MITIGATING POISONING ATTACKS TO FEDERATED LEARNING IN IoTs ANOMALY DETECTION WITH ATTENTION AGGREGATION

Ly Vu[1,*], *Tuan Phong Tran*[1], *Van Cuong Nguyen*[1], *Quang Uy Nguyen*[1]

Abstract

Federated Learning (FL) is a privacy-preserving approach to train deep neural networks across decentralized devices without sharing raw data. Thus, FL has been popularly applied in domains like anomaly detection in Internet of Things (IoTs). However, IoT networks or devices have limited protection capabilities, resulting in the vulnerability of FL to data poisoning attacks. In order to address this challenge, we propose a new robust FL system designed to counter data poisoning attacks. Our approach, named as Federated Learning with Attention Aggregation (FedAA), leverages AutoEncoder (AE) models for local anomaly detection in IoT networks. In FedAA, the global model is aggregated from local models by using a novel aggregation method, named as Attention Aggregation (AA). This method is specifically designed to mitigate the impact of data poisoning attacks, which often lead to high values of the loss functions in the local models. More precisely, the local models with high loss values are assigned lower attention weights when contributing to the global model aggregation, and vice versa. As a result, the proposed AA method enhances the robustness of FedAA against data poisoning attacks. The extensive experiments are conducted on three datasets including N-BaIoT, NSL-KDD, and UNSW, of IoT anomaly detection. The results show that FedAA is more robust than other FL systems in mitigating data poisoning attacks.

Index terms

Anomaly detection; IoT; Attention Aggregation; Federated Learning.

1. Introduction

The IoTs are transforming all aspects of life with its applications spanning across diverse domains, ranging from industry automation, healthcare, to smart home/city. The number of Internet of Thing (IoT) devices is anticipated to increase to approximately $11 billion by 2026 [1]. However, the popularity of IoT devices and their important role in critical applications also expose the network to greater risks of security attacks. In the first half of 2021, there were nearly 1.5 million attacks against IoT devices [2].

[1]Institute of Information and Communication Technology, Le Quy Don Technical University
[*]Corresponding author, email: vu.ly@lqdtu.edu.vn DOI: 10.56651/lqdtu.jst.v13.n02.925.ict

Therefore, it is crucial to identify and prevent attacks to IoT networks in order to advance the development of IoT applications.

In anomaly detection for IoT networks, AE is considered to be one of the most effective approaches [3]–[8]. The AE architecture enables to transform the raw IoT data into a more refined data representation that effectively highlights the inherent characteristics of the input data. Thus, it facilitates the downstream classifiers to detect attack samples.

However, collecting and centering data from IoT networks to train an effective AE is a challenging task [3]–[5], [7]. This is due to user data privacy concerns [9]. Users of IoT devices often refrain from sharing data that might possibly violate their privacy or expose sensitive information about their systems. Subsequently, FL is considered as the most effective approach to develop anomaly detection for IoT networks. FL does not require IoT devices nor networks to share their local data with the centralized server. Instead, distributed nodes (also referred to as clients) only need to share their local gradients with the centralized server, which in turn aggregates into the global gradient and shares it with the local nodes [10]–[16].

In order to train a robust FL scheme, one needs to assume that the training processes of all clients are honestly conducted. However, such an assumption is not always realistic owing to both inadvertent and deliberate causes [17]–[19]. For example, the attackers may take control of the number of IoT devices in the FL scheme, i.e., genuine clients, and then inject the poisoning data during the local training phase [10], [18]. This is a type of poisoning attacks. The poisoning attacks can be divided into two groups, i.e., data poisoning attacks and model poisoning attacks. In data poisoning attacks, attackers can take control of at least one client in an FL scheme to manipulate its training data. Besides, in model poisoning attacks, they also take the control of clients to modify their local models, e.g., deep neural network models [10]. Data poisoning attacks are generally easier to execute than model poisoning attacks because attackers can manipulate accessible local training data with less technical expertise and lower detection risk, exploiting a wider attack surface across diverse client datasets [17]. Therefore, developing a robust FL scheme for IoT anomaly detection that mitigates data poisoning attacks is highly desirable.

In an FL scheme, multiple local models residing on different devices or clients, collaborate to build a global model. Aggregating these local models into a coherent global model is a critical step in the training process. Averaging-based method [11], [12] is one of the most widely used aggregation methods that averages the model parameters across all participating clients. This aggregation method is simple, efficient, and exhibits good convergence properties [12]. However, it assumes that all clients have the same contribution to the global model. In reality, this assumption may not be also hold.

In this paper, a novel FL scheme for IoT anomaly detection called FedAA is proposed that is robust against the data poisoning attacks. FedAA uses an AutoEncoder (AE) as the local model to learn useful characteristics from only normal data. To mitigate the impacts of the data poisoning attack, FedAA utilizes a new aggregation method, namely

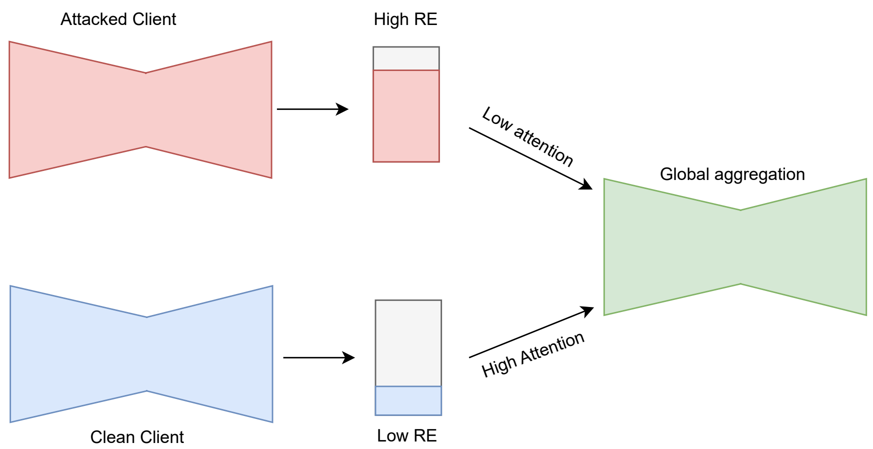

Fig. 1. Motivation of the Loss Attention Aggregation method.

AA, which aggregates the local models with different attention weights corresponding to the loss values of the local models. The motivation for FedAA can be shown in Fig. 1. When a client is attacked by the data poisoning attack, the training loss value of the local model tends to be significantly amplified. To mitigate the impact of attacked clients or poisoned clients to the global model, FedAA assigns a lower weight for the higher loss value and vice versa. This method helps to improve the robustness of the FedAA scheme against data poisoning attacks. The main contributions of this paper are as follows:

- A new aggregation method, i.e., AA, is proposed to aggregate local models at the server based on their training loss values. This helps to alleviate the influence of malicious clients on the global model.
- A new FL scheme which trains semi-supervised local models, i.e., the AE models, and aggregates the local models using the AA method is introduced.
- To assess the effectiveness of the proposed system (FedAA) for IoT anomaly detection, the comprehensive experiments are conducted on three datasets, i.e., N-BaIoT, NSL-KDD, and UNSW.

The rest of the paper is organized as follows. In Section 2, the essential foundation of our proposed FL scheme is presented. Section 3 reviews the most recent methods of machine learning and FL on network anomaly detection. Following that, a comprehensive description of the proposed scheme is provided in Section 4. In Section 5, the datasets and experimental settings are described in details. The experimental results are presented in Section 6. The conclusion is discussed in Section 7.

2. Background

This section presents the fundamental background of the AE architecture and the FL scheme. They are core components in our FL system for IoT anomaly detection.

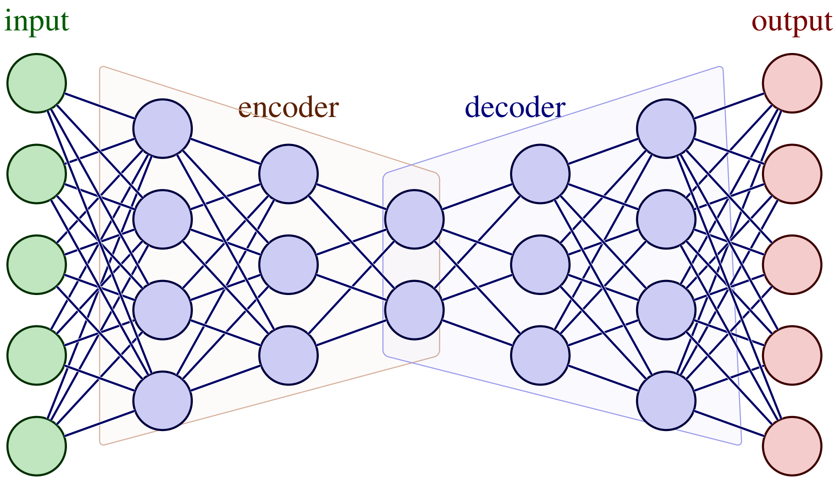

Fig. 2. Architecture of an AE.

2.1. AutoEncoder

An AE as illustrated in Fig. 2 is a neural network model that is commonly utilized for unsupervised or semi-supervised anomaly detection. The AE architecture consists of two components, i.e., an encoder and a decoder. The encoder attempts to map the input data onto a latent representation space that usually has a lower dimension than the input data. The decoder reconstructs the input at the output. The training process of AE attempts to minimize the difference between the input and the output. This difference is called the reconstruction error (RE) [20].

We denote $\omega = (\mathbf{W_e}, \mathbf{b_e})$ to be the weight matrix and the bias vector of the encoder, respectively, and denote $\mathbf{z^i}$ to be the latent representation of the input sample $\mathbf{x^i}$, the latent representation $\mathbf{z^i}$ is calculated as follows:

$$\mathbf{z}^i = q_\omega(\mathbf{x}^i) = \sigma_e(\mathbf{W_e}\mathbf{x}^i + \mathbf{b_e}), \tag{1}$$

where q_ω and σ_e represent the encoder and its activation function, respectively.

Similarly, $\theta = (\mathbf{W_d}, \mathbf{b_d})$ denotes the parameter vector of the decoder p_θ and σ_d is the activation function of the decoder, the reconstructed output $\hat{\mathbf{x}}$ is calculated as follows:

$$\hat{\mathbf{x}}^i = p_\theta(\mathbf{z}^i) = \sigma_d(\mathbf{W_d}\mathbf{z}^i + \mathbf{b_d}). \tag{2}$$

The training objective of AE is to minimize the loss function of the AE, which is often estimated as the mean squared error computed in Eq. (3).

$$\mathcal{L}_{AE}(\mathbf{x}, \phi, \theta) = \sum_{i=0}^{n} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2, \tag{3}$$

where n is the dataset size, and $\mathbf{x}_i$ and $\hat{\mathbf{x}}_i$ are the input and output of the AE, respectively. The weights of AE ϕ including the weights of the encoder and decoder, i.e., $\phi = (\omega, \theta)$.

For the anomaly detection problem, AE is commonly trained on benign or normal data samples. After training, the RE, which is $\mathcal{L}_{\mathrm{AE}}$ in Eq. (3), is used to calculate the anomaly score for the anomaly detection problem. Specifically, if an input sample results in RE that is higher than a predefined threshold, this sample will be considered as an anomaly [3], [11]; otherwise, it is considered normal.

2.2. Federated learning

Federated Learning is a decentralized machine learning (ML)/deep learning (DL) system wherein numerous clients collaborate to train a ML/DL model with the supervision of a central server [9], [10], [21]. The clients can be mobile phones, tablets, speakers, terminal IoT devices, and large institutions. The FL scheme aims to collaboratively train a global model using training data located on multiple clients. The role of the central server is to aggregate the models trained from the multiple clients. The aggregation based on averaging local models [12] is commonly used in FL frameworks.

Aggregating the global model in FL is a vital step that impacts the performance and convergence of the global model. In this paper, we propose a new aggregation method based on the loss of the client's model to mitigate the impact of the suspecting clients. The clients with a higher loss are assigned to a lower weight since these clients are more likely to be malicious clients that have been targeted by data poisoning attacks.

3. Related work

Applying ML/DL to anomaly detection in IoT networks has achieved many promising results [22]. There are many ML/DL models used for anomaly detection, such as Decision Tree [23], Support Vector Machines [24], [25], AE [3], [4]. Among them, the AE-based models [3], [4], [6], [26], [27] have been widely used thanks to their ability to automatically extract features from large amounts of network traffic data. This section briefly discusses the recent research on the application of AE-based models to anomaly detection.

Cao *et al*. [3] proposed an AE model for network anomaly detection named Shirk AutoEncoder (SAE). The SAE model is a semi-supervised learning model that aims to force the training benign data closer to the origin, hence better separating anomalies from the benign data. The results showed that the SAE model is effective unsupervised learning models for anomaly detection. Hwang *et al*. [27] proposed a model called D-PACK for detecting abnormal traffic patterns that combines convolutional neural network (CNN) and AE. This technique examines only few packets and bytes in a network connection. Thus, the D-PACK technique can dramatically minimize the amount of traffic that needs to be processed. Vu *et al*. [4] utilized the label information to propose two AE-based models for anomaly detection named Multiple AutoEncoder (MAE) and Multiple Variational AutoEncoder (MVAE). The MAE and MVAE models aim to project the benign and abnormal data into two separated regions on the latent representation space. Thus, distinguishing the abnormal data from the benign data is more effective.

Recently, Yin *et al*. proposed a model for time series anomaly detection based on the AE architecture that combines CNN and long short-term memory (LSTM) [26]. They used a sliding window technique to extract low-level temporal features. After that, they applied CNN and LSTM based on the AE architecture to enhance the accuracy of anomaly detection. Salahuddin *et al*. [6] proposed the Chronos technique, a novel time-based anomaly detection system that leverages an AE to detect anomalous Distributed Denial of Service (DDoS) traffic. Thus, the AE-based models are widely used to represent network traffic data for anomaly detection problems.

The above ML/DL models are trained on a central server, thus they exhibit the drawback of data privacy [19]. The FL technique has gained considerable attention as a promising approach to address data privacy concerns [11]–[16], [28]. DeepFed [14] was an FL scheme using a novel gated recurrent unit-convolution neural network (GRU-CNN) model to detect threats against industrial cyber-physical systems. Li *et al*. [16] introduced an FL framework based on Attention Mechanism-based Convolutional Neural Network Long Short Term Memory (AMCNN-LSTM) model to detect anomalies in time series data. The works in [11], [28] utilized the FL schemes to train the AE models for anomaly detection. Tuo *et al*. [11] demonstrated that the FL scheme based on the AE model, named as FedDetect, is effective in detecting anomalies in the IoT environment.

In an FL framework, aggregating local models to create a global model is an important task to collaborate multiple clients. Averaging based method is the most well-known aggregation method for FL frameworks [11], [12]. This method aggregates the weights of local models received from participating clients by averaging. However, this method is unable to recognize the signatures of suspecting clients, which can be poisoned by attackers, to mitigate or exclude them from the aggregation task. Thus, this technique is not robust against the poisoning attacks.

There are several methods for preventing malicious clients or attacked from aggregating the global model. One approach is to use K-mean clustering to group the clients based on their model weights or the performance [29], [30]. Clients are clustered into high-performance and low-performance groups. The global model only aggregates the local models of the high performance groups. Additionally, the suspecting clients can also be identified by their poor performance metrics, which are below the threshold set for the Top-K selection [31]. These clients are excluded from the aggregation process to prevent their potentially malicious updates from affecting the global model. These aggregation methods help to reduce the influence of potentially malicious actors on the global model. However, these aggregation methods use a fixed threshold to eliminate the suspecting clients. Thus, they may incorrectly remove the benign clients in the aggregation. In this paper, we propose a new aggregation method named as AA method, which assigns attention weights to local models based on their training loss values. The AA method attempts to reduce the impact of suspecting clients, which have large training loss values, in the aggregation process.

4. Proposed method

In this section, the AA method is first introduced to reduce the influence of attacked clients during the training process of FedAA. Then, the description of the proposed FL system using the AA method, namely FedAA, is introduced including the architecture and the training procedure.

4.1. Attention aggregation

To reduce the impact of the data poisoning attack, a new aggregation method, named as AA, is introduced. This method is executed at the server to coordinate the local models received from the participating clients. The main idea of AA is reducing the contribution of the local models with the large training loss value in aggregating. Specifically, the aggregation task at server is conducted as following.

First, the contribution value of the local model of the client i to the global model, p_i^t, is calculated by Eq. 4. The purpose of this step is to penalize the clients with large loss values. The logarithm operator in Eq. 4 serves to stabilize the calculations by mitigating the influence of extreme values, both very small and very large. This approach prevents these outliers from disproportionately affecting the contribution value, resulting in more manageable and consistent outcomes. The detail of calculating contribution value is presented as follows:

$$p_i^t = \log \frac{\sum_{i=1}^{K} \mathcal{L}_i^t}{\mathcal{L}_i^t}. \tag{4}$$

Second, the weight of the client i, i.e., w_i^t, is calculated in Eq. 5 by normalizing the contribution value p_i^t to the range between 0 and 1:

$$w_i^t = \frac{p_i^t}{\sum_{i=1}^{K} p_i^t}, \tag{5}$$

where K is the number of clients. Normalizing weights to the range of 0 and 1 makes the magnitudes of the weights more interpretable. In other words, it helps to understand the relative importance or contribution of each local model. The weight value closed to 0 indicates the less importance, while the weight value closed to 1 represents very importance local model to the aggregation task.

Finally, the aggregation is implemented in Eq. 6 as follows:

$$\phi_g^t = w_i \times \sum_{i=1}^{K} \phi_i^t, \tag{6}$$

where ϕ_i^t is the model weight of the client i at the round t and ϕ_g^t is the global model aggregated at the round t. By assigning different weights to different local models, the AA technique is able to reduce the weight of the local models, which have large loss

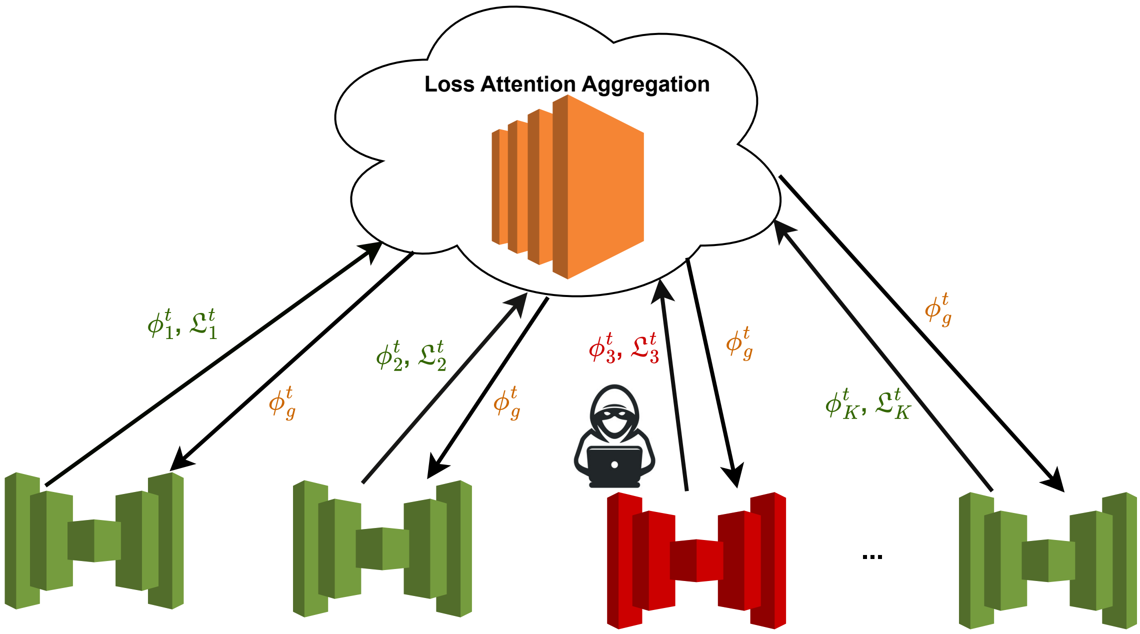

Fig. 3. Architecture of FedAA.

values. This is inspired from our observation that local models with the large loss values are often the victims of data poisoning attacked clients.

4.2. Architecture of FedAA

Fig. 3 illustrates the architecture of FedAA. FedAA consists of two main components, i.e., a server and multiple clients. In this figure, ϕ_i^t and ϕ_g^t represent the local model of the client i and the global model at the training round t, respectively. $\mathcal{L}_i^t$ is the loss value resulting from the training process of the local model ϕ_i^t at the round t. The local data can be poisoned by attackers, e.g., the third client in Fig. 3. The local models of attacked clients (also refer to as suspecting clients) have negative impacts on the performance of the FL scheme.

The training process of FedAA is described in Alg. 1. The inputs are the number of training round T and the training data x_{it} and the validation data x_{iv} for each participating client C_i. First, the server initializes the weights of the local model, i.e., the AE model, and sends them to all participating clients (as in the line 3 and 4). The stopping condition parameter $Stop_i$ is initialized as the False value for each client as in the line 5. Second, for each training round t, only clients, which do not meet the converge criteria, i.e., $Stop_i$ is False (as in the line 8), do the local training process. These clients initialize the local weight as ϕ_g^t as in the line 9 and do the local training described in Alg. 2. Third, after local training, the client sends the model's weights $\hat{\phi}_i^t$ and the loss value $\mathcal{L}_i^t$ to the server (as the line 10 and 11). Fourth, the sever aggregates the global weights ϕ_g^t using the AA algorithm and sends the global weights back to the participating clients (as the line 14 and 15). Fifth, the clients update their weights using the global weights and continue the training process using it's local data. This process is repeated for many training rounds until reaching the maximum number of training round T. The final local models are set as the global model (as the line 17).

Algorithm 1 . Federated training process of FedAA.

1: **Input**: Number of training round T, Client training and validation data x_{it}, x_{iv} for $i = 1, 2, \ldots K$, the batch size s, the learning rate lr, the optimization Op.
2: **Output**: Trained local model ϕ_i
3: Server initializes the global weights of AE ϕ_g^0,
4: Server sends the global weights to participating clients,
5: Initialize $Stop_i = False$ for $i = 1, 2, \ldots K$.
6: **for** $t \leftarrow 0$ to T **do**
7:     **for** $i \leftarrow 1$ to K **do**
8:         **if** $Stop_i$ = False **then**
9:             Initialize $\phi_i^t = \phi_g^t$
10:             $\hat{\phi}_i^t$, $\mathcal{L}_i^t$, $Stop_i$ = LOCAL-TRAINING (x_{it}, x_{iv}, ϕ_i^t, s, lr, Op),
11:             Client C_i sends its weights $\hat{\phi}_i^t$ and loss $\mathcal{L}_{iv}$ to the server,
12:         **end if**
13:     **end for**
14:     Server aggregates (ϕ_g^t by the AA algorithm described in Section 4-A,
15:     The server sends ϕ_g^t to participating clients.
16: **end for**
17: ϕ_i = ϕ_g^T
18: **Return**: ϕ_i

Specifically, the local training process for the local model ϕ_i^t at the client C_i in the round t is briefly presented. It follows the function $LOCAL - TRAINING$ outlined in Alg. 2. The inputs are the benign training data x_{it}, the benign validation data x_{iv}, the initialized local model ϕ_i^t, the batch size of local training s, the learning rate lr, the optimization algorithm Op. Moreover, the $Stop$ parameter is used to indicate when the client has met the convergence condition as in the line 3. The output of this algorithm is the updated weights of the local model, i.e., $\hat{\phi}_i^t$. The training process is executed by selecting a batch of training data x_{bs} as specified in the line 5 of the algorithm. The training data is then used to fit the current local model ϕ_i^t, allowing us to calculate the loss (as shown in the line 7) and update the parameters of $\hat{\phi}_i^t$ (in the line 8). At the end of the for loop, i.e., completing one training epoch, the validation loss value is calculated as in the line 10 and added to the list RE which monitors the validation loss value though training rounds. If the validation loss $\mathcal{L}_{iv}$ is larger than the mean of f previous values in the list RE, the current local model will meet the convergence requirement. Thus, the parameter $Stop$ is set as True value (as in the line 13).

To detect anomalies, the RE value resulted of the local training algorithm is leveraged as the Anomaly Score (AS). In the testing stage, if a data sample yields an RE value that exceeds the established AS, that data sample is classified as an anomaly. Conversely, if the RE value falls below AS, the data sample is deemed non-anomalous, i.e., a normal data sample.

Algorithm 2 . Local training in FedAA.

function LOCAL-TRAINING(x_{it}, x_{iv}, ϕ_i^t, RE, s, lr, Op)
  n is the number of data samples x_{it}
  Stop = False
  for $j = 0$ **to** n/s, **step** s **do**
    Take a batch of training data $x_{bs} \leftarrow x_{it}[j : j + s]$
    Fit x_{bs} to model ϕ_i^t
    Calculate the training loss $\mathcal{L}_{it}$ by Eq. 3 on x_{bs}.
    Obtain $\hat{\phi}_i^t$ by executing the Op algorithm with the loss $\mathcal{L}_{it}$ to update ϕ_i^t,
  end for
  Calculate the validation loss $\mathcal{L}_{iv}$ by Eq. 3 on x_{iv},
  Add $RE \leftarrow \mathcal{L}_{iv}$,
  if $\mathcal{L}_{iv}$ > mean (RE [-f:]) **then**
    Stop = True
  end if
  Return: $\hat{\phi}_i^t$, $\mathcal{L}_{iv}$, Stop
end function

5. Experimental settings

This section introduces the datasets used in the experiments, the performance metrics and the experimental scenarios.

5.1. Datasets

We evaluate the performance of FedAA using three well-known datasets, namely N-BaIoT [5], NSL-KDD [32], and UNSW [33] datasets. Each dataset is divided into three sets, i.e., the training set, the validation set, and the testing set, with a ratio of $7 : 1 : 2$, respectively. In the training and validating sets, we only use the benign (or normal) data samples. The testing sets include both normal and abnormal data samples. The number of data samples in each dataset is described in Table 1.

N-BaIoT dataset: The N-BaIoT dataset consists of actual traffic gathered from the 9 commercial IoT devices [5]. The dataset consists of 115 numeric characteristics representing network traffic data that was authentically infected by two common botnet attacks, i.e., Mirai and BASHLITE. In each IoT device, the dataset consists of either five or ten Distributed Denial of Service (DDoS) attacks. These attack types include scanning networks for vulnerable devices (Scan), sending spam data (Junk), UDP flooding (UDP), TCP flooding (TCP), and sending spam data while simultaneously opening a connection to a specified IP address and port (combo).

Table 1. The number of data samples in each dataset

Datasets	N-BaIoT	NSL-KDD	UNSW
Training set	19408	58925	65190
Validating set	19407	8418	25768
Testing set	38093	22544	51535

NSL-KDD dataset: The NSL-KDD dataset [32] is an intrusion detection dataset extensively used to evaluate various anomaly detection problems. Each data sample consists of 41 characteristics and is labeled as either an attack (or anomaly) or a normal packet. Three categorical features including *protocol* type, *service*, and *flag*, are preprocessed using one-hot-encoding. Thus, the total number of features increases to 122. The training set consists of 24 distinct attack types, while the testing set comprises 14 new attack types that are not included in the training set. All attacks fall into one of the following four categories, i.e., Denny of Service (DoS), Remote-to-Local (R2L), User-to-Root (U2R), and Probing.

UNSW dataset: The UNSW dataset [33] is widely utilized for abnormal detection. Each data sample in this dataset contains a total of 49 features. These features are designed to capture various aspects of network traffic and system behavior, providing a comprehensive representation of the data for intrusion detection purposes. The features allow for a detailed analysis of network activities and aid in the development and evaluation of anomaly detection algorithms. The attack types are DoS Attacks, Distributed DoS (DDoS) Attacks, Probe, R2L, U2R, Web Application Attacks, Botnet Attacks.

5.2. Evaluation metrics

We utilize the F1 score to evaluate comprehensively the accuracy of the proposed system for anomaly detection. Before defining the F1 score, the Precision and Recall metrics are introduced for the anomaly detection problem. Precision measures the ratio of the number of correctly detected anomalies, i.e., True Positives (TP), to the total number of detected anomalies. In the context of anomaly detection, Precision measures the ability of an anomaly detection system to avoid predicting normal samples as anomalies. A high Precision value indicates that the system is highly accurate to classify abnormal cases. The formula to calculate Precision is presented in Eq. 7 as follows:

$$Precision = \frac{TP}{TP + FP}, \tag{7}$$

where FP the number of incorrectly detected anomalies or False Positive.

Recall measures the ratio of the number of correctly detected anomalies, i.e., TP, to the total number of true anomalies. The number true anomalies includes the number of correctly detection of anomalies and the number of incorrectly detection of normal cases, i.e., False Negative (FN). Recall in the anomaly detection problem measures the

Table 2. Hyper-parameter settings for the experimental FL schemes

Hyper-parameter	Description	Value
s	Batch size in local training	32
lr	Learning rate	0.001
Op	Optimization algorithm in local training	Adam
K	Number of training clients	10
T	Maximum number of training rounds	10000
f	Number of previous rounds observed for early stopping	300

system's ability to find all real anomalies. A high Recall value indicates that the system has good coverage of anomalous samples. The formula to calculate Recall is presented in Eq. 8 as follows:

$$Recall = \frac{TP}{TP + FN}. \tag{8}$$

Precision and Recall often have an inverse relationship. When improving Precision, there can be a decrease in Recall and vice versa. Therefore, we use the F1 score evaluating the performance of an anomaly detection system needs to consider both of these metrics. It provides a comprehensive view of the ability to detect abnormal data samples. The F1 score is defined based on the Precision in Eq. 7 and Recall in Eq. 8 as follows:

$$F1 = \frac{(2 \times Precision \times Recall)}{(Precision + Recall)}. \tag{9}$$

5.3. Experimental setup

We implement the FL systems that simulate a distributed environment with multiple clients. The number of clients is set as 20. The server aggregates the local models of participating clients using one of the four aggregation methods, i.e., Average [12], K-mean [30], TopK [31], and AA corresponding to four FL systems including FedDetect [11], FedKmean, FedTopK, and FedAA.

In data poisoning attack scenarios, we inject noise to local data of $r\%$ clients where r is set as 25 and 50 in our experiments. To generate poisoning attacks on a single client, we add a randomly generated Gaussian noise vector with a mean 0 and a standard deviation 0.1 to each training data sample. Moreover, the main hyper-parameters presented in Table 2 using for all experimental FL systems.

Two following experimental scenarios are conducted:

- *Without Poisoning Attack:* There are no data poisoning attacks during the training process of FL systems.
- *With Poisoning Attack:* We evaluate the performance of both benign and attacked clients when the FL schemes are under attacks.

- *Loss Visualization:* We visualize the training losses of benign and attacked clients to understand the impact of each type of clients to the global model.

6. Results and discussion

This section evaluates the performance of FedAA in the absence and presence of poisoning attacks and compare this scheme to the other FL schemes.

6.1. No poisoning attack

Table 3. F1 Scores for the FL systems with no attacking

Metric	FedDetect	FedKmean	FedTopK	FedAA
N-BaIoT	**0.682**	0.658	0.666	**0.682**
NSL-KDD	0.921	0.917	0.920	**0.922**
UNSW	0.648	**0.666**	0.558	0.654

Table 3 presents the F1 scores of the four FL systems, i.e., FedDetect, FedKmean, FedTopK, and FedAA in the absence data poisoning attacks. The RE value is used to determine whether a testing data sample is a normal or abnormal data sample for each FL framework. We can observe that the F1 scores of FedAA are the highest scores in the N-BaIoT and NSL-KDD datasets while FedDetect and FedKmean enhance the best F1 scores on the N-BaIoT and the UNSW datasets, respectively. These results prove that when poisoning attack is absence, the proposed system, i.e., FedAA, is slightly better than the other FL systems on the experimental datasets.

6.2. With poisoning attack

6.2.1. Average performance

Table 4. F1 Scores for Benign and Attacked Clients on Three Datasets with 25% Data Poisoning Attack

Algorithm	Benign Clients				Attacked Clients			
	FedDetect	FedKmean	FedTopK	FedAA	FedDetect	FedKmean	FedTopK	FedAA
N-BaIoT	0.645	0.650	0.668	**0.738**	0.514	0.000	**0.573**	0.548
NSL-KDD	0.922	0.925	0.922	**0.935**	0.739	0.752	0.755	**0.761**
UNSW	0.639	0.542	0.519	**0.657**	0.012	0.000	0.000	**0.022**

Table 4 presents the average of F1 scores of local models trained by the four FL systems, i.e., FedDetect, FedKmean, FedTopK, and FedAA in the present of poisoning attack scenarios. We average these scores of benign clients and attacked clients separately to easy comparison. In this context, 25% of the clients are subjected to data poisoning attacks.

It can be seen from this table that FedAA consistently achieves the highest F1 scores on three datasets. For example, on the N-BaIoT dataset, FedAA achieves the F1 score of 0.738, surpassing the FedDetect, FedKmean, and FedTopK frameworks with the F1 scores as 0.645, 0.650, and 0.668, respectively. The similar observations can be seen in the NSL-KDD and UNSW datasets. These results indicate that the proposed FL technique, i.e., FedAA, is effective in mitigating the influence of the data poisoning attack on the FL system for the benign clients.

Furthermore, Table 4 also presents the average F1 scores for the local models of the attacked clients, which were trained using poisoned data. We observe that the F1 scores for the FL systems are significantly lower than those of the benign clients. This highlights the substantial impact of data poisoning attacks on the FL systems, resulting in reduced accuracy for the affected clients. For instance, the F1 scores for the attacked clients training with the UNSW dataset are alarmingly low, with many values approaching zero across all experimental FL systems. Among these, FedAA demonstrates slightly higher F1 scores compared to the other FL systems in detecting anomalies.

Additionally, we increase the proportion of attacked clients to 50% of the total client base. The averages of F1 scores for benign and attacked clients are presented in Table 5. We observe a decline in the average accuracy for both groups compared to the results in Table 4, where only 25% of clients were attacked. Among the four FL systems assessed, our proposed system, i.e., FedAA, consistently achieves the highest F1 scores across all three datasets. This suggests that the FedAA system effectively enhances the accuracy of anomaly detection for benign clients, even in the presence of data poisoning attacks. However, the F1 scores for the attacked clients remain very low, indicating significant room for improvement in the FL systems for these clients in future work.

Table 5. F1 Scores for Benign and Attacked Clients on Three Datasets with 50% Data Poisoning Attack

Algorithm	**Benign Clients**				**Attacked Clients**			
	FedDetect	**FedKmean**	**FedTop K**	**FedAA**	**FedDetect**	**FedKmean**	**FedTop K**	**FedAA**
N-BaIoT	0.719	0.686	0.714	**0.738**	**0.217**	0.000	0.000	0.182
NSL-KDD	0.934	0.905	0.902	**0.947**	0.727	0.741	0.701	**0.750**
UNSW	0.677	0.667	0.653	**0.687**	0.000	0.000	0.000	0.000

Overall, when evaluating the performance across all client types, including both benign and attacked clients, our proposed FL scheme, i.e., FedAA, demonstrates superior results compared to other FL schemes for the anomaly detection problems, particularly for benign clients. The new AA method helps FedAA to effectively mitigate the influence of attacked clients by diminishing their contributions during the aggregation process. As a result, benign clients are less affected by the negative impact of poisoning attacks. This evidence supports the effectiveness of our approach in safeguarding the integrity of IoT anomaly detection against data poisoning attacks. However, the attacked clients also diminish their learning capacity in the FL systems, resulting in very low accuracy in detecting anomalies.

6.2.2. Influence of data poisoning attacks

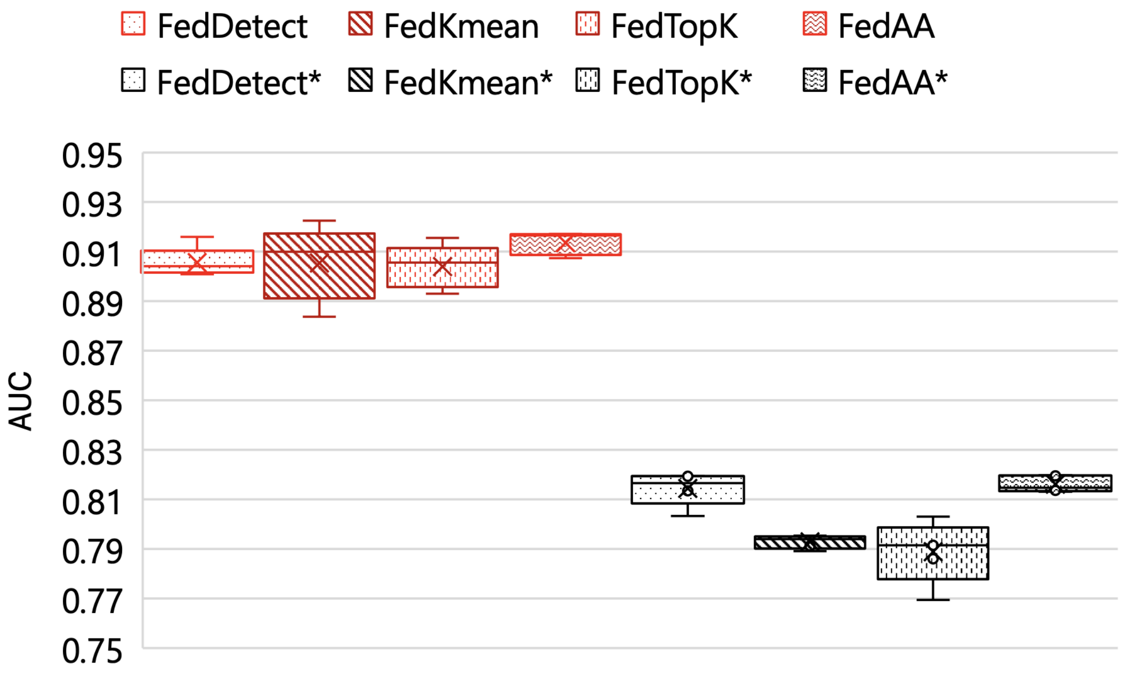

Fig. 4. The F1 scores of attacked clients on the N-BaIoT dataset. Here, the FL scheme M and M indicate the results of M on the benign clients and the corresponding clients when they are attacked.*

To assess the impact of data poisoning attacks on clients, we analyze how their accuracy changes during these attacks. Fig. 4 shows the F1 scores for various FL methods when 25% of clients are affected, using the N-BaIoT dataset. On the left side of the figure, the four boxes represent the F1 scores for benign clients, while the right side displays the scores for the same clients under attack. Each box plot provides a clear visual representation of accuracy across multiple clients, highlighting important statistics like the mean and standard deviation. The central line within each box indicates the average accuracy, serving as a benchmark for overall performance. The box itself contains the interquartile range, which represents the middle 50% of the data, while the whiskers show the full range of scores. Together, these features allow us to quickly understand both the average performance and variability, revealing how consistently the models perform relative to each other.

We can observe that when a client is subjected to a data poisoning attack, there is a significant reduction in that client's accuracy. This decline occurs because the poisoned data influences the learning process, leading to incorrect model predictions and a lower ability to detect anomalies effectively. Beside, the F1 scores of clients, in both scenarios when under attack and when not with the FedAA method consistently outperform those of other FL approaches, such as FedDetect, FedKmean, and FedTopK. It shows that FedAA exhibits greater robustness against data poisoning attacks. Moreover, the reduced variability in the F1 scores for FedAA indicates more stability among the attacked

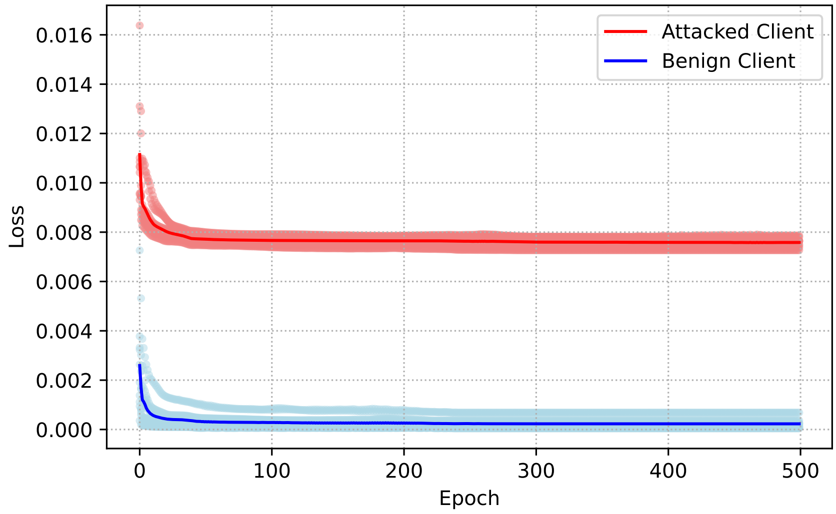

Fig. 5. Training loss visualization of the proposed solution on the N-BaIoT dataset.

clients, highlighting its superior resilience compared to the other FL systems for IoT anomaly detection.

6.3. Loss visualization

Fig. 5 illustrates the training loss of both benign and attacked clients on the N-BaIoT dataset. This figure clearly demonstrates a noticeable disparity in the training loss between the two groups of clients. The two curves of the losses gradually converge, suggesting that FedAA effectively reduces the training loss in both types of clients. Nonetheless, it is worth noting that the training loss for the benign clients consistently remains lower than that of the attacked clients, indicating negative impacts of the attacked clients during the training process. Thus, the aggregation process based on the AA algorithm, which reduces the weights of the attacked clients in aggregating the global model, is reasonable to mitigate the impact of poisoning attacks and enhance the accuracy of FL systems for IoT anomaly detection.

7. Conclusion

This paper introduced a novel FL system, called as FedAA. FedAA is designed to address the IoT anomaly detection problem while providing robustness against data poisoning attacks. The FedAA system enables collaborative training of local models across multiple clients. It ensures data privacy within the IoT network by eliminating the need for data transmission between IoT devices. Within the FedAA system, the AA method calculates the global model by considering the attention weights of each client's local model. This approach effectively mitigates the impact of local models of suspecting clients to the global model, thereby reducing the influence of data poisoning attacks. The effectiveness of the FedAA framework has been demonstrated through extensive experiments conducted on three anomaly detection datasets.

The FedAA framework is designed to mitigate the impact of poisoning attacks on benign clients. However, attacked clients struggle to achieve satisfactory accuracy when training FL systems under attack. Therefore, our future work will expand the FedAA framework to improve accuracy for both benign and attacked clients. Additionally, we will analyze adaptive attacks where an attacker first injects a small amount of poisoned data to familiarize the local model with it. This strategy allows the attacker to gradually increase the volume of poisoned data, ultimately preventing the server from recognizing the compromised clients.

Acknowledgment

The research is funded by Vingroup Innovation Foundation (VinIF), under grant number VINIF.2023.DA059.

References

[1] J. Wang, M. K. Lim, C. Wang, and M.-L. Tseng, "The evolution of the Internet of Things (IoT) over the past 20 years," *Computers & Industrial Engineering*, vol. 155, 2021. DOI: 10.1016/j.cie.2021.107174

[2] R. Daws, "Kaspersky: Attacks on IoT devices double in a year," *Kaspersky*, 07 Sept, 2021. [Online]. Available: https://iottechnews.com/news/kaspersky-attacks-on-iot-devices-double-in-a-year/ [Accessed 22-Dec-2022].

[3] V. L. Cao, M. Nicolau, and J. McDermott, "Learning neural representations for network anomaly detection," *IEEE Transactions on Cybernetics*, vol. 49, no. 8, pp. 3074–3087, 2019. DOI: 10.1109/TCYB.2018.2838668

[4] L. Vu, V. L. Cao, Q. U. Nguyen, D. N. Nguyen, D. T. Hoang, and E. Dutkiewicz, "Learning latent representation for IoT anomaly detection," *IEEE Transactions on Cybernetics*, vol. 52, pp. 3769–3782, 2020. DOI: 10.1109/TCYB.2020.3013416

[5] Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, A. Shabtai, D. Breitenbacher, and Y. Elovici, "N-BaIoT-network-based detection of IoT botnet attacks using deep autoencoders," *IEEE Pervasive Computing*, vol. 17, no. 3, pp. 12–22, 2018. DOI: 10.1109/MPRV.2018.03367731

[6] M. A. Salahuddin, V. Pourahmadi, H. A. Alameddine, M. F. Bari, and R. Boutaba, "Chronos: DDoS attack detection using time-based autoencoder," *IEEE Transactions on Network and Service Management*, vol. 19, no. 1, pp. 627–641, 2022. DOI: 10.1109/TNSM.2021.3088326

[7] A. S. Edun, C. LaFlamme, S. R. Kingston, C. M. Furse, M. A. Scarpulla, and J. B. Harley, "Anomaly detection of disconnects using SSTDR and variational autoencoders," *IEEE Sensors Journal*, vol. 22, no. 4, pp. 3484–3492, 2022. DOI: 10.1109/JSEN.2022.3140922

[8] S. Longari, D. H. Nova Valcarcel, M. Zago, M. Carminati, and S. Zanero, "CANnolo: An anomaly detection system based on LSTM autoencoders for controller area network," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1913–1924, 2021. DOI: 10.1109/TNSM.2020.3038991

[9] Q. Li, Z. Wen, Z. Wu, S. Hu, N. Wang, Y. Li, X. Liu, and B. He, "A survey on federated learning systems: Vision, hype and reality for data privacy and protection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, pp. 3347–3366, 2023. DOI: 10.1109/TKDE.2021.3124599

[10] G. Sun, Y. Cong, J. Dong, Q. Wang, L. Lyu, and J. Liu, "Data poisoning attacks on federated machine learning," *IEEE Internet of Things Journal*, vol. 9, no. 13, pp. 11 365–11 375, 2022. DOI: 10.1109/JIOT.2021.3128646

[11] T. Zhang, C. He, T. Ma, L. Gao, M. Ma, and S. Avestimehr, "Federated learning for Internet of Things," in *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*, Coimbra, Portugal, 2021, pp. 413–419. DOI: 10.1145/3485730.3493444

[12] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, 2017, pp. 1273–1282.

[13] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Communications Surveys Tutorials*, vol. 22, no. 3, pp. 2031–2063, 2020. DOI: 10.1109/COMST.2020.2986024

[14] 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, 2021. DOI: 10.1109/TII.2020.3023430

[15] D. Chen, C. S. Hong, Y. Zha, Y. Zhang, X. Liu, and Z. Han, "FedSVRG based communication efficient scheme for federated learning in MEC networks," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 7, pp. 7300–7304, 2021. DOI: 10.1109/TVT.2021.3089431

[16] Y. Liu, S. Garg, J. Nie, Y. Zhang, Z. Xiong, J. Kang, and M. S. Hossain, "Deep anomaly detection for time-series data in industrial IoT: A communication-efficient on-device federated learning approach," *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6348–6358, 2021. DOI: 10.1109/JIOT.2020.3011726

[17] G. Xia, J. Chen, C. Yu, and J. Ma, "Poisoning attacks in federated learning: A survey," *IEEE Access*, vol. 11, pp. 10 708–10 722, 2023. DOI: 10.1109/ACCESS.2023.3238823

[18] Y. Chen, X. Zhu, X. Gong, X. Yi, and S. Li, "Data poisoning attacks in internet-of-vehicle networks: Taxonomy, state-of-the-art, and future directions," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 1, pp. 20–28, 2023. DOI: 10.1109/TII.2022.3198481

[19] S. Huang, Y. Bai, Z. Wang, and P. Liu, "Defending against poisoning attack in federated learning using isolated forest," 2022, pp. 224–229. DOI: 10.1109/ICCCR54399.2022.9790094

[20] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[21] L. Zhu, X. Tang, M. Shen, F. Gao, J. Zhang, and X. Du, "Privacy-preserving machine learning training in IoT aggregation scenarios," *IEEE Internet of Things Journal*, vol. 8, no. 15, pp. 12 106–12 118, 2021. DOI: 10.1109/JIOT.2021.3060764

[22] Z. Chkirbene, A. Erbad, and R. Hamila, "A combined decision for secure cloud computing based on machine learning and past information," in *IEEE Wireless Communications and Networking Conference (WCNC)*, 2019, pp. 1–6. DOI: 10.1109/WCNC.2019.8885566

[23] S. Sahu and B. M. Mehtre, "Network intrusion detection system using J48 decision tree," in *International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2015, pp. 2023–2026. DOI: 10.1109/ICACCI.2015.7275914

[24] M. A. Ambusaidi, X. He, P. Nanda, and Z. Tan, "Building an intrusion detection system using a filter-based feature selection algorithm," *IEEE Transactions on Computers*, vol. 65, no. 10, pp. 2986–2998, 2016. DOI: 10.1109/TC.2016.2519914

[25] Y. Lei, "Network anomaly traffic detection algorithm based on SVM," in *International Conference on Robots Intelligent System (ICRIS)*, 2017, pp. 217–220. DOI: 10.1109/ICRIS.2017.61

[26] C. Yin, S. Zhang, J. Wang, and N. N. Xiong, "Anomaly detection based on convolutional recurrent autoencoder for IoT time series," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 1, pp. 112–122, 2022. DOI: 10.1109/TSMC.2020.2968516

[27] R.-H. Hwang, M.-C. Peng, C.-W. Huang, P.-C. Lin, and V.-L. Nguyen, "An unsupervised deep learning model for early network traffic anomaly detection," *IEEE Access*, vol. 8, pp. 30 387–30 399, 2020. DOI: 10.1109/ACCESS.2020.2973023

[28] D. Novoa-Paradela, O. Fontenla-Romero, and B. Guijarro-Berdiñas, "Fast deep autoencoder for federated learning," *Pattern Recognition*, vol. 143, 2023. DOI: 10.1016/j.patcog.2023.109805

[29] S. Shen, S. Tople, and P. Saxena, "Auror: Defending against poisoning attacks in collaborative deep learning systems," in *Proceedings of the 32nd annual conference on computer security applications*, 2016, pp. 508–519.

[30] D. Li, W. E. Wong, W. Wang, Y. Yao, and M. Chau, "Detection and mitigation of label-flipping attacks in federated learning systems with KPCA and k-means," in *8th International Conference on Dependable Systems and Their Applications (DSA)*, 2021, pp. 551–559. DOI: 10.1109/DSA52907.2021.00081

[31] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Advances in Neural Information Processing Systems*, vol. 30, 2017, pp. 118–128.

[32] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *IEEE Symposium on Computational Intelligence for Security and Defense Applications*, 2009, pp. 1–6. DOI: 10.1109/CISDA.2009.5356528

[33] N. Moustafa and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Military Communications and Information Systems Conference (MilCIS)*, 2015, pp. 1–6. DOI: 10.1109/MilCIS.2015.7348942

Manuscript received 29-05-2024; Accepted 27-12-2024. ■

Ly Vu received the Ph.D degree at Le Quy Don Technical University, Vietnam in 2021 in the major of Mathematic Foundation for Information Technology and M.S. degree at Inha University, South Korea in 2014 in the major of Communication and Information Technology. She is currently a lecturer of Le Quy Don Technical University. Her research interest includes data mining, machine learning, deep learning, network security. E-mail: vu.ly@lqdtu.edu.vn

Tuan Phong Tran received his B.S. degree in Information Technology at Le Quy Don Technical University, Vietnam in 2019. He is currently pursuing the M.S. degree in Information Communication Convergence Technology at Soongsil University, South Korea. His current research interests includes machine learning, deep learning, and edge computing. E-mail: phongtt@lqdtu.edu.vn

Van Cuong Nguyen received his Master's degree in Information Systems from Le Quy Don University of Technology, Vietnam, in 2021. He is currently a Ph.D. candidate at Le Quy Don University of Technology, specializing in Computer Science. His research interests include machine learning, deep learning, and network security. E-mail: cuongpd@lqdtu.edu.vn

Quang Uy Nguyen received his PhD degree at University College Dublin, Ireland in 2011. Currently, he is an Associate Professor and a senior lecturer at Le Quy Don Technical University, Vietnam. Nguyen is also the head of the Computer Network Department and the director of the Intelligent Computing Research Group at Le Quy Don Technical University. His research interest includes Machine Learning, Computer Vision, Information Security, and Evolutionary Algorithms. E-mail:quanguyhn@lqdtu.edu.vn

GIẢM TÁC ĐỘNG CỦA TẤN CÔNG ĐẦU ĐỘC VÀO LƯỢC ĐỒ HỌC LIÊN KẾT TRONG PHÁT HIỆN BẤT THƯỜNG IoTs VỚI KỸ THUẬT TỔNG HỢP CÓ CHÚ Ý

Vũ Ly, Trần Tuấn Phong, Nguyễn Văn Cường, Nguyễn Quang Uy

Tóm tắt

Học liên kết là một phương pháp bảo vệ quyền riêng tư dữ liệu khi huấn luyện mạng nơ-ron sâu trên các thiết bị phi tập trung bằng cách không chia sẻ dữ liệu huấn luyện. Do đó, học liên kết đã được áp dụng phổ biến trong các lĩnh vực như phát hiện bất thường trong mạng Internet vạn vật (IoT). Tuy nhiên, mạng hoặc thiết bị IoT có khả năng tự bảo vệ hạn chế, dẫn đến học liên kết dễ bị tấn công đầu độc dữ liệu. Để giải quyết thách thức này, một lược đồ học liên kết mới được thiết kế để chống lại các cuộc tấn công đầu độc dữ liệu. Phương pháp của chúng tôi, được gọi là Học liên kết sử dụng kỹ thuật chú ý trong tác vụ Tổng hợp (FedAA), sử dụng các mô hình AutoEncoder (AE) để phát hiện bất thường cục bộ trong các mạng IoT. Trong FedAA, việc tổng hợp mô hình toàn cục từ các mô hình cục bộ được thực hiện bằng phương pháp tổng hợp mới, được gọi là Tổng hợp có chú ý (AA). Phương pháp này được thiết kế để giảm thiểu tác động của các cuộc tấn công đầu độc dữ liệu, thường dẫn đến giá trị cao của các hàm mất mát trong các mô hình cục bộ. Chính xác hơn, các mô hình cục bộ có giá trị mất mát cao được gán trọng số chú ý thấp hơn khi đóng góp vào tổng hợp mô hình toàn cục và ngược lại. Do đó, phương pháp AA tăng hiệu quả của FedAA trước các cuộc tấn công đầu độc dữ liệu. Các thí nghiệm về phát hiện bất thường IoT được thực hiện trên ba tập dữ liệu, đó là N-BaIoT, NSL-KDD và UNSW. Kết quả cho thấy FedAA tốt hơn các lược đồ học liên kết khác trong việc giảm thiểu các cuộc tấn công đầu độc dữ liệu.

Từ khóa

Phát hiện bất thường; IoT; tổng hợp có chú ý; học liên kết.