Edge-based Federated Learning to Enhance the Security of IoT

Zahra Eskandari[†], Non-member

ABSTRACT

The integration of Artificial Intelligence (AI) and Internet of Things (IoT) has enabled network components to make independent decisions, but this capability also poses a risk of malicious attacks in unsupervised and trustless environments. To overcome this issue, the article proposes a distributed collaborative detection approach that utilizes edge nodes as voters to monitor the training process, tackle PA, and improve the accuracy of the global model. The proposed approach is evaluated using the UNSW-NB15 dataset in both IID and non-IID scenarios, and the results demonstrate the effectiveness of the approach in improving the accuracy of FL even in the presence of PA.

Keywords: Artificial Intelligence of Things, Edge Computing, Federated Learning, Poisoning Attack, UNSW-NB15 Dataset

1. INTRODUCTION

As the Internet of Things (IoT) develops rapidly, it is having a significant impact on various domains, such as healthcare, education, industry, agriculture, and more, with a human-centric focus. With the growth of Artificial Intelligence (AI) discussions, concepts such as Artificial Intelligence of Things (AIoT) have emerged in recent years [1]. AIoT refers to the integration of AI with IoT, where the components of the network can make decisions based on Machine Learning (ML). As a result, IoT is moving towards a more distributed approach, where data and decisions become increasingly independent of human intervention. However, this capability may be used for malicious purposes in unsupervised and trustless environments. The Mirai botnet in 2016 [2] is an example of such an attack in the IoT landscape. Using vulnerabilities in IoT devices, Mirai created a large-scale botnet capable of launching DDoS attacks.

Numerous approaches have been suggested to identify and thwart malicious attacks on networks. A key solution is the implementation of intrusion detection systems (IDS) [3]. In many of these approaches, ML is employed for traffic analysis and attack detection [4-16]. Conventional ML methods require gathering all network traffic data in the cloud and training a model to recognize attacks using this data [4-8]. While these centralized methods yield excellent results, they face significant challenges that hinder their adoption in IoT applications. Transferring data from devices to servers, coupled with overloading cloud infrastructure, leads to security and privacy concerns, and many data owners are reluctant to share their data for ML model training. To address these issues, some studies [9-16] have introduced decentralized techniques, such as federated learning (FL) [17], which enable model training without the need for data sharing.

Unlike centralized approaches, FL shares the model instead of raw data on the network to protect user privacy [11-14,16]. With the rise of edge computing, which brings cloud computing, processing, and storage to the network's edge, the discussion around FL has become more prominent. However, the potential of malicious activities in unsupervised and trustless environments poses a significant concern for FL in IoT. As FL relies on a central server to aggregate gradient values, it becomes an easier target for malicious attacks. Therefore, it is crucial to implement robust security measures to safeguard data privacy and prevent attacks. One of these vulnerabilities is a Poisoning Attack (PA) [18], where compromised nodes provide inaccurate or low-quality data, leading to a decrease in the accuracy of the global model. To tackle this issue, centralized and distributed methods have been proposed to detect and prevent such attacks [19-34].

To address this issue, we propose a distributed collaborative approach that utilizes edge nodes as voters to monitor the training process. The voters evaluate the local models received from clients and identify poisoned models. Then, the final decision is made by the server based on the majority of the votes. As a result of this distributed monitoring, the accuracy of the global model is improved even in the presence of PA. We evaluate the proposed detection approach using the UNSW-NB15 dataset [37, 38], which includes various attacks and has been widely used in previous studies [16, 41]. It is important to note that FL is inherently non-independent and identically distributed (non-IID), however, most studies have focused on the IID scenario, with less exploration of non-IID scenarios. In this work, we study the impact of PA on the accuracy of FL in both IID and non-IID scenarios, considering the presence and absence of detection. The main contributions of this paper are summarized as follows:

Manuscript received on January 25, 2024; revised on April 11, 2024; accepted on April 30, 2024. This paper was recommended by Associate Editor Siraporn Sakphrom.

The author is with Department of Computer Engineering, Faculty of Electrical and Computer Engineering, Quchan University of Technology, Quchan, Iran.

[†]Corresponding author: z.eskandari@qiet.ac.ir

Digital Object Identifier: 10.37936/ecti-eec.2024222.252524

1. To ensure data privacy in IoT, we have highlighted the significance of edge nodes in FL for detecting attacks.
2. By collaborating with edge nodes, we address one of the vulnerabilities in FL by detecting and removing poisoned models from the training process. This results in improved accuracy and faster convergence.
3. We evaluate our approach using the UNSW-NB15 dataset in both IID and non-IID scenarios.

Structure of the work: The rest of this paper is organized as follows: A review of initial preliminaries, including FL to secure IoT, PA and defense strategies, follows in section 2. The system model and the proposed detection approach to defeat PA are discussed in Section 3, and experimental results are presented in Section 4. Finally, Section 5 concludes the paper with a summary of the findings and suggestions for future work.

2. PRELIMINARIES

This section reviews methods utilizing FL to improve security in IoT networks. It then discusses a major vulnerability of FL - PA - and countermeasures that have been proposed to address this vulnerability.

2.1 FL to secure IoT

Previous centralized approaches [4-8] for detecting and defeating attacks in IoT networks involve aggregating all network traffic data in the server and training a model on this data to identify attacks. Despite the high accuracy, these methods require collecting all training data on a single machine, which is expensive in terms of time and computing resources. Furthermore, transferring and storing data samples from devices on a central server raises security and privacy concerns, as many data owners are unwilling to share their data to train a ML model. To address these issues, decentralized techniques like federated learning (FL) have been applied to train models without sharing the underlying data. FL allows collaborative model training across distributed devices without centralizing the data.

In recent years, FL has been applied to anomaly detection in IoT networks. In 2019, a federated self-learning approach [9] achieved a 95.6% detection rate for IoT networks. In 2020, a federated deep Gaussian mixture model [10] was proposed for anomaly detection and outperformed centralized models on the KDDCUP 99 dataset in some cases. More recently, in 2021, a federated deep learning model combining CNN and GRU [11] and an encoder-decoder LSTM approach [12] were presented for anomaly detection in industrial IoT data. Both studies demonstrated FL models can achieve performance close to centralized versions.

In 2022, several studies explored FL for attack detection in IoT networks and compared it to centralized ML. A FL framework was proposed in [13] that reduced computational and data transfer costs while achieving comparable performance to a centralized version. [14] presented supervised and unsupervised federated learning approaches for malware detection in IoT devices, demonstrating improved performance over multiple local models. [15] proposed a blockchain-based federated learning approach with generative adversarial networks for anomaly detection, demonstrating robustness, accuracy, and fast convergence on real-world data. Finally, [16] introduced a two-stage distributed architecture called FedGame that uses FL to allow multiple edge nodes to build an efficient attack detection model for IIoT networks collaboratively. Game theory is then used for the edge nodes to compete for more virtual resources. Overall, these recent works highlight the advantages of FL for enabling collaborative and privacy-preserving attack detection in IoT.

2.2 Poisoning Attack as a vulnerability of FL

A critical vulnerability of FL is the lack of access to raw training data. Participant devices or edge nodes may intentionally or unintentionally submit erroneous model updates that reduce the accuracy of the global model. This type of attack is known as a poisoning attack (PA) [18]. By aggregating these poisoned models, errors are introduced into the global model. Based on the attacker's intent, poisoning attacks can be targeted, causing the FL model to output a specific incorrect target label for chosen samples. Or the attacks can be untargeted, causing the model to misclassify all testing samples indiscriminately. Examples of targeted and untargeted attacks are backdoor attacks [35] and label-flipping attacks [36], respectively. PA poses a sever threat to FL and must be addressed through robust aggregation methods and other defense strategies.

To the best of our knowledge, the current state-of-the-art countermeasures against PA in FL can be discussed as follows: One approach to address this issue on the server side is to cluster the received models [19]. Another server-side approach involves checking the received local models using auditing or generated datasets [20]. Some works have also focused on performing aggregation with robust techniques to minimize the effect of poisonous models [21, 22]. These techniques aim to ensure that the aggregated model is not significantly influenced by the presence of poisoned models. However, it is important to note that for the majority of server-side validation methods, auditing or generated datasets are employed to ensure the quality of the received local models. These approaches assume that the validation datasets share a similar distribution to the local datasets. However, in FL, the non-IID property makes this approach impractical. Additionally, the local training datasets are inaccessible to central servers, assuming to having a validation dataset with similar distribution, unrealistic.

There are several distributed approaches to address PA in FL. These include exchanging model parameters only with neighbors instead of a central server [23], gossip-based methods over a peer-to-peer network [24-26], collaboration between several servers over a fully connected network [27], peer-to-peer Bayesian-like approach [28, 29], consensus-driven federated learning [30,

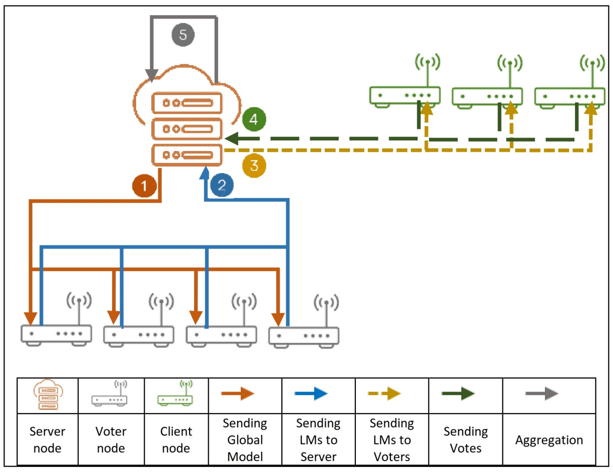

***Fig. 1**: Proposed Edge-based FL approach. The procedure consists of: 1) sending the global model to the clients by the server, 2) sending local models to the server by the clients, 3) sending local models to the voters by the server, 5) sending votes to the server by the voters, 4) decision-making by the server for updating the global model.*

31], cross-cluster blockchain-based federated learning [32], blockchain-enabled federated learning scheme [33], and using random clients' datasets to audit local models [34]. Each of these approaches considers different performance parameters such as load balancing, privacy issues, latency, communication, consensus delays, and computation cost.

3. SYSTEM MODEL

To secure IoT, edge nodes, such as routers, can analyze the traffic of connected devices and detect attacks collaboratively with other participating edge nodes in Federated Learning (FL). This approach allows the model to be trained without sharing raw data while respecting users' privacy. In traditional FL, the local models were extracted and sent to the server for aggregation, then centralized methods relying on the server were used for detecting PA. To address this issue in a decentralized manner, which is particularly helpful in non-IID scenarios, as shown in Figure 1, the proposed approach involves the cooperation of edge nodes as voters. They examine local models and send their votes to the server for decision-making about updating the global model. The aim of these two steps, distributed evaluation of local models and decision-making by the server, is to detect and defeat malicious clients, resulting in an improvement in the learning process, an increase in model accuracy, and faster learning.

3.1 Attack model:

To perform PA, we focus on Label flipping [36], where malicious participants flip the labels of data in local datasets. Assuming a normal dataset including N samples as:

$[(x_1.y_1).(x_2.y_2).(x_3.y_3)\dots.(x_N.y_N)]$, where $x_i|i = 1\dots.N$ is the sample, and $y_i|i = 1\dots.N$ is the corresponding label. However, after label flipping, y_i becomes y_i', where y_i' could be any other label based on the probability of PA. Due to privacy concerns, the server cannot access the participants' local datasets, so the validity of these datasets could not be verified. As a result, malicious participants use the label-flipped datasets for model training and upload the poisoned extracted model to the server to launch the data poisoning attack. This attack deteriorates the accuracy of the global model and causes the failure of the convergence of the global model.

3.2 Edge-based federated learning

This section provides a detailed description of the proposed Edge-based federated learning approach. Algorithm 1 outlines the steps taken by the server, while Algorithms 2 and 3 illustrate the steps executed by clients and voters, respectively. Table 1 describes the notations used in the algorithms.

In the first line of Algorithm 1, the *System_initilazation* method sets the security parameters to provide security measurements such as privacy, integration, resiliency to replay attacks, and leakage prevention. Every entity in the network, including devices and edge nodes, must authenticate as legitimate entities to participate in FL by a global Trust Authority (TA). Upon successful authentication, the entity obtains its public and private keys, as well as the corresponding certificates for signing, encryption, and decryption. To ensure confidentiality, replay attack avoidance, and integration, the sender encrypts the message with the receiver's public key, adds a timestamp, and signs the message with its private key. When broadcasting messages to voters, signing and adding a timestamp are considered. Additionally, to prevent information leakage, local models are sent in an anonymized format, as sending the raw models may reveal information about the local datasets. The article does not delve into the specific details of these security algorithms, but it suggests that standard anonymity approaches, signature algorithms, and asymmetric cryptography can be adopted for system initialization [39, 40].

After setting up the security parameters (line 1), the global model is initiated along with other algorithm parameters (lines 2-4). The training rounds are then started (line 5). At the beginning of each training round r, the server sends GM_{r-1} to the nodes in the network (line 6), and some of the nodes are selected as clients to share their local model (line 7). After receiving GM_{r-1}, the clients train their local model with their local dataset and send the extracted local model to the server for aggregation (Algorithm 2).

On the server side, after collecting the received local models (line 10), the server sends the set of received local models to voters for evaluation to overcome PA (line 11). As mentioned earlier, by incorporating the two-step process of evaluating local models by voters and decision-making by the server based on the opinions received from the voters, the detection of malicious nodes can be improved.

```
    $Algorithm1: Proposed\ Edge - bsed\ FL$
    $executed\ at\ the\ server$

    $Inputs: N.Voters.Acc_{Max}$
    $Output: GM_r$

1   $System\_initilazation\ ();$
2   $GM_0 = Init\_Model();$
3   $Acc_{value} = calclulate\_Accuracy(GM_0)$
4   $r = 1;$
5   $while(Acc_{value} < Acc_{Max})$
6       $Broadcast(GM_{r-1})$
7       $Clients = Client\_selection(N)$
8       $for\ each\ client\ i\ in\ Clients:$
9           $LM_i^r = Train\_local\_model(GM_{r-1})$
10                  $LM^r.append(LM_i^r)$
11      $Send\_to\_Voters(LM^r)$

12      $for\ each\ voter\ j\ in\ Voters:$
13          $vote_j^r = Evaluation(GM_{r-1}, LM^r)$
14          $Votes.append(vote_j^r)$
15      $if\ IID - scenario:$
16          $for\ each\ LM_i^r\ in\ LM^r:$
17              $if\ majority\_votes(Votes, LM_i^r):$
18                  $P = P \cup LM_i^r$
19          $GM^r = Aggregate(\{LM_i^r \mid LM_i^r ! \in P\})$
20          $Acc_{value} = calclulate\_Accuracy(GM_r)$
21      $elif\ Non - IID - scenario:$
22              $if\ majority\_votes(Votes):$
23                  $GM^r = Aggregate(\{LM_i^r \mid LM_i^r \in LM^r\})$
24                  $Acc_{value} = calclulate\_Accuracy(GM_r)$
25      $r = r + 1;$
```

```
$Algorithm2: Train\_local\_model()$
$executed\ at\ the\ client\ i$

$Inputs: GM_{r-1}$
$Output: LM_i^r$

$LM_i^r = Local\_training(GM_{r-1})$
$return\ (LM_i^r)$
```

```
$Algorithm3: Evaluation()$
$executed\ at\ the\ voter\ j$

$Inputs: GM_{r-1}.LM^r$
$Output:$ vote

$if\ IID - scenario:$
        $Global\_Acc\_value = calclulate\_Accuracy(GM_{r-1})$
        $for\ each\ LM_i^r\ in\ LM^r:$
            $Local\_Acc\_value = calclulate\_Accuracy(LM_i^r)$
            $If\ Local\_Acc\_value < Global\_Acc\_value :$
                            $vote[i] = 1$
            $else:$
                            $vote[i] = 0$
$elif\ Non - IID - scenario:$
        $Acc_{Global} = calclulate\_Accuracy(GM_r)$
        $TM = Aggregate(\{LM_i^r \mid LM_i^r \in LM^r\})$
        $Acc_{Temp} = calclulate\_Accuracy(TM)$
        $if\ Acc_{Global} < Acc_{Temp}:$
                $vote[i] = 0$
        $else:$
                $vote[i] = 1$
$return\ vote$
```

To perform these steps in the IID scenario, as described in Algorithm 3, each voter evaluates the global model, GM_{r-1}, and each local model in the LM set by its dataset. If the accuracy of the global model is improved by the local model i, the local model LM_i^r is considered non-poisonous. Otherwise, the voter votes for the model i as a poisoned model. After evaluating all local models, the voter sends its votes to the server. On server-side, after collecting the votes from all voters, the server investigates the votes for each local model (line 16). If the majority of the voters have evaluated a local model as a poisoned model, it is inserted into the poisoned clients set, P (lines 17-18). After that, all local models not in the P set are selected for aggregation to calculate the next global model, GM_r (lines 19-20).

***Table 1**: Notations.*

Notation	Definition
$N = \{N_i \mid i = 1 \cdots N\}$	Set of clients
$Voters = \{V\_j \mid j = 1 \ldots M\}$	Set of voters
$LM = \{LM_i^r \mid i = 1 \cdots N\}$	Set of local models at round r for clients $i = 1 \cdots N$
P	Set of poisoned clients
$GM = \{GM^r \mid r = 0 \cdots R\}$	Set of global models at rounds $r = 0 \cdots R$
Acc_{Max}	Maximum required Accuracy
r	Round number

In the non-IID scenario, clients do not have access to samples of all available classes in the dataset and only learn from a subset of classes, which leads to low accuracy in evaluating the validation dataset, including all classes. Therefore, it is difficult to determine whether the low accuracy obtained is due to this reason or as a result of a PA. For this reason, in this type of scenario, instead of individually evaluating local models, as shown in Algorithm 3, voters aggregate the received local models into a temporary model TM, and the accuracy of the temporary model and the global model is calculated with their dataset. If the temporary model increases the accuracy, the voter reports its positive opinion to the server. Otherwise, it disagrees with the aggregating local models and updating the global model in this round and reports its negative opinion to the server.

After receiving voters' opinions, the server investigates the votes (line 22). If more than half of the voters agree with aggregating and updating the global model, these operations will be performed; otherwise, the round will be repeated without updating the global model (lines 22-24). These procedures are repeated until a maximum accuracy, Acc_{Max}, is achieved.

It should be mentioned that for easy understanding and implementation, the voter selection strategy is considered simply as a random selection in this article. Although the strategy is simple, it can effectively detect lazy and malicious clients. Moreover, by considering the possibility of compromised voter nodes, it is more accurate to design a more sophisticated voter selection strategy based on the reputation rate, contribution rate, or even the size of their dataset. This highlights an area for future work.

4. PERFORMANCE EVALUATION

In this section, we evaluate the performance of the proposed approach to mitigate PA in FL. Firstly, we explain the dataset and the experiment setup. Then, the numerical results are presented in two main scenarios:

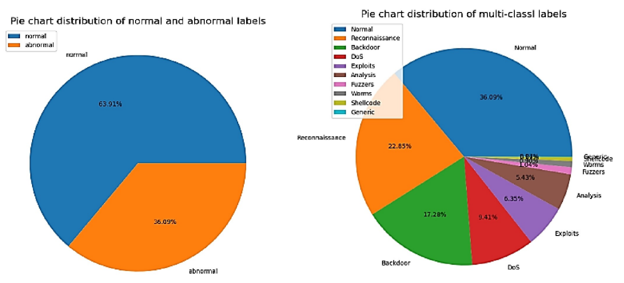

Fig. 2: Pie Charts for distribution of 2-class and 10-class labels.

IID and non-IID.

4.1 Dataset description

We evaluate the proposed approach using the UNSW-NB15 dataset [37, 38], which contains a total of 257573 records, including some records for normal behavior and a variety of IoT attacks, divided as follows: analysis, fuzzers, DDoS, backdoors, reconnaissance, generic, exploits, shellcode, and worms. The Distribution of attacks in the dataset is shown in Figure 2. For the CL experiments, 70% and 30% of the data is considered as the train and test dataset, respectively. For FL experiments, 70% of the data is divided equally among all nodes in the network, and the remaining 30% is designated for voters. However, this division is not necessary, and all the data can be equally divided among all nodes, this approach is adopted here for ease of implementation.

4.2 Experiment setup

In the scenarios, according to [42-48], a Long Short-Term Memory (LSTM) Neural Network model was chosen as a binary classifier to detect Normal and Attack labels, while a Deep Neural Network model was selected as a categorical classifier to detect Normal and each type of Attack. The LSTM model, considering 2-class labels, consists of a single neuron in the output layer with a sigmoid activation function and Binary Cross-Entropy loss function, using the Adam optimizer. The number of hidden layers in the model was carefully selected to account for the heterogeneity of edge nodes and the delay in local model training, aiming to minimize complexity while maintaining an acceptable level of accuracy. This goal is gained by selection one hidden layer.

On the other hand, the DNN model utilizes Convolution1D, MaxPooling1D and *BatchNormalization* layers to capture local patterns, prevent overfitting, and enhance model stability, respectively. Two Bidirectional LSTM layers are then employed to account for the sequential nature of the data, followed by a Dropout layer to mitigate overfitting. Finally, a Dense layer with 10 neurons and a *softmax* activation function is used for the 10-class classification, corresponding to the classes (normal and 9 types of attacks) under the categorical cross-entropy loss function and the Adam optimizer. The

***Table 2**: Parameters of LSTM NN.*

Parameter	value
Number Of Hidden Layers	1
Loss Function	Binary Cross-Entropy
Optimizer Gradient	Adam
Activation Function	Sigmoid
Output Layer Size	1

***Table 3**: Parameters of Deep NN.*

Parameter	value
Number Of Hidden Layers	7
Loss Function	Categorical Cross-Entropy
Optimizer Gradient	Adam
Activation Function	SoftMax
Output Layer Size	10

***Table 4**: Simulation Parameters.*

Parameter	value	Description
N	25, 50	Number of Clients Candidates
C	5, 10, 12, 25	Number of Clients
M	5, 10	Number of Voters
Epochs_CL	25	Number of Epochs in CL
R_FL	25	Number of Training Rounds in FL
Epochs_FL_Local	5	Number of Epochs in Local model Training
α	[0.3,0.5,0.8]	Fraction of Poisoned Data
β	[0.3,0.5,0.8]	Fraction of Poisoned clients
ClassesPerClient	[3,4,5,6,10]	Number of Classes in each client

details of these classification models are presented in Table 2 and Table 3, respectively. The effectiveness of both classification models is evaluated based on the accuracy metric, with the goal of maximizing it.

In the experiments, the label flipping attack is evaluated under two parameters: α as the fraction of poisoned data of each participant and β as the fraction of malicious clients. Each poisonous participant clones α fraction of the local dataset with flipped labels for injecting poisonous data into training. The α and β are varied from 0.3 to 0.8. Table 4 provides details about the simulation parameters.

It should be mentioned that all experiments were carried out on a Windows 10 machine with a 13th

***Table 5**: Accuracy of the 2-class model in different values of α and β in Non-PA-Detection and PA-Detection conditions.*

Accuracy		α					
		0.3		0.5		0.8	
		Non PA-D	PA-D	Non PA-D	PA-D	Non PA-D	PA-D
β	0.3	91.05%	91.08%	84.13%	90.87%	83.51%	90.43%
	0.5	90.35%	90.69%	82.01%	90.26%	46.24%	90.60%
	0.8	90.08%	90.39%	70.30%	90.39%	10.33%	90.13%

Gen Intel(R) Core (TM) i7-13700K 3.40 GHz CPU, Asus GeForce RTX 4060 TI TUF OC GPU, and 32 GB RAM, using Python 3.9.17 with TensorFlow 2.10.1.

4.3 Numerical Results

Here in two separate scenarios, we evaluate the proposed approach to mitigate PA in FL.

IID scenario: Here, the goal of FL is to train a two-class model to distinguish normal and attack traffic. The LSTM NN presented in Table 2 is used in the server and clients, and Algorithms 1 to 3 are executed in the corresponding nodes. Without PA, FL achieves an accuracy of 91.20% after 25 rounds, comparable to the CL approach with an accuracy of 92.33% after 25 epochs.

To evaluate the proposed approach, PA is conducted under different values of parameters α and β, and the accuracy of the global model after 25 rounds is presented in Table 5. It can be observed that in the event of PA, under the non-detection condition (Non PA-D), the accuracy significantly decreases with an increase in the values of parameters α and β. However, applying the proposed detection approach (PA-D) leads to a significant improvement in the accuracy of the global model, indicating the successful identification of poisonous clients through the evaluation in the voters and their exclusion from the aggregation process. For lower values of parameter α, even with an increase in parameter β, the detrimental effect on the accuracy is insignificant, suggesting that the proportion of poisoned data in the clients is more influential in the attack process. However, with a simultaneous increase in both parameters, there is a severe drop in the accuracy. It should be noted that 10 voters supervise the clients in these experiments.

To further evaluation, considering the goal of FL to train a model not only to distinguish normal and attack traffic but also to detect the type of attacks, the Deep NN proposed in Table 3 is utilized in the server and clients, and Algorithms 1 to 3 are executed in the corresponding nodes. As shown in Figure 1, due to the imbalance of the dataset in some attack types, the accuracy of the CL after 25 epochs of training reached 81.42%, so it is not far from the expectation that FL in this case is more complicated.

Thus, here we study the impact of the size of local datasets and the number of participating clients on the accuracy of FL and the results are presented in Figure 3 with different values for N and C. it is observed that although in N=50, C=25 and N=25, C=12 cases, almost

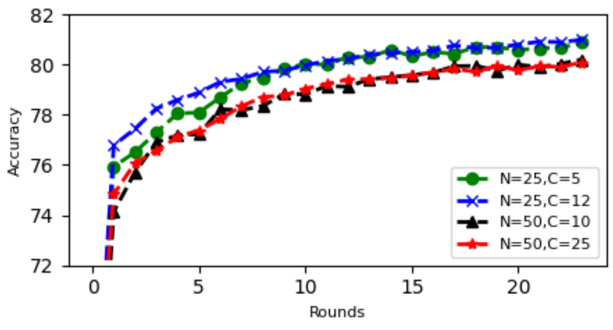

***Fig. 3**: Accuracy of FL in different values of N and C.*

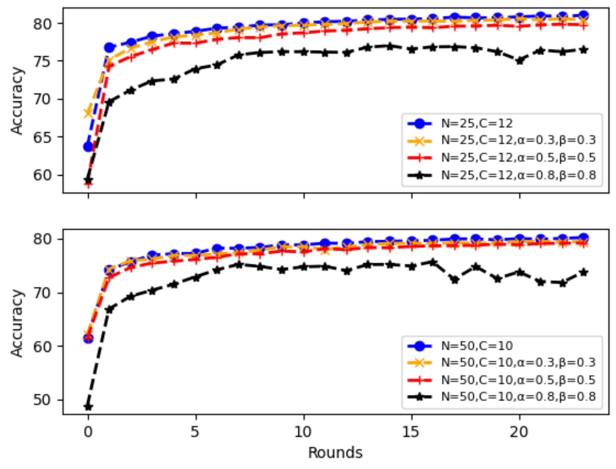

***Fig. 4**: Accuracy of FL under PA in different values of α and β.*

equal amounts of data from the overall dataset are used for training, the size of the local dataset of each client has a more significant impact on the accuracy of local models and, consequently, the accuracy of the global model. As shown in Figure 3, under N=25 and C=12, which is the best of the studied cases, FL achieves an accuracy of 81% after 25 rounds.

In the following, PA is conducted for different values of N and C. Similar to the 2-class model, it is observed that in the event of PA, the accuracy decreases with an increase in the values of parameters α and β. As shown in Figure 4, in the worst reduction case, for α=β=0.8, the accuracy decreases from 81% to 74.99% in the N=25 and C=12 case, and in the N=50 and C=10 case, it falls from 80.24% to 73.84%.

In continue, to investigate the effectiveness of the proposed detection approach, Figure 5 shows the accuracy of FL in the presence of a PA with detection in the N=50, C=25 and N=25, C=12 cases. As shown in Figure 5, the proposed approach successfully neutralizes the effect of PA and improves the accuracy of the global model. It should be noted that the number of voters monitoring clients in these experiments is 5 voters.

Non-IID scenario: FL becomes more challenging due to the inherent complexities of non-IID scenario. One

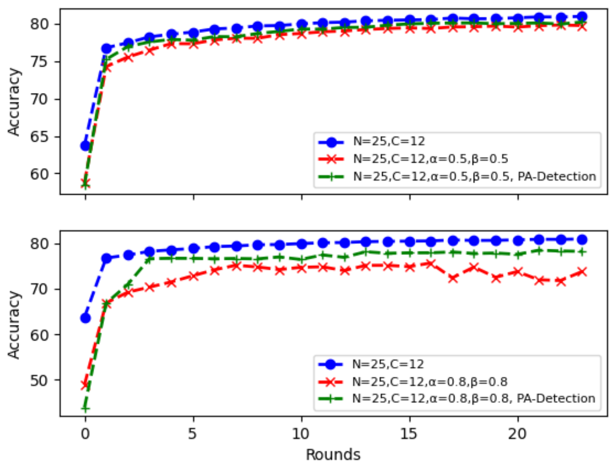

***Fig. 5**: Accuracy of IID scenario FL in different values of α and β in Non-Detection and PA-Detection conditions.*

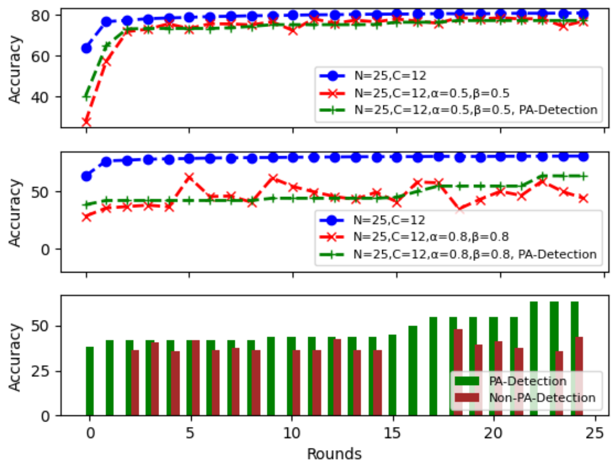

***Fig. 6**: Accuracy of non-IID scenario FL in different values of α and β in Non-Detection and PA-Detection conditions.*

***Table 6**: Accuracy of the 10-class model in different values of N,C and ClassPerClient parameters.*

Accuracy		ClassPerClient				
		3	4	5	6	10
N	C					
25	12	71.01%	77.48%	77.710%	80.33%	81%
50	10	59.74%	76.08%	76.91%	76.21%	80%
50	25	75.36%	75.67%	78.78%	78.27%	80.01%

issue to consider is that in this scenario, clients not only lack access to all samples of existing classes in the dataset but also may have different classes. As a result, the model's accuracy decreases as it relies on learning from a sufficient and balanced number of samples from all classes. This is evident in Table 6, where an increase in the number of available classes for each client leads to a rise in accuracy. Additionally, as shown in Table 6, by learning under different values of N and C, the accuracy is influenced by the size of the clients' local datasets.

To evaluate the effectiveness of the proposed detection approach in mitigating PA in the non-IID scenario, we conduct experiments with N=25, C=12, and ClassesPerClient =6, applying PA to the participants with different values for parameters α and β. It is important to note that 5 voters are used to evaluate the local models. As presented in Figure 6, similar to the IID scenario, applying PA to the participants without any detection, results in a decrease in accuracy, with a sharper decline in larger values of parameters α and β. As shown in Figure 6, in the case of α=β=0.5, the accuracy of the global model under PA fluctuates but ultimately remains close to the original accuracy. However, with larger values, α=β=0.8, the reduction in the accuracy worsens, and in the non-detection condition, the accuracy fluctuates and eventually decreases to 43.82% after 25 rounds of training. By applying the proposed detection approach, the accuracy improves and a higher percentage, 63.68%, is achieved in 25 rounds of training.

The bar chart in Figure 6 shows that if the server aggregates local models without considering the votes examined by the distributed voters, the accuracy of the global model will definitely decrease. This highlights the successful cooperation and effectiveness of the distributed voters in overcoming PA. It is important to note that by examining Figures 5-7 regarding the efficiency and effectiveness of the proposed approach, it can be concluded that despite the distribution in the proposed approach to neutralize the PA, this level of distribution works well in the IID scenario. However, for the non-IID scenario, solutions with a higher level of distribution are needed. This indicates a potential area for future work.

5. CONCLUSION AND FUTURE WORKS

In this article, considering the importance of security in Internet of Things (IoT) networks and the need to protect users' privacy, a collaborative federated Learning approach was proposed to detect attacks. However, due to the inherent characteristics of Federated Learning in the lack of access to raw network data, attacks such as Poisoning Attacks on Federated Learning are inevitable. Due to the importance of the issue, in this work, a distributed approach based on edge nodes was proposed to detect these attacks, and with a focus on the accuracy of the model as the most important parameter in Federated Learning, the proposed approach was evaluated in IID and non-IID scenarios. The results indicate that the proposed detection approach has performed well in detecting Poisoning Attacks and has improved the accuracy of the global model even in the presence of such attacks. However, it is worth mentioning that the performance of the proposed detection method, despite being distributed, has been more successful in IID scenario than in non-IID scenario. As an open problem in this field, a more distributed solution will be needed to improve detection in non-IID scenario in more successful and efficient manner. This indicates a potential area for future work.

REFERENCES

[1] Y. Jin et al., ”Edge-Based Collaborative Training System for Artificial Intelligence-of-Things,” *IEEE Transactions on Industrial Informatics*, vol. 18, no. 10, pp. 7162-7173, Oct. 2022.

[2] M. Antonakakis et al., ”Understanding the mirai botnet,” in *26th USENIX security symposium (USENIX Security 17)*, pp. 1093-1110, 2017.

[3] M. Baich, T. Hamim, N. Sael, and Y. Chemlal, “Machine Learning for IoT based networks intrusion detection: a comparative study,” *Procedia Computer Science*, vol. 215, pp. 742–751, 2022.

[4] Y. Guo, T. Ji, Q. Wang, L. Yu, G. Min, and P. Li, “Unsupervised Anomaly Detection in IoT Systems for Smart Cities,” *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 4, pp. 2231–2242, Oct. 01, 2020.

[5] J.-H. Moon, J.-H. Yu, and K.-A. Sohn, “An ensemble approach to anomaly detection using high- and low-variance principal components,” *Computers and Electrical Engineering*, vol. 99, p. 107773, Apr. 2022.

[6] S. M. Nagarajan, G. G. Deverajan, A. K. Bashir, R. P. Mahapatra, and M. S. Al-Numay, “IADF-CPS: Intelligent Anomaly Detection Framework towards Cyber Physical Systems,” *Computer Communications*, vol. 188, pp. 81–89, Apr. 2022.

[7] J. Li, Z. Zhao, R. Li, and H. Zhang, “AI-Based Two-Stage Intrusion Detection for Software Defined IoT Networks,” *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2093–2102, Apr. 2019.

[8] C. Luo, Z. Tan, G. Min, J. Gan, W. Shi, and Z. Tian, “A Novel Web Attack Detection System for Internet of Things via Ensemble Classification,” *IEEE Transactions on Industrial Informatics*, vol. 17, no. 8, pp. 5810–5818, Aug. 2021.

[9] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A.-R. Sadeghi, “DÏoT: A Federated Self-learning Anomaly Detection System for IoT,” *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, Jul. 2019.

[10] Y. Chen, J. Zhang, and C. K. Yeo, “Network Anomaly Detection Using Federated Deep Autoencoding Gaussian Mixture Model,” *Machine Learning for Networking. Springer International Publishing*, pp. 1–14, 2020.

[11] B. Li, Y. Wu, J. Song, R. Lu, T. Li, and L. Zhao, “DeepFed: Federated Deep Learning for Intrusion Detection in Industrial Cyber–Physical Systems,” *IEEE Transactions on Industrial Informatics*, vol. 17, no. 8, pp. 5615–5624, Aug. 2021.

[12] T. Huong, T. Bac, D. Long, T. Luong, N. Dan, L. Quang, L. Cong, B. Thang, and K. Tran, “detecting cyberattacks using anomaly detection in industrial control systems: A federated learning approach,” *Comput. Ind. 132*, 2021.

[13] L. Zhao, J. Li, Q. Li, and F. Li, “A Federated Learning Framework for Detecting False Data Injection Attacks in Solar Farms,” *IEEE Transactions on Power Electronics*, vol. 37, no. 3, pp. 2496–2501, Mar. 2022.

[14] V. Rey, P. M. Sánchez Sánchez, A. Huertas Celdrán, and G. Bovet, “Federated learning for malware detection in IoT devices,” *Computer Networks*, vol. 204, p. 108693, Feb. 2022.

[15] L. Cui et al., “Security and Privacy-Enhanced Federated Learning for Anomaly Detection in IoT Infrastructures,” *IEEE Transactions on Industrial Informatics*, vol. 18, no. 5, pp. 3492–3500, May 2022.

[16] Z. A. E. Houda, B. Brik, A. Ksentini, L. Khoukhi, and M. Guizani, “When Federated Learning Meets Game Theory: A Cooperative Framework to Secure IIoT Applications on Edge Computing,” *IEEE Transactions on Industrial Informatics*, vol. 18, no. 11, pp. 7988–7997, Nov. 2022.

[17] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, “Federated Learning: Challenges, Methods, and Future Directions,” *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, May 2020.

[18] O. Suciu, R. Marginean, Y. Kaya, and H. Daume, “When does machine learning fail? Generalized transferability for evasion and poisoning attacks,” in *Proc. USENIX Conference on Security Symposium (USENIX Security’18)*, pp. 1299–1316, 2018.

[19] N. Baracaldo, B. Chen, H. Ludwig, and J. A. Safavi, “Mitigating Poisoning Attacks on Machine Learning Models,” *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security. ACM*, Nov. 03, 2017.

[20] Y. Zhao, J. Chen, J. Zhang, D. Wu, M. Blumenstein, and S. Yu, “Detecting and mitigating poisoning attacks in federated learning using generative adversarial networks,” *Concurrency and Computation: Practice and Experience*, vol. 34, no. 7, Jun. 29, 2020.

[21] D. Yin, Y. Chen, K. Ramchandran, and P. Bartlett, “Byzantine-Robust Distributed Learning: Towards Optimal Statistical Rates.” *arXiv*, 2018.

[22] J. Zhang, C. Ge, F. Hu, and B. Chen, “RobustFL: Robust Federated Learning Against Poisoning Attacks in Industrial IoT Systems,” *IEEE Transactions on Industrial Informatics*, vol. 18, no. 9, pp. 6388–6397, Sep. 2022.

[23] M. Du, H. Zheng, X. Feng, Y. Chen, and T. Zhao, “Decentralized Federated Learning with Markov Chain Based Consensus for Industrial IoT Networks,” *IEEE Transactions on Industrial Informatics*, vol. 19, no. 4, pp. 6006–6015, Apr. 2023.

[24] C. Hu, J. Jiang, and Z. Wang, “Decentralized Federated Learning: A Segmented Gossip Approach”, *1st International Workshop on Federated Machine Learning for User Privacy and Data Confidentiality*, 2019.

[25] M. Blot, D. Picard, M. Cord, and N. Thome, “Gossip training for deep learning.” *arXiv*, 2016.

[26] J. Daily, A. Vishnu, C. Siegel, T. Warfel, and V.

Amatya, "GossipGraD: Scalable Deep Learning using Gossip Communication based Asynchronous Gradient Descent." *arXiv*, 2018.

[27] A. G. Roy, S. Siddiqui, S. Pölsterl, N. Navab, and C. Wachinger, "BrainTorrent: A Peer-to-Peer Environment for Decentralized Federated Learning." *arXiv*, 2019.

[28] A. Lalitha, S. Shekhar, T. Javidi, and F. Koushanfar, "Fully decentralized federated learning", *3rd Workshop on Bayesian Deep, NIPS Workshop*, 2018.

[29] A. Lalitha, O. C. Kilinc, T. Javidi, and F. Koushanfar, "Peer-to-peer Federated Learning on Graphs." *arXiv*, 2019.

[30] L. Barbieri, S. Savazzi, M. Brambilla, and M. Nicoli, "Decentralized federated learning for extended sensing in 6G connected vehicles," *Vehicular Communications*, vol. 33, p. 100396, Jan. 2022.

[31] S. Savazzi, M. Nicoli, and V. Rampa, "Federated Learning with Cooperating Devices: A Consensus Approach for Massive IoT Networks," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4641–4654, May 2020.

[32] H. Jin, X. Dai, J. Xiao, B. Li, H. Li, and Y. Zhang, "Cross-Cluster Federated Learning and Blockchain for Internet of Medical Things," *IEEE Internet of Things Journal*, vol. 8, no. 21, pp. 15776–15784, Nov. 01, 2021.

[33] Y. Qu et al., "Decentralized Privacy Using Blockchain-Enabled Federated Learning in Fog Computing," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 5171–5183, Jun. 2020.

[34] S. Andreina, G. A. Marson, H. Mollering, and G. Karame, "BaFFLe: Backdoor Detection via Feedback-based Federated Learning," *2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS)*, Jul. 2021.

[35] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How To Backdoor Federated Learning." *arXiv*, 2018.

[36] V. Tolpegin, S. Truex, M. E. Gursoy, and L. Liu, "Data Poisoning Attacks Against Federated Learning Systems." *arXiv*, 2020.

[37] N. Moustafa and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," *2015 Military Communications and Information Systems Conference (MilCIS)*, Nov. 2015.

[38] N. Moustafa, "the future of IoT mini guide: The burgeoning IoT market continues," Available: www.unsw.adfa.edu.au/australian-centre-for-cyber-security/cybersecurity/ADFA-NB15-Datasets, Dec. 2021.

[39] M. Malik, M. Dutta, and J. Granjal, "A Survey of Key Bootstrapping Protocols Based on Public Key Cryptography in the Internet of Things," *IEEE Access*, vol. 7, pp. 27443–27464, 2019.

[40] A. Arya and S. Jha, "An Analytical Review on Data Privacy and Anonymity in 'Internet of Things (IoT) Enabled Services,"' *2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N)*, Dec. 17, 2021.

[41] N. Moustafa, G. Misra, and J. Slay, "Generalized Outlier Gaussian Mixture Technique Based on Automated Association Features for Simulating and Detecting Web Application Attacks," *IEEE Transactions on Sustainable Computing*, vol. 6, no. 2, pp. 245–256, Apr. 01, 2021.

[42] https://www.kaggle.com/code/carlkirstein/unsw-nb15-modelling-97-7.

[43] S. M. Kasongo and Y. Sun, "Performance Analysis of Intrusion Detection Systems Using a Feature Selection Method on the UNSW-NB15 Dataset," *Journal of Big Data*, vol. 7, no. 1. Springer Science and Business Media LLC, Nov. 25, 2020.

[44] A. Halbouni, T. S. Gunawan, M. H. Habaebi, M. Halbouni, M. Kartiwi, and R. Ahmad, "CNN-LSTM: Hybrid Deep Neural Network for Network Intrusion Detection System," *IEEE Access*, vol. 10, pp. 99837–99849, 2022.

[45] H. C. Altunay and Z. Albayrak, "A hybrid CNN+LSTM-based intrusion detection system for industrial IoT networks," *Engineering Science and Technology, an International Journal*, vol. 38, p. 101322, Feb. 2023.

[46] S. Kim, L. Chen, and J. Kim, "Intrusion Prediction using LSTM and GRU with UNSW-NB15," *2021 Computing, Communications and IoT Applications (ComComAp)*, Nov. 26, 2021.

[47] P. TS and P. Shrinivasacharya, "Evaluating neural networks using Bi-Directional LSTM for network IDS (intrusion detection systems) in cyber security," *Global Transitions Proceedings*, vol. 2, no. 2, pp. 448–454, Nov. 2021.

[48] L. Ashiku and C. Dagli, "Network Intrusion Detection System using Deep Learning," *Procedia Computer Science*, vol. 185, pp. 239–247, 2021.

Zahra Eskandari received the B.S degree in Computer Engineering from Kharazmi University, Tehran, Iran, in 2006. She received her M.S. and Ph.D. degrees in Computer Engineering from Ferdowsi University of Mashhad, Iran, in 2008 and 2020, respectively. She was with the cybersecurity section at DTU compute, Denmark as a visiting researcher from July 2016 to March 2017. She is a full-time Assistant-Professor in the Department of Computer Engineering, Faculty of Electrical and Computer Engineering, Quchan University of Technology, Quchan, Iran. Her research interests include security in IoT, AIoT and Edge computing.