 2031–2063.
14. Xiao, H.; Xiao, H.; Eckert, C. Adversarial label flips attack on support vector machines. In *ECAI 2012*; IOS Press, 2012; pp. 870–875.
15. Northcutt, C.; Jiang, L.; Chuang, I. Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research* **2021**, *70*, 1373–1411.
16. Bagdasaryan, E.; Veit, A.; Hua, Y.; Estrin, D.; Shmatikov, V. How to backdoor federated learning. In Proceedings of the International Conference on Artificial Intelligence and Statistics. PMLR, 2020, pp. 2938–2948.
17. Sun, Z.; Kairouz, P.; Suresh, A.T.; McMahan, H.B. Can you really backdoor federated learning? *arXiv preprint arXiv:1911.07963* **2019**.
18. Melis, L.; Song, C.; De Cristofaro, E.; Shmatikov, V. Exploiting unintended feature leakage in collaborative learning. In Proceedings of the 2019 IEEE symposium on security and privacy (SP). IEEE, 2019, pp. 691–706.
19. Zhu, L.; Liu, Z.; Han, S. Deep leakage from gradients. *Advances in neural information processing systems* **2019**, *32*.
20. Ovi, P.R.; Dey, E.; Roy, N.; Gangopadhyay, A. Mixed Precision Quantization to Tackle Gradient Leakage Attacks in Federated Learning. *arXiv preprint arXiv:2210.13457* **2022**.
21. Ovi, P.R.; Gangopadhyay, A.; Erbacher, R.F.; Busart, C. Secure Federated Training: Detecting Compromised Nodes and Identifying the Type of Attacks. In Proceedings of the 2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA). IEEE, 2022, pp. 1115–1120.
22. Sun, G.; Cong, Y.; Dong, J.; Wang, Q.; Lyu, L.; Liu, J. Data poisoning attacks on federated machine learning. *IEEE Internet of Things Journal* **2021**, *9*, 11365–11375.
23. Nasr, M.; Shokri, R.; Houmansadr, A. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In Proceedings of the 2019 IEEE symposium on security and privacy (SP). IEEE, 2019, pp. 739–753.
24. Gu, T.; Dolan-Gavitt, B.; Garg, S. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733* **2017**.
25. Baruch, G.; Baruch, M.; Goldberg, Y. A little is enough: Circumventing defenses for distributed learning. *Advances in Neural Information Processing Systems* **2019**, *32*.
26. Biggio, B.; Nelson, B.; Laskov, P. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389* **2012**.
27. Fung, C.; Yoon, C.J.; Beschastnikh, I. The limitations of federated learning in sybil settings. In Proceedings of the 23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020), 2020, pp. 301–316.
28. Shen, S.; Tople, S.; Saxena, P. Auror: Defending against poisoning attacks in collaborative deep learning systems. In Proceedings of the Proceedings of the 32nd Annual Conference on Computer Security Applications, 2016, pp. 508–519.
29. Blanchard, P.; El Mhamdi, E.M.; Guerraoui, R.; Stainer, J. Machine learning with adversaries: Byzantine tolerant gradient descent. *Advances in neural information processing systems* **2017**, *30*.
30. Fung, C.; Yoon, C.J.; Beschastnikh, I. Mitigating sybils in federated learning poisoning. *arXiv preprint arXiv:1808.04866* **2018**.
31. Zhang, Z.; Cao, X.; Jia, J.; Gong, N.Z. FLDetector: Defending federated learning against model poisoning attacks via detecting malicious clients. In Proceedings of the Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2022, pp. 2545–2555.
32. Liu, X.; Li, H.; Xu, G.; Chen, Z.; Huang, X.; Lu, R. Privacy-enhanced federated learning against poisoning adversaries. *IEEE Transactions on Information Forensics and Security* **2021**, *16*, 4574–4588.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2025 doi:10.20944/preprints202506.2218.v1

33. Awan, S.; Luo, B.; Li, F. Contra: Defending against poisoning attacks in federated learning. In Proceedings of the Computer Security–ESORICS 2021: 26th European Symposium on Research in Computer Security, Darmstadt, Germany, October 4–8, 2021, Proceedings, Part I 26. Springer, 2021, pp. 455–475.
34. Jebreel, N.M.; Domingo-Ferrer, J.; Sánchez, D.; Blanco-Justicia, A. Defending against the label-flipping attack in federated learning. *arXiv preprint arXiv:2207.01982* **2022**.
35. Li, D.; Wong, W.E.; Wang, W.; Yao, Y.; Chau, M. Detection and mitigation of label-flipping attacks in federated learning systems with KPCA and K-means. In Proceedings of the 2021 8th International Conference on Dependable Systems and Their Applications (DSA). IEEE, 2021, pp. 551–559.
36. Ma, C.; Li, J.; Ding, M.; Wei, K.; Chen, W.; Poor, H.V. Federated learning with unreliable clients: Performance analysis and mechanism design. *IEEE Internet of Things Journal* **2021**, *8*, 17308–17319.
37. Li, Z.; Sharma, V.; Mohanty, S.P. Preserving data privacy via federated learning: Challenges and solutions. *IEEE Consumer Electronics Magazine* **2020**, *9*, 8–16.
38. Jagielski, M.; Oprea, A.; Biggio, B.; Liu, C.; Nita-Rotaru, C.; Li, B. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In Proceedings of the 2018 IEEE symposium on security and privacy (SP). IEEE, 2018, pp. 19–35.
39. Yin, D.; Chen, Y.; Kannan, R.; Bartlett, P. Byzantine-robust distributed learning: Towards optimal statistical rates. In Proceedings of the International Conference on Machine Learning. PMLR, 2018, pp. 5650–5659.
40. Chang, H.; Shejwalkar, V.; Shokri, R.; Houmansadr, A. Cronus: Robust and heterogeneous collaborative learning with black-box knowledge transfer. *arXiv preprint arXiv:1912.11279* **2019**.
41. Nelson, B.; Barreno, M.; Chi, F.J.; Joseph, A.D.; Rubinstein, B.I.; Saini, U.; Sutton, C.; Tygar, J.D.; Xia, K. Exploiting machine learning to subvert your spam filter. *LEET* **2008**, *8*, 16–17.
42. Li, S.; Ngai, E.; Ye, F.; Voigt, T. Auto-weighted robust federated learning with corrupted data sources. *ACM Transactions on Intelligent Systems and Technology (TIST)* **2022**, *13*, 1–20.
43. Nguyen, T.D.; Rieger, P.; De Viti, R.; Chen, H.; Brandenburg, B.B.; Yalame, H.; Möllering, H.; Fereidooni, H.; Marchal, S.; Miettinen, M.; et al. {FLAME}: Taming backdoors in federated learning. In Proceedings of the 31st USENIX Security Symposium (USENIX Security 22), 2022, pp. 1415–1432.
44. Siegel, A.F. Robust regression using repeated medians. *Biometrika* **1982**, *69*, 242–244.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.