Journal of Information Systems Engineering and Management
2025, 10(37s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Advanced Cybersecurity Framework for Intrusion Detection Utilizing Federated Machine Learning

Mahantesh Laddi[1], Prakash Sonwalkar[2], Dr. Shridhar Allagi[3], Shrikant Athanikar[4],Basavarajeshwari Nadagoudra[5],Nanda Kishore C V[6]

[1]*Assistant Professor, Dept. of CSE, Bharatesh Institute of Technology, Visvesvaraya Technological University, Belagavi, India mahantesh18689@gmail.com.*

[2]*Professor and HOD, Department of Computer Science and Engineering (AIML), Jain College of Engineering and Research, Visvesvaraya Technological University, Belagavi, India prakashksonwalkar@gmail.com.*

[4]*Associate professor, Dept. of CSE, KLE Institute of Technology Hubballi, Visvesvaraya Technological University, Belagavi, India shridharallagi1@gmail.com.*

VSM's Somashekhar R. Kothiwale Institute of Technology Nippani , Visvesvaraya Technological University, Belagavi India athanikar.s@gmail.com

[5]*Assistant Professor, Dept. of CSE, Bharatesh Institute of Technology, Visvesvaraya Technological University, Belagavi, India rajeshwari.nadagoudra@gmail.com.*

[6]*Assistant Professor, Department of Computer Science and Engineering, Nagarjuna College of Engineering and Technology Bangalore ,Nanda_div@ncetmail.com*

** Corresponding author's Email: mahantesh18689@gmail.com*

ARTICLE INFO

Received: 30 Dec 2024

Revised: 19 Feb 2025

Accepted: 27 Feb 2025

ABSTRACT

Cybersecurity risks have grown a lot as the number of devices connected to the Internet of Things (IoT) has proliferated. It's hard for old breach detection systems to keep up with new threats and protect data protection at the same time. This study suggests a way to find intrusions that don't invade privacy by using a Deep Neural Network (DNN) built on Autoencoders and Federated Learning (FL). Our method lets many devices work together to train the model without sharing private information, improving security and privacy. We test the model using standard datasets like RT-IoT 2022, CIC IoT 2023, BoT-IoT, IoT-23, and CIC IoMT 2024. We were able to show that our model can correctly find large-scale threats like DDoS, DoS, DEnial-of-Service, and malware. But it's still hard to spot sneaky attacks like Man-in-the-Middle (MITM), Spoofing, and Command & Control (C&C) because memory rates are so low. Also, FL makes data safer, but it adds extra work to computation and communication, which makes realtime distribution harder. Our results show how essential dataset properties, feature selection, and hyperparameter tuning are for making models work better. Future studies should focus on developing FL to work better with big IoT networks and finding more advanced hacking dangers. Even though there are some problems, our method is a good step toward strong IoT security that protects user privacy.

Keywords: Intrusion Detection, Autoencoder, Deep Neural Network, Federated Learning, IoT Security, Cyber Threats, Machine Learning, Privacy-Preserving AI, Anomaly Detection, Network Security.

1 INTRODUCTION

The increasing dependence on data has exploded the field of cyber security; the foundation of society rests in servers and equipment that all need to be safeguarded from malicious actors. The significant rise in cyber insurance use highlights how much companies depend on cyber security measures as a defence, emphasizing the need to protect our systems and data from malign actors [1]. Recent years have seen the development and use of technology to lessen the dangers associated with cyber security. Software for intrusion detection systems (IDS) is among the leading technologies; it detects cyberattacks using network traffic or internal operations analysis. Historically, these were exhaustive collections of attack signatures analyzing packets and system behaviour to identify criminal activities or cyberattacks; a good example is an attack known as distributed denial of service (DDoS) that overloads a system from several sources with a significant quantity of data. More sophisticated intrusion detection systems allow

Journal of Information Systems Engineering and Management
2025, 10(37s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

anomalybased detection thanks to machine learning algorithms able to recognize abnormal behavior patterns, obstructing the source, and activating alerts. Although signature-based and anomaly-based techniques are used in the field, they are more prevalent since they less frequently result in false intrusion detections (fake positives) that might cause system crashes. Such could cause significant financial damages for a company relying on internet services for revenue. Classification tasks sometimes yield false positives, which point to a model with increased sensitivity. Still, in a complex task like Intrusion Detection Systems (IDS), for instance, false positives are regular; their frequency, however, can be decreased.

These solutions mainly aimed at businesses with extensive IoT-enabled systems subject to hostile actor assaults. Including linked refrigerators, coffee machines, and even toothbrushes, a new age of smart devices has been the explosion of Internet of Things (IoT) gadgets in our houses and businesses. Partly because of the remoteness of IoT devices and partly due to the rarity of attacks, IoT security has historically been a neglected sector inside Cybersecurity. Still, the arrival of cloud storage has less isolation. This shows that IoT security, and therefore IoT IDS research, is an emerging and changing area marked by several deficits and unanswered issues; current research has revealed exposures at every operating level [6]. Particularly considering problematic resource limitations that call for a focus on training latency and memory use, the Internet of Things poses unique problems that demand careful attention. This has started a modern academic initiative to grasp better IoT assaults and a cooperative project for more efficient defences [7].

Despite these solutions, IoT security still faces significant challenges not found in other cybersecurity fields. Another considerable constraint is resource restrictions; disrupting normal operations is unacceptable; hence, firewalls, activity monitors, and IDS platforms cannot use too much memory and processing time. Specific cyberattacks, including Mirai, threaten IoT devices, so the broader field of cybersecurity research calls for the simultaneous improvement of test sets and datasets.

A subfield of machine learning, first created in 2016, is federated learning (FL). Using patient data as a contemporary example gives a wise method for training machine learning models. Instead of sending the data, each data centre develops its model, which is then aggregated. Centralized or decentralized provides two different answers. Using the centralized model, a central &' parent&' node aggregates the parameters from worker nodes to create a master model that is distributed, and the process repeats to improve the model a little bit by bit. Parameters are exchanged and adjusted according to the data of every model using decentralized techniques. As shown in Figure 1, federated learning is perfect for IoT systems since it allows the creation of a complete model using fewer parts of the original data, reducing the resource constraints inherent in IoT [9].

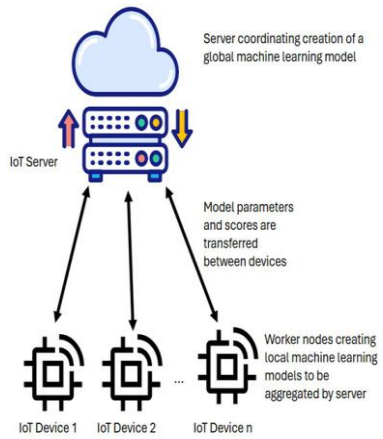


Figure 1. A top-level diagram of federated Learning

As rising cyber threats in IoT networks are tackled, an innovative Intrusion Detection System (IDS) was used in this study. Design melding Autoencoder and Federated Learning using a Deep Neural Network (DNN). Through Autoencoder, the feature extraction and anomaly detection capabilities find latent patterns that point to cyber threats. The aggregated features subsequently instruct a DNN classifier, improving threat detection accuracy. By

Journal of Information Systems Engineering and Management
2025, 10(37s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

enabling distributed training among IoT devices, federated Learning improves security, respects data privacy, and reduces demand on computing resources. Maximizing efficiency, the combined Autoencoder, DNN, and FL model provides a workable and expandable way to identify cyber threats in the Internet of Things environments. By tackling the critical concerns of false positives, resource restrictions, and privacy fears, this research helps to advance the discipline of IoT security. Through thorough testing and assessment, the suggested IDS system seeks to offer a very adaptable and effective defence system, raising the security resilience of current IoT networks.

2 RELATED WORK

This part discusses what recent studies on machine learning and federated learning in IoT networks and attack detection have added and what hathey'haven'tota et al. [10] suggested MV-FLID. This FL-based intrusion detection method learns from different Decentralized views of IoT network data used to identify, categorize, and prevent breaches. The multi-perspective group learning part makes learning how to use other types of attacks easier. Peer learning is used by the FL function to successfully combine profiles, as the data from the device is not sent to the computer. Still, their study didn't look into uncontrolled and reinforcement machine learning systems, which could help with intrusion detection by finding attacks that haven't been taught. An idea from Tabassum et al. [11] to find cyber risks in intelligent e-healthcare systems, smart cities, and smart homes, among other Internet of Things (IoT) systems, was to use federated Deep Learning (DL) and a network analysis network (GAN) to create FEDGAN-IDS. They sent it to IoT devices so the GAN network could learn from more local data and be a predictor. They demonstrated that their model outperformed the majority of standalone IDS in terms of performance and convergence time. By checking the model's correctness and convergence. In their article [12], Driss et al. described a method to use federated Learning to find hacks in in-vehicle sensor networks (VSNs).

The suggested shared learning method lets computers share their resources and makes training with devices easier. The proposed method uses a Random Forest (RF)-based ensemble unit and a collection of Gated Recurrent Units (GRU) to improve attack recognition. Du et al. [13] pointed out that IoT devices in cars have sensors that produce information unique to the device and could damage its security if made public. In joint driving, more people need to use GPS, cams, radar, and other similar technologies. The writers suggested using shared Learning to make the system safer and faster by connecting many IoT devices to large networks. To protect the whole healthcare system network, Ghourabi et al. [14] developed a novel method to find intrusions and viruses. The suggested method has two parts: a malware detection system for medical staff data servers and other devices they utilize and an intrusion detection system (IDS) for medical equipment linked to the hospital network. The intention was to preserve the whole network. Safe, no matter what computers and gadgets were on it. The problem is that it needs to be installed on many different computers across the hospital network. To put all the data together, you need a matching process. It says in [15] that People are worried about their safety because IoT devices are increasingly used in healthcare. Machine learning is used by a low-power Intrusion Detection System (IDS) to group assaults such as Distributed Denial of Service (DDoS) and Man-in-the-Middle (MitM) so that data doesn't leave without permission.

Threats that haven't been seen before are found by anomaly recognition, which instructs the machine learning model to recognize them once again in the future. Fed Learning (FL) is a way to keep information secret and safe. The Enhanced Random Forest method makes predictions for the first step. After that, the Support Vector Machine (SVM) technique with one class is used to find strange data points. The Federated Learning approach combines the overall system model up to date. A Raspberry Pi MIoT gateway and an IoT system were used to test the method. It consistently found known and new threats while using as few resources and energy as possible.

Table 1: Summary Of the existing studies

References	Dataset	Algorithm	Findings	Accuracy
[16]	CICIDS2017	GRU	FL-based IDS addresses network intrusion detection and privacy issues, allowing ISPs to conduct deep learning training while preserving local data. However, no real-world	0.98%

Journal of Information Systems Engineering and Management
2025, 10(37s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

			scenario simulations were carried out to demonstrate its viability.	
[17]	MOST	**CNN**	The study explores using the Lagrange multiplier technique to optimize learning performance and resources in wireless FL systems, with the goal of saving communication costs.	N/M
[18]	Edge_IIoTset	CNN, RNN, Fed Avg Algorithm	The study suggested a FL method for IIoT intrusion detection that uses centralized CNN and RNN machine learning and FL, and evaluated using Dataset Edge-IIo Tset for performance.	92.49%
[19]	KDD – Cup 1999	LSTM-KPCA	This study uses recurrent deep learning models to offer an end-to-end model for detecting and categorizing network attacks.	0.98%
[20]	NIDS datasets	Stacked unsupervised FL using deep encoder	The study presents a versatile approach using stacked unsupervised federated Learning for flow-based NIDS, incorporating a deep autoencoder, energy flow classifier, and ensemble learning for effective intrusion detection.	0.98%
[21]	NSL-KDD Dataset	FL-MA	This synergistic search enhances IoT and WSN security by utilizing machine learning and the Firefly Altically.	0.992%
[22]	WSN-DS and CIC-IDS-2017 datasets.	FL-SCNN-Bi-LSTM	Federated Learning assisted Hybrid Stacked CNN model with Bi-directional LSTM is proposed	0.999%

Existing work in FL-based IDS in IoT networks is primarily for supervised Learning, while unsupervised and reinforcement learning methods remain little explored, promising better detection for new attacks. Scalability and communication overhead remain issues as not many works seek to maximize the FL model's updates to reduce

Journal of Information Systems Engineering and Management
2025, 10(37s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

resource constraints. Existing models are non-real-time and threat-evolution-specific, utilizing fixed datasets that potentially do not generalize well. The use of auto encoders within FL-based IDS is minimal, even though they have the capability for better anomaly detection. Cross-domain generalization is also poor, as most studies only test models on one dataset, which decreases robustness across varied IoT environments. Finally, security threats like adversarial attacks and model poisoning in FL systems require more effective mitigation measures while being efficient and having accurate detection.

3 BACKGROUND

Cyber-attacks have become more likely as IoT (Internet of Things) networks and edge computing are used more in key infrastructure. Intrusion detection systems (IDS) are used to safeguard these. Settings. Centralized IDS models get data from many places and train one model on a central computer. However, these models also have big privacy problems, especially when data privacy is paramount, like healthcare, banking transactions, and industrial IoT (IoT). Federated Learning (FL) is an autonomous method that lets many devices learn a standard worldwide model without transmitting their local information to a central computer. In intruder detection, FL lets people work together to learn on multiple devices while keeping their data private. Still, strict feature extraction is needed to successfully handle high-dimensional attack patterns and network data. In this situation, Auto encoders (AE) and (DNNs) work together to make finding known and new threats easier while still using computers efficiently.

3.1 Auto encoder (AE)

The neural network type known as an auto encoder (AE) design is meant to capture a small representation of input data to enable the reconstruction of the original inputs [23]. As illustrated in Figure 2, the Auto encoder (AE) comprises a hidden latent vector, a bottleneck layer, an encoder network, and a decoder network. Many hidden layers mark a deep learning model, a stacked autoencoder, in the two encoder and decoder parts.

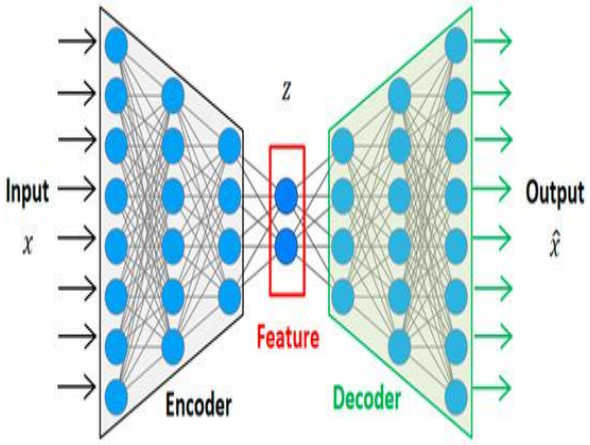


Figure 2 . General structure of Autoencoder (AE).

An auto encoder: The following equations may be used to define the encoder and decoder using a single, hidden layer. [24]:

$$z = \sigma(W_{encoder}x + b_{encoder}) \quad (1)$$

$$\bar{x} = \sigma(W_{decoder}z + b_{decoder}) \quad (2)$$

In this context, W indicates the weight and b the bias parameters, s the bottleneck low-dimensional feature space, x the input vector (benign or normal data), and σ an activation function.

Using Equation (1), the encoder network transforms the input features to low-dimensional representations; the decoder network uses Equation (2) to reconstruct the original data x from the projected low-dimensional features z.

As defined in [25], The primary goal of creating the AE model is to reduce the reconstruction error (RE) function, quantifying the variation between the model's input and output.

Journal of Information Systems Engineering and Management
2025, 10(37s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

$$L(x, \bar{x}) = \|x - \bar{x}\|_2^2 \qquad (3)$$

where $\|\cdot\|_2$ denotes the Euclidean norm. Upon acquiring and determining a threshold value, the AE model from the training dataset may be applied to the validation dataset. The actual validation data's mean square error (MSE) x_{val}, as well as the anticipated validation data $\hat{x}_{val}$, is computed as:

$MSE_{val} - 1/N \sum_N (x_{val} - \bar{x}_{val})^2$ (4)

N denotes the number of validation samples. The sample mean and standard deviation are used to determine the threshold value that separates normal from attack data as [26]:

$Threshold = mean(MSE_{val}) + std(MSE_{val})$ (5)

Once trained, the encoder extracts latent features h, which are then entered for classification into a Deep Neural Network (DNN).

Deep Neural Network (DNN)

The deep neural network (DNN) was first proposed for a class of deep probabilistic models called Deep Belief Networks (DBNs) by [27]. One type of neural network, Multiple layers of Restricted Boltzmann Machines (RBMs), makes up Deep Belief Networks (DBNs). The network's design consists of two layers, with no connections across units within the same layer from binary stochastic variables v that are visible and h that are concealed.

There are several designs for deep learning, but we concentrate on the feedforward architecture in Figure 3 [28]. The central unit of the model is the several levels of connected neurons in a feedforward network. Review a training dataset comprised of N occurrences, denoted as $\{(x^1, y^1), (x^2, y^2), \ldots, (x^n, y^n)\}$.

The input vector $\vec{x}$ is the vector of features consisting of the probability of the bit-symbol "1," whereas y represents the class label assigned to each instance.

During the training phase $\vec{x}$, the input vector moves across the network's visible nodes, where $\vec{w}$ The Deep Belief Network (DBN) 's starting weights provide them. Our goal is to reduce the cost function. C:

$C(\vec{w}; \vec{x}, y) = \frac{1}{2}\|h_w(\vec{x}) - y\|^2$ (6)

where $h_w\left(\vec{x}\right)$ the hypothesis function produces an estimated output. The whole amount spent is shown as follows:

$$C(\vec{w}) = \frac{1}{N}\sum_N C\left(\vec{w}; \overrightarrow{x^n}, y^n\right) \frac{\lambda}{2} \sum_k^K \sum_i^{L_m} \sum_j^{L_{m+1}} \left(w_{ij}^k\right)^2 \qquad (7)$$

Let K denote the network's depth; L_m represents the number of nodes in the m-th layer, and $w_{ij}^k \in \vec{w}$ signifies the weight of the edges connecting node i in layer k − 1 to node j in layer k. Consequently, to minimize the entire cost function, we determine the parameter set $\vec{w}^*$ as follows:

$$\vec{w}^* = \underset{w}{\operatorname{argmin}} C(\vec{w}) \qquad (8)$$

They $\vec{w}$ can be acquired by the backpropagation process, wherein the weight vectors are adjusted from the top layer to the bottom layer utilizing the following Equation:

$$w_{ij}^k = w_{ij}^{k-1} + \zeta \frac{\partial}{\partial w_{ij}^{k-1}} C(\vec{w}) \qquad (9)$$

where ζ is an adaptation parameter.

Journal of Information Systems Engineering and Management
2025, 10(37s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

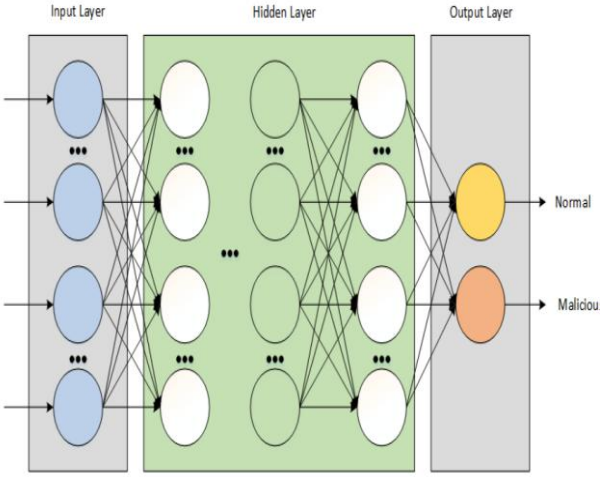


Figure 3.Deep neural network architecture for anomaly-based IDS

3.2 Federated Learning with Autoencoder and DNN

The Autoencoder (AE) combined with Deep Neural Network (DNN) architecture in Federated Learning leverages the unsupervised feature learning capabilities of autoencoders alongside the classification efficacy to establish an efficient and privacy-enhancing intrusion detection system (IDS). The federated technique allows several IoT devices or edge nodes to locally train models and transmit just the acquired parameters to a central server, lowering data security threats. Exposure.

Let $X \in$ □ ^{nimsed} represent the dataset, where n is the number of samples and d is the feature dimension. The Autoencoder consists of an encoder f_θ and a decoder g_ϕ, where:

$$h = f_\theta(x) = \sigma(W_e x + b_e)\bar{x} = g_\phi(h) = \sigma(W_d h + b_d) \qquad (10)$$

Here, W_e, b_e $W_d b_d$ denote the adjustable weight and bias parameters while signifying an activation function, such as ReLU or Sigmoid. The Autoencoder aims to reduce the reconstruction loss.

$$L_{rec} = \|x - \bar{x}\|^2 \qquad (11)$$

The latent representation h obtained from the encoder is then used as input to the DNN classifier, where:

$$y = F_\psi(h) = soft\max(W_h h + b_h) \qquad (12)$$

Where W_h, b_h are the trainable parameters of the classifier, and y represents the predicted output. The classification loss is given by:

$$L_{cls} = -\sum_i y_i \log(\bar{y}_i) \qquad (13)$$

where y_i [?] is the actual class designation and $\bar{y}_i$ [?] is the anticipated likelihood.

Each client k locally trains the AE-DNN model in the federated setting using its own Dk dataset. The updated local model is computed as follows:

$$w_k^{t+1} = w_k^t - \eta \nabla L_k \left(w_k^t \right) \qquad (14)$$

Where η is the learning rate, and $\nabla L_k \left(w_k^t \right)$ is the gradient of the loss function concerning the model parameters at client k? The central server aggregates the local updates. Using Federated Averaging (FedAvg):

Journal of Information Systems Engineering and Management
2025, 10(37s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

$$w^{t+1} = \sum_{k=1}^{K} \frac{n_k}{N} w_k^t \qquad (15)$$

Where K is the total number of clients, n_k represents how many local samples each customer has k, and $N = \sum_k n_k$ is the size of the whole dataset for every client. This repeated aggregation improves the global model parameters and is then dispersed among the clients for further local training.

4 METHODOLOGY

4.1 Introduction to Methodology

It uses a Federated Learning (FL) model, including an Autoencoder and Security for the Internet of Things (IoT) using Deep Neural Networks (DNN). The approach is designed to efficiently detect cyber risks from various IoT datasets while preserving data privacy. The strategy comprises many essential stages: dataset preparation, model design, FL-based training, performance metric evaluation, and comparative FL analysis with and without Privacy-Preserving (PP) methods. Federated Learning preserves sensitive IoT data on local devices while facilitating collaborative model training, making it the best choice for widespread IoT in the real world. Settings.

4.2 Dataset Description

Using seven opensource IoT security datasets—BoTIoT, IoT23, TONIoT, CIC IoT 2023, CIC EdgeIIoT, RTIoT 2022, and IoMT 2024—this study assesses our Federated Learning (FL) model's efficiency. Included in the datasets are thorough network traffic records and many different sorts of assaults: Malware, Injection, Spoofing, ManintheMiddle (MITM), and Distributed denial of service (DDoS). The kind, size, formatting, and nature of the attack on the datasets offer our model a complete assessment terrain. A thorough data cleaning process ensures consistency and quality across all data sets before training. Data cleaning, imputation of missing values, feature encoding, and normalization are done. Preservation of data integrity calls of disappeared values to be replaced with zeroes; categorical data—including protocol kinds and attack labels—are transformed into numerical values with the help of the LabelEncoder() from the Scikitlearn library. Normalizing numerical elements to a [0,1] range guarantees even contributions to the learning process, and MinMaxScaler() is the tool for that. Spurious characteristics are removed in sets containing null or duplicate columns to reach the best model performance. We refrain from synthetic undersampling or oversampling to maintain the actual distribution of attacks due to the class imbalance in certain sets where some attacks have significantly fewer samples than others. Consequently, the model is taught on unbiased actual IoT traffic patients; we can comprehensively assess the generalization power of FL-based models in different risk settings through the variance of these data set settings.

4.3 Model Architecture and Techniques Used

Our approach utilizes a Federated Learning (FL) framework integrated with an Autoencoder and Deep Neural Network (DNN) model to detect IoT risks. This hybrid approach facilitates distributed Learning while preserving data security, which makes it perfect for extensive IoT settings where data privacy is crucial. In the following sections, we delineate each setup component, elucidate its purpose, and demonstrate how it contributes to the system's optimal functionality.

4.3.1 Autoencoder for Feature Extraction

An unsupervised autoencoder learning model extracts meaningful patterns from network traffic by learning the characteristics of regular and attack traffic. It consists of a Decoder that reconstructs the input from the lower-dimensional latent representation created by the encoder, compressing the input characteristics. Anomalies, such as cyberattacks, typically exhibit higher reconstruction errors. The architecture includes an Input Layer for preprocessed IoT data, multiple Hidden Layers in the encoder using ReLU activation to reduce dimensionality, a Decoder that enlarges the Latent Space Representation, which captures key characteristics, and a feature vector back to its original form through a mirror network structure. The Output Layer reconstructs the input, with reconstruction errors as an anomaly detection metric. Loss Function: We use Mean Squared Error (MSE) to measure reconstruction loss:

Journal of Information Systems Engineering and Management
2025, 10(37s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2$$

Where x_i is the first input, and $\hat{x}_i$ has the output been reassembled.? A high reconstruction error indicates an anomaly.

4.4 Deep Neural Network (DNN) for Classification

After extracting latent features with the Autoencoder, a Deep Neural Network (DNN) classifies network traffic into standard, DDoS, MITM, and Spoofing. The Input Layer receives the encoded feature representation, followed by multiple Hidden Layers with fully connected neurons and ReLU activation to capture complex patterns. Batch Normalization stabilizes learning and accelerates convergence, while Dropout Regularization (e.g., 30%) mitigates overfitting. The Output Layer employs SoftMax activation to assign probability scores to each attack category, enabling precise categorization into many classes. Loss Function: We use Categorical Cross-Entropy Loss for multi-class categorization.:

$$Loss = -\sum_{i=1}^{C} y_i \log(\hat{y}_i)$$

Where y_i is the actual class label, and $\hat{y}_i$ what is the predicted probability.

4.4.1 Federated Learning (FL) Framework

Traditional centralized architectures aggregate raw data on a single server, resulting in significant transmission costs and privacy problems. Federated Learning (FL) allows IoT devices to train models locally and communicate just model updates to solve this problem. The FL workflow begins with Each device using its dataset to train the Autoencoder + DNN in a process known as local training. Instead of transmitting raw data, Model Updates (weights and biases) are sent to a central server. The server uses the Federated Averaging (FedAvg) method, weighted by the quantity of the client data, to aggregate these updates using Federated Aggregation. After that, customers will receive a new distribution of the Global Model Update for the next training session. Ensuring efficient, privacy-preserving learning across decentralized IoT environments.

$$w_t = \sum_{i=1}^{K} \frac{n_i}{N} w_t(i)$$

Where w_t is the global model n_i , how many samples does the customer ha *i* ve, and N is the sum of all samples for all customers?

4.4.2 Privacy-Preserving Techniques in FL

Since in federated learning, clients communicate with a central server. It is vulnerable to data leakage and adversarial attacks. To mitigate this, we incorporate Privacy-Preserving (PP) strategies like:

1. Differential Privacy (DP): Adds controlled model updates to model noise before forwarding them to the server, preventing adversaries from inferring sensitive details.

- Mathematical Formulation: Given a function $f(x)$ that maps data x to a model update, we apply noise η drawn from a Laplace or Gaussian distribution:

$$f(x) + \eta \sim N(0, \sigma^2)$$

- Ensures ε -differential privacy, where ε controls the privacy level.

2. Secure Aggregation: Encrypts model updates before transmission so the central server only sees aggregated results, preventing it from reconstructing individual client data.

Journal of Information Systems Engineering and Management
2025, 10(37s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

- Calculations on encrypted values may be performed without decryption thanks to homomorphic encryption.

3. Federated Dropout: Randomly drops specific neurons during training at different clients, making the original data more difficult for attackers to recreate.

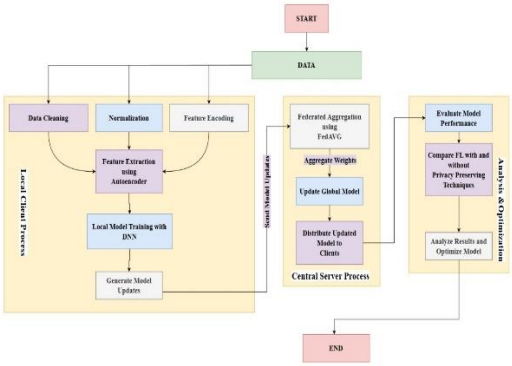


Figure 4 Flow-chart

4.5 Experimental Setup & Training Process

This part explains how we set up our computers, configured the training, and put the Federated Learning (FL) system into action with an Autoencoder + Deep Neural Network (DNN) model. Our experiment has several steps: spreading the data, getting the model ready, training it, bringing everything together, and checking how well it works. Here's a full breakdown of how we set up and trained the system.

4.5.1 Computational Environment

Researchers carried out tests using a powerful computer system and several spread-out IoT edge points to mimic real-world FL situations. The setup details for both hardware and software are as follows:

Component	Specification
Central Server	Windows 10 Pro, Intel i7-1987H (8 cores, 2.3 GHz), 64GB RAM, RTX 2080, 2TB SSD
FL Clients	Raspberry Pi 4 (4GB RAM), ESP32 microcontrollers, AWS EC2 (t2.large, 8GB RAM)
Programming	Python 3.9.12
Deep Learning	TensorFlow 2.11.0, Keras 2.11.0
Data Processing	NumPy 1.23.5, Pandas 1.5.2, Scikit-learn 1.0.2
Federated Learning	TensorFlow Federated (TFF)
Privacy Techniques	PySyft (Secure Aggregation, Differential Privacy)

4.5.2 Data Distribution & Preprocessing

To mimic real-world decentralized IoT networks, datasets were partitioned across FL clients based on different IoT environments. Each client receives a unique subset of data to simulate independent local datasets.

4.5.2.1 Data Partitioning Strategy

- IID (Independent and Identically Distributed): Data was evenly distributed across clients, ensuring each device had a balanced dataset.
- Non-IID (Non-Identically Distributed): Some clients received only specific attack types to simulate real-world IoT security scenarios with localized attack patterns

Journal of Information Systems Engineering and Management
2025, 10(37s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

4.5.2.2 eprocessing Steps at Each Client

No under-sampling or over-sampling techniques were applied across the seven datasets to ensure model robustness and prevent overfitting. The BoT-IoT dataset (73M records) was cleansed to remove inaccuracies, convert erroneous entries to integers or 0, and encode categorical data using LabelEncoder (). TON-IoT (461K records) required only label encoding for consistency. IoT-23 (325.3M records) was cleaned by replacing string values in numerical columns with 0, removing empty columns, and excluding low-sample attacks. CIC IoT 2023 (46.7M records) was already preprocessed, including feature extraction and aggregation, while web-based and brute-force attacks were excluded. CIC IoMT 2024 (9M records) underwent minor preprocessing, including missing value imputation, label encoding, and MinMaxScaler () for normalization. RT-IoT 2022 (1.5M records) involved converting hex values, replacing string values with 0, removing empty columns, and applying label encoding and MinMaxScaler (). EdgeIIoT (2.5M records) was refined by resolving inconsistencies, filling missing values, renaming columns, and applying LabelEncoder () and MinMaxScaler () for standardization.

4.5.3 Model Training in Federated Learning

4.5.3.1 Local Model Training at Each Client

Each IoT client trained the Autoencoder + DNN model locally using the Adam optimizer with a batch size of 0.001 (learning rate) 64 and 5 epochs per training round. The Autoencoder used Its loss function as the Mean Squared Error (MSE), while the DNN classifier employed Categorical Cross-Entropy. To enhance model generalization, Dropout (0.3 probability) was applied to prevent overfitting, and Batch Normalization ensured stable training. Clients sent just their model parameters—weights and biases—to the central FL server after local training. for aggregation, maintaining data privacy while improving global model performance.

4.5.4 Federated Learning Aggregation Strategy

At the central server, the local model updates from each client are combined using the Federated Averaging (FedAvg) approach. Customers are then given access to the updated global model for the next training cycle.

FedAvg Algorithm:

$$w_t = \sum_{i=1}^{K} \frac{n_i}{N} w_t^{(i)}$$

Where:

- w_t = updated global model parameters
- n_i = quantity of samples at customer iii
- N = aggregate quantity of samples from all clients

4.5.5 Privacy-Preserving Mechanisms During FL Training

To enhance security and privacy, we implement the following privacy-preserving techniques (PP):

4.5.5.1 Differential Privacy (DP)

- Noise Injection: Each client adds random introduced noise to model changes before transmission to the server.
- Mathematical Formula:

$$f(x) + \eta \sim N(0, \sigma^2)$$

Where η Gaussian noise ensures ε differential privacy.

Journal of Information Systems Engineering and Management
2025, 10(37s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

4.5.5.2 Secure Aggregation

- Homomorphic Encryption: Ensures that client updates remain encrypted during aggregation.
- It prevents attackers from reconstructing individual client updates.

4.5.6 Performance Evaluation

After training, the global model is evaluated on an independent test set using multiple performance metrics. Classification metrics include Accuracy (A) for overall correctness, Precision (P) for correctly predicted attack instances, Recall (R) for detecting actual attacks, and F1-Score (F1) to balance precision and recall. Additionally, Federated Learning metrics assess the training efficiency, including Local Accuracy (L-A) before aggregation, Federated Accuracy (F-A) after aggregation, Client Computation Time (CCT) for local training, and Time to Receive Federated Data (TFD) for server-client communication. These metrics guarantee a thorough assessment of the efficacy of categorization and federated learning efficiency.

5 RESULTS

Here, we evaluate our Federated Learning's (FL) performance. Platform combined with the Autoencoder + Deep Neural Network (DNN) model over various IoT security datasets. The outcomes are evaluated based on classification F1 score, memory, accuracy, precision, computational efficiency, and the effect of privacy-preserving (PP) technology. To see how they affect detection performance and system overhead, we also evaluate the efficacy of FL with and without PP systems in the context of with and without PP systems. Regarding detecting different cyber threats—from DDoS to Spoofing to MITM to Malware assault—across many IoT environments, this appraisal shows our approach's advantages and drawbacks. The following sections offer a thorough analysis of the experimental data.

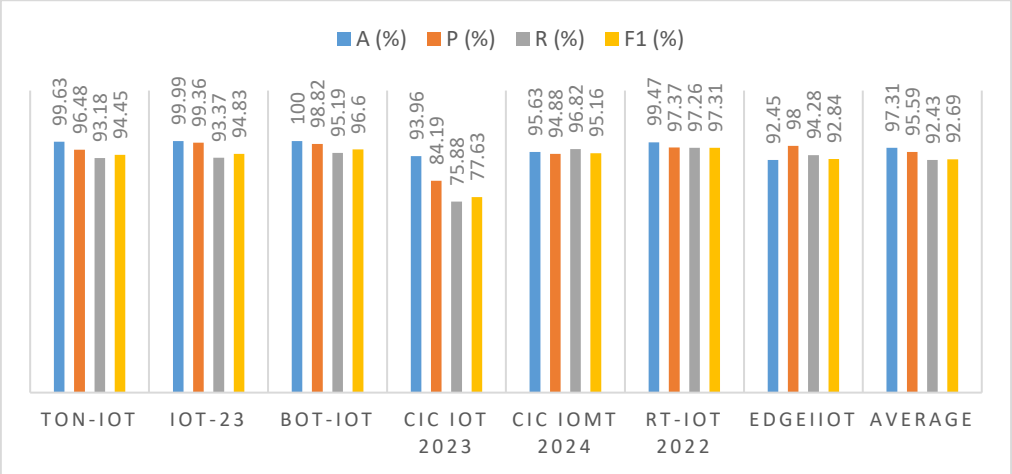
Datasets	A (%)	P (%)	R (%)	F1 (%)
TON-IoT	99.63	96.48	93.18	94.45
IoT-23	99.99	99.36	93.37	94.83
BoT-IoT	100.00	98.82	95.19	96.60
CIC IoT 2023	93.96	84.19	75.88	77.63
CIC IoMT 2024	95.63	94.88	96.82	95.16
RT-IoT 2022	99.47	97.37	97.26	97.31
EdgeIIoT	92.45	98.00	94.28	92.84
Average	97.31	95.59	92.43	92.69

Table 10. Overall Results for the Seven IoT Datasets

Across all datasets, the Autoencoder + DNN shows outstanding accuracy (often above 99%) for most common attacks, especially DDoS, DoS, and malware-based threats. However, there are notable challenges in detecting MITM, Spoofing, and C&C attacks, which tend to be more subtle and rely on deception rather than high-volume traffic patterns. The lower recall values for these attacks suggest that false negatives (undetected attacks) remain an issue. This could be mitigated by improving feature selection, fine-tuning hyperparameters, and incorporating more contextual learning mechanisms to enhance the model's capacity for detecting stealthy threats. Additionally, handling class imbalance more effectively—especially for rare attack types—would help reduce misclassification rates and improve recall scores for difficult-to-detect threats. Overall, while the Autoencoder + DNN approach demonstrates a strong potential for real-world IoT security applications, further optimizations are necessary to improve recall rates for stealthy attacks, ensuring comprehensive security coverage for modern IoT environments.

Journal of Information Systems Engineering and Management
2025, 10(37s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article



5.1 Federated Learning Performance

			FL		Autoencoder + DNN	
FL	Dataset	I	CCT	TFD	L-A (%)	F-A (%)
C1	TON-IoT	9	7.2 s	27.8 s	98.24	98.34
C2		9	7.2 s	30.2 s	98.15	98.34
C3		9	7.2 s	37.2 s	98.03	98.34
C4		10	10.2 s	35.0 s	98.23	98.21
Average		9.25	7.9 s	32.6 s	98.16	98.31
C1	IoT-23	4	20:03.0 s	20:23.6 s	99.98	99.99
C2		7	17:06.3 s	17:29.3 s	99.99	99.99
C3		7	17:06.3 s	17:36.3 s	99.99	99.99
C4		3	08:46.5 s	09:11.5 s	99.99	99.98
Average		5.25	15:45.6 s	16:10.2 s	99.99	99.99
C1	BoT-IoT	2	9:45.2 s	10:05.8 s	99.99	99.99
C2		2	9:45.2 s	10:08.2 s	99.99	99.99
C3		2	9:45.2 s	10:15.2 s	99.99	99.99
C4		2	9:45.2 s	10:10.2 s	99.99	99.99
Average		2	9:45.2 s	10:09.8 s	99.99	99.99
C1	CIC IoT 2023	4	1:16.1 s	1:36.7 s	82.60	81.23
C2		2	2:09.8 s	2:32.8 s	81.12	80.01
C3		4	1:16.1 s	1:46.1 s	82.40	81.23
C4		2	2:09.8 s	2:34.8 s	80.04	80.01
Average		3	1:43 s	2:07.6 s	81.54	80.62
C1	CIC IoMT 2024	3	3.57 s	3.18 s	87.77	87.57
C2		2	4.74 s	4.57 s	77.07	76.87
C3		4	5.13 s	5.13 s	88.12	88.14
C4		4	7.43 s	8.08 s	88.33	88.14
Average		3.25	5.22 s	5.24 s	87.82	87.68
C1	RT-IoT 2022	7	22.32 s	53.43 s	98.43	98.18
C2		6	23.88 s	55.3 s	98.73	98.23
C3		7	22.35 s	55.92 s	98.74	98.24
C4		8	26.87 s	57.76 s	98.16	98.22
Average		7	23.85 s	55.59 s	98.5	98.22
C1	EdgeIIoT	6	44.91 s	55.03 s	99.79	99.84
C2		2	1:06 s	1:29.3 s	99.67	99.27

Journal of Information Systems Engineering and Management
2025, 10(37s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

C3		3	59.43 s	1:29.12 s	99.77	99.67
C4		6	54.43 s	59.98 s	99.87	99.84
Average		4.25	54.70 s	1:13.36 s	99.77	99.66

Table 11. Performance without PP Techniques

Evaluating Federated Learning (FL) without privacy-preserving (PP) methods across multiple IoT security datasets demonstrates high classification accuracy and reasonable computational efficiency, though performance varies based on dataset complexity. Local (L-A) and federated accuracy (F-A) exceed 98% in datasets like BoT-IoT, IoT-23, TON-IoT, RT-IoT 2022, and EdgeIIoT, highlighting FL's robustness in distributed cybersecurity. However, CIC IoT 2023 and CIC IoMT 2024 show lower accuracy (80–88%), suggesting challenges in handling diverse or evolving attack patterns.TON-IoT achieves 98.16% L-A and 98.31% F-A with an average of 9.25 iterations per client and a low CCT (7.9s), though TFD is relatively high (32.6s), indicating possible network latency. IoT-23 demonstrates near-perfect accuracy (99.99%) but with significant computational demands (CCT: 8:46.5–20:03.0s; TFD: 9:11.5–20:23.6s). BoT-IoT achieves 99.99% accuracy with minimal computational burden (2 iterations per client, CCT: 9:45.2s), suggesting an efficient dataset structure.In contrast, CIC IoT 2023 has the lowest F-A (80.62%) and L-A (81.54%), with CCT exceeding 2 minutes in some cases, indicating the complexity of attack patterns. CIC IoMT 2024 achieves 87.68% F-A with an average CCT of 5.2s, suggesting efficiency but lower classification accuracy. RT-IoT 2022 maintains high accuracy (>98%) but experiences communication delays (CCT: 22.32–26.87s, TFD: 55.59s), likely due to large data transfers. EdgeIIoT achieves 99.66% F-A, but client processing times vary widely, with some requiring over a minute per iteration, highlighting potential deployment challenges.FL performs exceptionally well in datasets with well-defined attack patterns (e.g., IoT-23, BoT-IoT, RT-IoT 2022, EdgeIIoT), achieving near-perfect accuracy with minimal iterations. However, performance declines in CIC IoT 2023 and CIC IoMT 2024 due to more complex attacks, adversarial behaviour, or class imbalances. Additionally, datasets requiring extensive processing (e.g., IoT-23, RT-IoT 2022) may pose challenges for real-time applications. While FL proves effective for IoT security, further optimization in communication, model tuning, and handling complex attack scenarios is necessary to reduce processing time and enhance classification accuracy in challenging datasets.

			FL		Autoencoder + DNN	
FL	Dataset	I	CCT	TFD	L-A (%)	F-A (%)
C1	TON-IoT	9	9.3 s	31.8 s	98.06	98.26
C2		8	12.5 s	36.8 s	98.09	98.03
C3		9	9.6 s	41.2 s	98.02	98.26
C4		9	8.6 s	36.2 s	98.2	98.26
Average		8.75	10.0 s	36.5 s	98.09	98.20
C1	IoT-23	7	24:33.4 s	24:54.7 s	99.99	99.99
C2		2	8:07.0 s	08:31.2 s	99.98	99.97
C3		7	24:33.7 s	25:04.1 s	99.99	99.99
C4		2	8:07.8 s	8:33.2 s	99.98	99.97
Average		4.5	16:20.5 s	16:45.8 s	99.98	99.98
C1	BoT-IoT	3	10:30.1	10:52.6 s	99.99	99.99
C2		3	10:30.4 s	10:55.0 s	99.99	99.99
C3		3	10:30.6 s	11:02.0 s	99.99	99.99
C4		3	10:29.9 s	10:57.0 s	99.99	99.99
Average		3	10:30.2 s	10:56.6 s	99.99	99.99
C1	CIC IoT 2023	4	1:47.1 s	2:09.9 s	81.97	81.82
C2		3	2:17.6 s	2:42.2 s	81.49	81.44
C3		4	1:47.4 s	2:19.3 s	81.37	81.82
C4		4	1:47.0 s	2:14.3 s	82.81	81.82
Average		3.75	1:54.8 s	2:21.4 s	81.91	81.72
C1	CIC IoMT 2024	4	3.34 s	3.58 s	86.4	87.44
C2		2	4.40 s	4.11 s	76.96	75.74

Journal of Information Systems Engineering and Management
2025, 10(37s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

C3		4	5.34 s	6.07 s	87.14	87.44
C4		3	7.42 s	8.11 s	87.6	87.58
Average		3.25	5.87 s	5.46 s	87.72	87.05
C1	RT-IoT 2022	5	25.76 s	56.43 s	98.07	98.20
C2		8	27.2 s	55.98 s	98.84	98.34
C3		8	22.54 s	1:02.91 s	98.28	98.31
C4		7	28.98 s	57.89 s	98.44	98.03
Average		7	26.12 s	56 s	98.41	98.22
C1	EdgeIIoT	3	42.12 s	59.12 s	99.73	99.87
C2		2	1:07.41 s	1:39.52 s	99.6	99.31
C3		2	1:10.61 s	1:36.98 s	99.46	99.34
C4		3	44.55 s	1:03 s	99.77	99.8
Average		2.5	58.3 s	1:16.41 s	99.64	99.55

Table 12. Performance with PP Techniques

This study examines the performance of Federated Learning (FL) combined with an Autoencoder and Deep Neural Network (DNN) across multiple IoT datasets while employing privacy-preserving (PP) techniques. The analysis considers key performance metrics, including client computation time (CCT), time to receive federated data (TFD), local accuracy (LA), and federated accuracy (FA), to evaluate FL's scalability and efficiency in real-world IoT security applications. Results indicate that FL consistently achieves high accuracy, with datasets such as IoT-23, BoT-IoT, and EdgeIIoT reaching 99.99% in LA and FA, demonstrating FL's ability to learn complex attack patterns while preserving data privacy. However, computational overhead varies significantly and is influenced by dataset complexity. The IoT-23 dataset, for instance, incurs substantial processing demands, with an average CCT exceeding 16 minutes, whereas BoT-IoT demonstrates high efficiency with just 10.5 seconds for CCT and TFD. This highlights the impact of dataset properties on FL's computational performance. IoT 2023 and CIC IoMT 2024 show lower accuracy (80–87%), suggesting that FL may struggle with newer datasets containing diverse or sophisticated attack patterns. Factors such as class imbalance, unseen attack vectors, and high-dimensional feature spaces may contribute to this decline. RT-IoT 2022 and EdgeIIoT maintain high accuracy (>98%) but exhibit slight increases in CCT and TFD, reflecting a trade-off between accuracy and computational efficiency. Communication latency is a critical challenge in distributed IoT environments, where network bandwidth and device processing power vary. Higher TFD values indicate that federated communication overhead could be a constraint for applications for real-time security requiring minimal latency for rapid threat detection. Despite these differences, FL maintains federated accuracy close to local accuracy across all datasets, ensuring that PP techniques effectively secure model training without compromising performance. These findings underscore the importance of considering dataset properties when deploying FL in IoT security settings. Datasets with higher complexity or real-time constraints may require optimized model architectures, enhanced communication strategies, and efficient resource allocation. When integrated with Autoencoder + DNN and PP methods, FL offers a scalable, privacy-preserving solution for IoT security, balancing performance, data confidentiality, and computational efficiency, making it ideal for use in industrial IoT and smart home applications., and large-scale connected systems.

5.2 Discussion

Our results show that combining an Autoencoder with a (DNN) is highly effective for detecting cyber threats in IoT environments. By integrating (FL), we achieve strong detection performance and ensure data privacy—a critical factor in real-world applications. The model performs exceptionally well in identifying large-scale attacks such as DDoS, DoS, and malware, delivering near-perfect accuracy on datasets like IoT-23, BoT-IoT, and RT-IoT 2022. However, we notice a drop in performance for stealthier threats like MITM, Spoofing, and Command & Control (C&C) attacks, where recall scores are lower. This suggests that while the model is powerful, it struggles with subtle, evasive attacks. Another key takeaway is how the model's performance varies across different datasets. It works best with well-structured datasets like BoT-IoT and IoT-23 but faces challenges in more complex and imbalanced datasets such as CIC IoT 2023 and CIC IoMT 2024. This highlights the importance of dataset characteristics in cybersecurity research.

Additionally, while FL enhances security by decentralizing data, it introduces some computational overhead. We observe slight increases in processing times, particularly in Client Computation Time (CCT) and Time to Receive Federated Data (TFD). However, despite these challenges, FL remains a strong option for securing IoT devices without compromising privacy. Overall, our findings emphasize the need for continued refinement, especially in improving recall rates for stealthy attacks. Future improvements include refining feature selection, fine-tuning model parameters, and incorporating adversarial learning techniques. Additionally, optimizing FL's communication and computational efficiency will ensure real-world usability in large-scale IoT networks.

5.3 CONCLUSION

Our study demonstrates that combining Autoencoder + DNN with Federated Learning was the best approach for securing IoT environments. The model delivers outstanding accuracy for most attack types, particularly those with distinct, high-volume patterns. However, stealthier attacks remain challenging, as seen in the lower recall rates for MITM, Spoofing, and C&C threats. This suggests that while the approach is highly effective, there is still room for improvement in detecting more sophisticated cyber threats. Federated Learning is a valuable IoT security strategy, offering strong detection performance while preserving privacy. However, some practical challenges, such as increased computation and communication overhead, need to be addressed to make it more efficient for realtime applications. Future work should focus on improving detection for stealthy attacks, optimizing FL's communication efficiency, and exploring more advanced privacy-preserving techniques. Despite its limitations, this approach lays a solid foundation for enhancing IoT security and offers a promising direction for further research and development.

Conflicts of Interest (Mandatory)

Declare conflicts of interest or state “The authors declare no conflict of interest.” Authors must identify and declare any personal circumstances or interest that may be perceived as inappropriately influencing the representation or interpretation of reported research results.

Author Contributions (Mandatory)

For research articles with several authors, a short paragraph specifying their individual contributions must be provided. The following statements should be used as follows: “Conceptualization, Mr. Mahantesh Laddi and Dr. Shridhar Allagi; methodology, Mr. Mahantesh Laddi; software, Dr. Shridhar Allagi; validation, Prakash Sonwalkar, YYY, and ZZZ; formal analysis, Mr. Mahantesh Laddi; investigation, Mr. Mahantesh Laddi; resources, Prakash Sonwalkar; data curation, Mr. Mahantesh Laddi; writing—original draft preparation, Mrs. Basavarajeshwari Nadagoudra; writing—review and editing, Shrikant Athanikar, Mrs. Basavarjeshwari Nadagoudra; visualization, Shrikant Athanikar; supervision, Dr. Shridhar Allagi; project administration, Dr. Shridhar Allagi; funding acquisition, Dr. Shridhar Allagi”, etc. Authorship must be limited to those who have contributed substantially to the work reported.

REFERENCES

[1] A. Granato and A. Polacek, “The growth and challenges of cyber insurance,” *Chicago Fed Lett.*, 426, 2019.
[2] M. A. Jabbar and R. Aluvalu, “Intrusion detection system for the internet of things: A review,” *IET Conf. Publ.*, vol. 2018, no. CP747, 2018.
[3] A. Khraisat* and P. V. and J. K., Iqbal Gondal, "Survey of intrusion detection systems: techniques, datasets and challenges,” *Cybersecurity*, vol. 2, no. 20, 2019.
[4] J. M. Kizza, “Internet of Things (IoT): Growth, Challenges, and Security BT - Guide to Computer Network Security,” J. M. Kizza, Ed. Cham: Springer International Publishing, 2024, pp. 557–573.
[5] V. Mayoral-Vilches, U. A. Carbajo, and E. Gil-Uriarte, “Industrial robot ransomware: Akerbeltz,” *Proc. - 4th IEEE Int. Conf. Robot. Comput. IRC 2020*, pp. 432–435, 2020.
[6] H. Pei Breivold, “Towards factories of the future: migration of industrial legacy automation systems in the cloud computing and Internet-of-things context,” *Enterp. Inf. Syst.*, vol. 14, no. 4, pp. 542–562, 2019.
[7] L. Xiao, X. Wan, X. Lu, Y. Zhang, and D. Wu, “IoT Security Techniques Based on Machine Learning,” pp. 1–20, 2018.
[8] B. A. y A. H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, “Communication-Efficient

Learning of Deep Networks from Decentralized Data," *Proc. 20th Int. Conf. Artif. Intell. Stat. 2017, Fort Lauderdale, Florida, USA*, vol. 54, p. 10, 2017.

[9] A. Imteaj, U. Thakker, S. Wang, J. Li, and M. H. Amini, "A Survey on Federated Learning for Resource-Constrained IoT Devices," *IEEE Internet Things J.*, vol. 9, no. 1, pp. 1–24, 2022.

[10] D. C. Attota, V. Mothukuri, R. M. Parizi, and S. Pouriyeh, "An Ensemble Multi-View Federated Learning Intrusion Detection for IoT," *IEEE Access*, vol. 9, pp. 117734–117745, 2021.

[11] A. Tabassum, A. Erbad, W. Lebda, A. Mohamed, and M. Guizani, "FEDGAN-IDS: Privacy-preserving IDS using GAN and Federated Learning," *Comput. Commun.*, vol. 192, pp. 299–310, 2022.

[12] M. Driss, I. Almomani, Z. e Huma, and J. Ahmad, "A federated learning framework for cyberattack detection in vehicular sensor networks," *Complex Intell. Syst.*, vol. 8, no. 5, pp. 4221–4235, 2022.

[13] Z. Du, C. Wu, T. Yoshinaga, K.-L. A. Yau, Y. Ji, and J. Li, "Federated Learning for Vehicular Internet of Things: Recent Advances and Open Issues," *IEEE Open J. Comput. Soc.*, vol. 1, pp. 45–61, 2020.

[14] A. Ghourabi, "A Security Model Based on LightGBM and Transformer to Protect Healthcare Systems From Cyberattacks," *IEEE Access*, vol. 10, pp. 48890–48903, 2022.

[15] I. Ioannou, P. Nagaradjane, P. Angin, P. Balasubramanian, K. J. Kavitha, P. Murugan, and V. Vassiliou, "GEMLIDS-MIOT: A Green Effective Machine Learning Intrusion Detection System based on Federated Learning for Medical IoT network security hardening," *Comput. Commun.*, vol. 218, pp. 209–239, 2024.

[16] Z. Tang, H. Hu, and C. Xu, "A federated learning method for network intrusion detection," *Concurr. Comput. Pract. Exp.*, vol. 34, no. 10, p. e6812, May 2022.

[17] H. Chen, S. Huang, D. Zhang, M. Xiao, M. Skoglund, and H. V Poor, "Federated Learning Over Wireless IoT Networks With Optimized Communication and Resources," *IEEE Internet Things J.*, vol. 9, no. 17, pp. 16592–16605, 2022.

[18] M. M. Rashid, S. U. Khan, F. Eusufzai, M. A. Redwan, S. R. Sabuj, and M. Elsharief, "A Federated Learning-Based Approach for Improving Intrusion Detection in Industrial Internet of Things Networks," *Network*, vol. 3, no. 1, pp. 158–179, 2023.

[19] V. Ravi, R. Chaganti, and M. Alazab, "Recurrent deep learning-based feature fusion ensemble meta-classifier approach for intelligent network intrusion detection system," *Comput. Electr. Eng.*, vol. 102, p. 108156, 2022.

[20] G. de Carvalho Bertoli, L. Alves Pereira Junior, O. Saotome, and A. L. dos Santos, "Generalizing intrusion detection for heterogeneous networks: A stacked-unsupervised federated learning approach," *Comput. Secur.*, vol. 127, p. 103106, 2023.

[21] M. Karthikeyan, D. Manimegalai, and K. RajaGopal, "Firefly algorithm based WSN-IoT security enhancement with machine learning for intrusion detection," *Sci. Rep.*, vol. 14, no. 1, 2024.

[22] S. M. S. Bukhari, M. H. Zafar, M. A. Houran, S. K. R. Moosavi, M. Mansoor, M. Muaaz, and F. Sanfilippo, "Secure and privacy-preserving intrusion detection in wireless sensor networks: Federated learning with SCNN-Bi-LSTM for enhanced reliability," *Ad Hoc Networks*, vol. 155, no. January, 2024.

[23] W. Xu, J. Jang-Jaccard, A. Singh, Y. Wei, and F. Sabrina, "Improving Performance of Autoencoder-Based Network Anomaly Detection on NSL-KDD Dataset," *IEEE Access*, vol. 9, pp. 140136–140146, 2021.

[24] B. Min, J. Yoo, S. Kim, D. Shin, and D. Shin, "Network Anomaly Detection Using Memory-Augmented Deep Autoencoder," *IEEE Access*, vol. 9, pp. 104695–104706, 2021.

[25] L. Mhamdi, D. McLernon, F. El-moussa, S. A. R. Zaidi, M. Ghogho, and T. Tang, "A Deep Learning Approach Combining Autoencoder with One-class SVM for DDoS Attack Detection in SDNs," in *2020 IEEE Eighth International Conference on Communications and Networking (ComNet)*, 2020, pp. 1–6.

[26] 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 Comput.*, vol. 17, no. 3, pp. 12–22, 2018.

[27] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A Fast Learning Algorithm for Deep Belief Nets," *Neural Comput.*, vol. 18, pp. 1527–1554, 2006.

[28] D. Yu and L. Deng, "Deep Learning and Its Applications to Signal and Information Processing [Exploratory DSP]," *IEEE Signal Process. Mag.*, vol. 28, no. 1, pp. 145–154, 2011.