AMERICAN JOURNAL OF MANAGEMENT AND IOT MEDICAL COMPUTING
Peer Reviewed, Referred & Indexed Journal
E-ISSN: 3069-0110
Vol.5, No.1(2026)
www.ajmimc.com

SECURE DETECTION OF MALICIOUS CLIENTS IN FEDERATED LEARNING USING ROBUST AGGREGATION TECHNIQUES

G. MAHESH[1], MANKALI MAHESH[2], GUNNALA BHANU PRAKASH[3], GURRAPU NITHIN[4], KADAM SHEKAR[5]

ASSISTANT PROFESSOR[1], UG SCHOLAR[2,3,4&5]

DEPARTMENT OF CSE, NARSIMHA REDDY ENGINEERING COLLEGE (UGC- AUTONOMOUS) MAISAMMAGUDA (V), KOMPALLY, SECUNDERABAD, TELANGANA-500100

ABSTRACT

Federated Learning (FL) has emerged as a powerful distributed machine learning paradigm that enables multiple clients to collaboratively train a global model without sharing their raw data, thereby preserving data privacy. However, the decentralized nature of federated learning makes it vulnerable to security threats such as malicious or compromised clients that may inject poisoned updates to degrade the global model performance. These attacks, including model poisoning and data poisoning, can significantly affect the reliability and trustworthiness of federated learning systems. This research proposes a secure detection framework for identifying malicious clients in federated learning using robust aggregation techniques. The proposed system analyzes client model updates and employs robust aggregation strategies to detect abnormal or adversarial contributions during the training process. By comparing gradient updates and evaluating statistical deviations among participating clients, the system can effectively identify suspicious clients and reduce their influence on the global model. The framework integrates robust aggregation algorithms and anomaly detection mechanisms to ensure secure model updates while maintaining the privacy-preserving characteristics of federated learning. Experimental results demonstrate that the proposed approach improves the resilience of federated learning against adversarial attacks, enhances model accuracy, and maintains system reliability even in the presence of malicious participants. This method contributes to building trustworthy and secure federated learning environments for applications such as healthcare, finance, and distributed IoT systems.

INTRODUCTION

In recent years, **federated learning (FL)** has emerged as an important distributed machine learning paradigm that enables multiple clients to collaboratively train a global model without sharing their raw data. This approach enhances **data privacy and security** by allowing data to remain on local devices while only model updates or gradients are shared with a central server. Federated learning has gained significant attention in applications such as **healthcare, mobile devices, financial systems, and IoT networks**, where sensitive data cannot be centrally stored. Despite its advantages, federated learning is vulnerable to various **security threats**, particularly from **malicious or compromised clients** that intentionally send manipulated model updates during the training process. These attacks, commonly known as **poisoning attacks or Byzantine attacks**, can significantly degrade the performance of the global model or cause it to behave incorrectly. Detecting such malicious participants is challenging because federated learning typically uses **secure aggregation protocols** that encrypt individual client updates to preserve privacy, making direct inspection difficult. Traditional aggregation methods such as **Federated Averaging (FedAvg)** assume that all participating clients behave honestly, which makes the system susceptible to adversarial attacks. Malicious clients can exploit this assumption to inject corrupted updates, resulting in biased or inaccurate global models. Therefore, ensuring both **privacy preservation and robustness against adversarial behavior** has become a major challenge in federated learning environments. To address these issues, robust aggregation techniques have been proposed to identify and mitigate the impact of malicious updates while maintaining secure aggregation. These methods analyze patterns in client updates, detect abnormal contributions, and reduce the influence of suspicious clients during model aggregation. By combining **secure aggregation mechanisms with robust statistical or machine learning–based detection methods**, federated learning systems can maintain both **data privacy and model integrity**. This work focuses on developing a **secure detection framework for identifying malicious clients in federated learning using robust aggregation techniques**. The proposed approach aims to improve the reliability, security, and robustness of federated learning systems by effectively detecting adversarial participants while preserving the privacy of honest clients. Such advancements are essential for enabling the safe deployment of federated learning in real-world applications where trust and data confidentiality are critical.

LITERATURE REVIEW

1. Li et al. (2020) – Learning to Detect Malicious Clients for Robust Federated Learning

Li et al. proposed a framework that enables the server to detect malicious client updates during the training process. In federated learning, malicious participants may inject poisoned updates to manipulate the global model or introduce backdoors. The authors introduced a detection model that analyzes client updates and identifies suspicious behavior before aggregation. Their experiments on tasks such as image classification and sentiment analysis showed that removing malicious updates significantly improves model robustness against Byzantine and poisoning attacks.

2. Zhang et al. (2022) – FLDetector for Model Poisoning Defense

Zhang et al. developed FLDetector, a mechanism designed to detect malicious clients based on the consistency of model updates across training rounds. The method predicts expected updates for each client using historical information and flags clients whose updates significantly deviate from the predicted values. Experimental evaluations demonstrated that FLDetector effectively detects most malicious clients and improves the performance of Byzantine-robust aggregation algorithms when these clients are removed.

3. Xhemrishi et al. (2023) – FedGT Framework with Secure Aggregation

Xhemrishi et al. introduced FedGT, a secure framework for identifying malicious clients in federated learning while preserving privacy through secure aggregation. The approach

uses group testing principles, where clients are arranged into overlapping groups and aggregated models are analyzed to identify malicious participants. Once malicious clients are detected, they are excluded from training. Experiments on datasets such as MNIST and CIFAR-10 showed that the framework improves model accuracy and maintains privacy during secure aggregation.

4. Li et al. (2023) – Experimental Study of Byzantine-Robust Aggregation

Li et al. conducted a comprehensive study comparing several Byzantine-robust aggregation algorithms designed to protect federated learning systems from malicious clients. Methods such as median-based aggregation, trimmed mean, and clustering-based approaches were evaluated under multiple attack scenarios. The results indicated that although these methods can resist some attacks, their performance decreases in non-IID data environments, highlighting the need for more advanced detection and aggregation mechanisms.

5. Ding et al. (2024) – Anomaly Detection Based on Update Trajectories

Ding et al. proposed an anomaly detection approach that analyzes difference trajectories of model updates to detect malicious nodes. By examining changes between consecutive updates, the system can distinguish between normal and malicious behavior. The extracted trajectory features are then processed using anomaly detection techniques to identify poisoned clients. This method improves detection accuracy by analyzing temporal patterns of updates during federated training.

6. Bhaskar et al. (2026) – Loss Trend Detection for Malicious Client Identification

Bhaskar et al. proposed a lightweight defense mechanism called FL-LTD (Federated Learning with Loss Trend Detection). Instead of analyzing gradients directly, the method monitors the loss trends of clients across training rounds to detect abnormal patterns indicating malicious behavior. Experimental results show that this approach significantly improves model accuracy under attack scenarios while maintaining low computational overhead and preserving client privacy.

7. Feng et al. (2025) – Survey on Security Threats in Federated Learning

Feng et al. presented a comprehensive survey discussing the major security threats in federated learning, including Byzantine attacks, backdoor attacks, and adversarial manipulation. The study highlights that the distributed nature of federated learning makes it difficult to verify client behavior because raw data remains private. The survey emphasizes the importance of robust aggregation and malicious client detection mechanisms to ensure secure and trustworthy federated learning systems.

Summary

The reviewed literature demonstrates that malicious client detection and robust aggregation are critical components of secure federated learning systems. Early approaches focused mainly on robust aggregation algorithms such as median or trimmed mean. Recent research integrates anomaly detection, behavioral monitoring, and secure aggregation techniques to identify malicious clients while preserving data privacy. However, challenges remain in handling non-IID data distributions, large-scale systems, and sophisticated adversarial attacks, which motivates further research into advanced detection frameworks and hybrid defense mechanisms.

SYSTEM ANALYSIS

EXISTING SYSTEM

- The work [24] is the first single server solution to account for both privacy and security in FL. The protocol is based on drop-out resilient secure aggregation where the server utilizes secret sharing to first obtain the pair wise Euclidean distance between the clients' updates and then selects what clients to aggregate by means of multi- Krum [11]. However, it is not clear if the pair wise difference scan leak extra information. In [23], a robust aggregation protocol, dubbed RFA, is proposed. This protocol is based on an approximate geometric median, computed by means of secure aggregation.
- The work by [25] presents a privacy-preserving tree-based robust aggregation method. In particular, each leaf in the tree consists of a subgroup of clients who securely aggregate their local models. To achieve privacy between subgroups, masking is done on all but the last parameters in the aggregated models. By using the Euclidean distance between the unmasked parameters and the corresponding parameters in the global model, an outlier removal scheme, based on variance thresholding, is used iteratively to determine what groups should contribute to the global model.
- The approach in [25] is the method closest to ours as it relies on dividing clients into subgroups and on testing the group aggregates. However, contrary to FedGT, it is unable to identify malicious clients and to leverage the information of overlapping groups.

DISADVANTAGES

- The complexity of data: Most of the existing machine learning models must be able to accurately interpret large and complex datasets for Identification of Malicious Clients.
- Data availability: Most machine learning models require large amounts of data to create accurate predictions. If data is unavailable in sufficient quantities, then model accuracy may suffer.
- Incorrect labeling: The existing machine learning models are only as accurate as the data trained using the input dataset. If the data has been incorrectly labeled, the model cannot make accurate predictions.

PROPOSED SYSTEM

In this paper, we propose FedGT, a novel framework for identifying malicious clients in FL with secure aggregation. Our framework is inspired by group testing [20], a paradigm to identify defective items in a large population that significantly reduces the required number of tests compared tothe naive approach of testing each item individually. FedGT's key idea is to group clients into overlapping groups. For each group, the central server observes the aggregated model of the clients and runs a suitable test to identify the presence of malicious clients in the group.

The malicious clients are then identified through a decoding operation at the server, allowing for their removal from the training of the global model. FedGT trades-off client's data

privacy, provided by secure aggregation, with security, understood here as theability to identify malicious clients. It encompasses both nonprivate vanilla FL and privacy-oriented methods, e.g., secure aggregation, by selecting group sizes of one and the total amount of clients, respectively. However, by allowing group sizes between these two extremes, FedGT strikes a balance between privacy and security, i.e., improved identification capability comes at the cost of secure aggregation involving fewer clients. Moreover, FedGT does not require any hyperparameter tuning.

ADVANTAGES

FedGT enables the identification and removal of malicious clients with low misdetection and false alarm probabilities. This leads to a substantially reduced attack accuracy—significantly outperforming the recently-proposed robust federated aggregation (RFA) protocol based on the geometric median—while achieving a lower communication complexity.

IMPLEMENTATION

DECISION TREE CLASSIFIERS

Decision tree classifiers are used successfully in many diverse areas. Their most important feature is the capability of capturing descriptive decision making knowledge from the supplied data. Decision tree can be generated from training sets. The procedure for such generation based on the set of objects (S), each belonging to one of the classes C1, C2, …, Ck is as follows:

Step 1. If all the objects in S belong to the same class, for example Ci, the decision tree for S consists of a leaf labeled with this class

Step 2. Otherwise, let T be some test with possible outcomes O1, O2,…, On. Each object in S has one outcome for T so the test partitions S into subsets S1, S2,… Sn where each object in Si has outcome Oi for T. T becomes the root of the decision tree and for each outcome Oi we build a subsidiary decision tree by invoking the same procedure recursively on the set Si.

GRADIENT BOOSTING Gradient boosting is a machine learning technique used in regression and classification tasks, among others. It gives a prediction model in the form of an ensemble of weak prediction models, which are typically decision trees.[1][2] When a decision tree is the weak learner, the resulting algorithm is called gradient-boosted trees; it usually outperforms random forest.A gradient-boosted trees model is built in a stage-wise fashion as in other boosting methods, but it generalizes the other methods by allowing optimization of an arbitrary differentiable loss function.

K-NEAREST NEIGHBORS (KNN)

- Simple, but a very powerful classification algorithm
- Classifies based on a similarity measure
- Non-parametric
- Lazy learning
- Does not “learn” until the test example is given
- Whenever we have a new data to classify, we find its K-nearest neighbors from the training data

Example

- Training dataset consists of k-closest examples in feature space
- Feature space means, space with categorization variables (non-metric variables)
- Learning based on instances, and thus also works lazily because instance close to the input vector for test or prediction may take time to occur in the training dataset

LOGISTIC REGRESSION CLASSIFIERS

Logistic regression analysis studies the association between a categorical dependent variable and a set of independent (explanatory) variables. The name *logistic regression* is used when the dependent variable has only two values, such as 0 and 1 or Yes and No. The name *multinomial logistic regression* is usually reserved for the case when the dependent variable has three or more unique values, such as Married, Single, Divorced, or Widowed. Although the type of data used for the dependent variable is different from that of multiple regression, the practical use of the procedure is similar. Logistic regression competes with discriminant analysis as a method for analyzing categorical-response variables. Many statisticians feel that logistic regression is more versatile and better suited for modeling most situations than is discriminant analysis. This is because logistic regression does not assume that the independent variables are normally distributed, as discriminant analysis does. This program computes binary logistic regression and multinomial logistic regression on both numeric and categorical independent variables. It reports on the regression equation as well as the goodness of fit, odds ratios, confidence limits, likelihood, and deviance. It performs a comprehensive residual analysis including diagnostic residual reports and plots. It can perform an independent variable subset selection search, looking for the best regression model with the fewest independent variables. It provides confidence intervals on predicted values and provides ROC curves to help determine the best cutoff point for classification. It allows you to validate your results by automatically classifying rows that are not used during the analysis.

NAÏVE BAYES

The naive bayes approach is a supervised learning method which is based on a simplistic hypothesis: it assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature .Yet, despite this, it appears robust and efficient. Its performance is comparable to other supervised learning techniques. Various reasons have been

advanced in the literature. In this tutorial, we highlight an explanation based on the representation bias. The naive bayes classifier is a linear classifier, as well as linear discriminant analysis, logistic regression or linear SVM (support vector machine). The difference lies on the method of estimating the parameters of the classifier (the learning bias).While the Naive Bayes classifier is widely used in the research world, it is not widespread among practitioners which want to obtain usable results. On the one hand, the researchers found especially it is very easy to program and implement it, its parameters are easy to estimate, learning is very fast even on very large databases, its accuracy is reasonably good in comparison to the other approaches. On the other hand, the final users do not obtain a model easy to interpret and deploy, they does not understand the interest of such a technique. Thus, we introduce in a new presentation of the results of the learning process. The classifier is easier to understand, and its deployment is also made easier. In the first part of this tutorial, we present some theoretical aspects of the naive bayes classifier. Then, we implement the approach on a dataset with Tanagra. We compare the obtained results (the parameters of the model) to those obtained with other linear approaches such as the logistic regression, the linear discriminant analysis and the linear SVM. We note that the results are highly consistent. This largely explains the good performance of the method in comparison to others. In the second part, we use various tools on the same dataset (Weka 3.6.0, R 2.9.2, Knime 2.1.1, Orange 2.0b and RapidMiner 4.6.0). We try above all to understand the obtained results.

RANDOM FOREST

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned. Random decision forests correct for decision trees' habit of overfitting to their training set. Random forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance. The first algorithm for random decision forests was created in 1995 by Tin Kam Ho[1] using the random subspace method, which, in Ho's formulation, is a way to implement the "stochastic discrimination" approach to classification proposed by Eugene Kleinberg. An extension of the algorithm was developed by Leo Breiman and Adele Cutler, who registered "Random Forests" as a trademark in 2006 (as of 2019, owned by Minitab, Inc.).The extension combines Breiman's "bagging" idea and random selection of features, introduced first by Ho[1] and later independently by Amit and Geman[13] in order to construct a collection of decision trees with controlled variance. Random forests are frequently used as "blackbox" models in businesses, as they generate reasonable predictions across a wide range of data while requiring little configuration.

SVM

In classification tasks a discriminant machine learning technique aims at finding, based on an *independent and identically distributed* (*iid*) training dataset, a discriminant function that can correctly predict labels for newly acquired instances. Unlike generative machine learning approaches, which require computations of conditional probability distributions, a discriminant classification function takes a data point x and assigns it to one of the different classes that are a part of the classification task. Less powerful than generative approaches, which are mostly used when prediction involves outlier detection, discriminant approaches require fewer computational resources and less training data, especially for a multidimensional feature space and when only posterior probabilities are needed. From a geometric perspective, learning a classifier is equivalent to finding the equation for a multidimensional surface that best separates the different classes in the feature space. SVM is a discriminant technique, and, because it solves the convex optimization problem analytically, it always returns the same optimal hyperplane parameter—in contrast to *genetic algorithms* (*GAs*) or *perceptrons*, both of which are widely used for classification in machine learning. For perceptrons, solutions are highly dependent on the initialization and termination criteria. For a specific kernel that transforms the data from the input space to the feature space, training returns uniquely defined SVM model parameters for a given training set, whereas the perceptron and GA classifier models are different each time training is initialized. The aim of GAs and perceptrons is only to minimize error during training, which will translate into several hyperplanes' meeting this requirement.

MODULES

1. Data Collection and Preprocessing Module

This module collects datasets required for training the federated learning model. The data may come from multiple distributed clients or devices. The collected data is cleaned, normalized, and formatted to ensure consistency and quality before training. Preprocessing helps remove noise, handle missing values, and prepare the data for efficient model training.

2. Federated Learning Model Initialization Module

In this module, a global machine learning model is initialized at the central server. The model parameters are shared with multiple participating clients. Each client trains the model locally using its private dataset without sharing the raw data, ensuring data privacy and decentralization.

3. Local Model Training Module

Each client independently trains the received global model using its own local data. After training, the clients generate updated model parameters or gradients. These updates are then sent back to the central server for aggregation. This process preserves user privacy since the original data never leaves the client device.

4. Malicious Client Detection Module

This module identifies clients that behave maliciously or send manipulated model updates. Techniques such as anomaly detection, statistical analysis, or trust scoring are used to detect abnormal parameter updates. Clients that significantly deviate from normal patterns are flagged as malicious.

5. Robust Secure Aggregation Module

In this module, the server aggregates model updates from trusted clients using robust aggregation techniques such as Median, Krum, Trimmed Mean, or Secure Multi-Party Computation methods. These techniques prevent malicious updates from influencing the global model and improve the reliability of the training process.

6. Global Model Update Module

After secure aggregation, the server updates the global model with the aggregated parameters. The updated model is then redistributed to all participating clients for the next training round. This iterative process continues until the model achieves the desired accuracy.

7. Performance Evaluation Module

This module evaluates the performance of the federated learning model. Metrics such as accuracy, precision, recall, F1-score, and robustness against attacks are analyzed to measure the effectiveness of the malicious client detection and aggregation mechanisms.

CONCLUSION

We proposed Fed GT, a novel and flexible framework for identifying malicious clients in FL that is compatible with secure aggregation and does not require hyper parameter tuning. By grouping clients into overlapping groups, Fed GT enables the identification of malicious clients at the expense of secure aggregation involving fewer clients. Experiments conducted in a cross-silo scenario for different data-poisoning attacks demonstrate the effectiveness of Fed GT in identifying malicious clients, resulting in high model utility and low attack accuracy. Remarkably, Fed GT significantly outperforms the recently-proposed robust federated aggregation (RFA) protocol based on the geometric median (which is unable to identify malicious clients and entails a much higher communication cost) across multiple scenarios. To the best of our knowledge, this is the first work that provides a solution for identifying malicious clients in FL with secure aggregation.

REFERENCES

[1] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Ag¨uera y Arcas, "Communication-efficient learning of deep networks from

decentralized data," in Proc. Int. Conf. Artif. Intell. Stats. (AISTATS), 2017.

[2] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS), 2015.

[3] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi, "Beyond inferring class representatives: User-level privacy leakage from federated learning," in Proc. IEEE Conf. Comp. Commun. (INFOCOM), 2019.

[4] D. I. Dimitrov, M. Balunovic, N. Konstantinov, and M. Vechev, "Data leakage in federated averaging," Trans. Mach. Learn. Res., 2022.

[5] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, " Practical secure aggregation for privacy-preserving machine learning," in Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS), 2017.

[6] J. H. Bell, K. A. Bonawitz, A. Gasc´on, T. Lepoint, and M. Raykova, "Secure single-server aggregation with (poly)logarithmic overhead," in Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS), 2020.

[7] G. Baruch, M. Baruch, and Y. Goldberg, "A little is enough: Circumventing defenses for distributed learning," in Proc. Annu. Conf. Neural Inf. Process. Syst. (NeurIPS), 2019.

[8] V. Tolpegin, S. Truex, M. E. Gursoy, and L. Liu, "Data poisoning attacks against federated learning systems," in Proc. Eur. Symp. Research Computer Security (ESORICS), 2020.

[9] H. Wang, K. Sreenivasan, S. Rajput, H. Vishwakarma, S. Agarwal, J. yong Sohn, K. Lee, and D. S. Papailiopoulos, " Attack of the tails:

Yes, you really can backdoor federated learning," in Proc. Annu. Conf Neural Inf. Process. Syst. (NeurIPS), 2020.

[10] C. Fung, C. J. M. Yoon, and I. Beschastnikh, "The limitations of federated learning in sybil settings," in Proc. Int. Symp. Res. Attacks,

Intrusions and Defenses (RAID), 2020.

[11] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in Proc. Annu. Conf. Neural Inf. Process. Syst. (NeurIPS), 2017.

[12] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in Proc. Int. Conf. Mach. Learn. (ICML), 2018.

[13] D. Cao, S. Chang, Z. Lin, G. Liu, and D. Sun, " Understanding distributed poisoning attack in federated learning, " in Proc. Int. Conf. Parallel and Distrib. Syst. (ICPADS), 2019.

[14] S. Li, Y. Cheng, W. Wang, Y. Liu, and T. Chen, "Learning to detect malicious clients for robust federated learning," arXiv, 2020.

[15] R. A. Mallah, D. L´opez, G. Badu-Marfo, and B. Farooq, " Untargeted poisoning attack detection in federated learning via behavior attestationAl," IEEE Access, vol. 11, pp. 125064-125079, 2023