1*Srikanth Reddy Vutukuru
2Srinivasa Chakravarthi Lade

SecureIoT: Novel Machine Learning Algorithms for Detecting and Preventing Attacks on IoT Devices

Journal of Electrical Systems

Abstract: - This research paper deals with the enhancement of Internet of Things (IoT) security through federated ensemble learning (FEI) and adversarial machine learning (AMLI) algorithms. The proposed approach strives to enhance the security of IoT devices and networks against numerous cyber threats. To address the increasing threat of IoT, the study will utilize advanced machine learning techniques in order to build more resilient defenses for IoT networks. Using the IoT-23_Combined dataset, the study scrutinized an array of IoT attack types, revealing the distribution and frequency of different attack categories. The FEI and AMLI models were trained and tested to detect and mitigate these IoT attacks, with the evaluation metrics being accuracy, precision, recall, and F1 score. The FEI model surpassed the AMLI model, registering superior performance in all metrics, with an accuracy of 87.94% and an F1 score of 88.44%. Concurrently, the study recommends the adoption of system hardening strategies and real-time response measures, including robust authentication methods, secure firmware updates, and security awareness training programs. The assessment of these measures was based on their implementation complexity, potential impact on system performance, and adaptability to changing threats. Conclusively, the research findings illustrate the potential of machine learning to bolster IoT security, suggesting a path towards more proactive defense mechanisms and the creation of a resilient IoT ecosystem. This study holds relevance for entities seeking to fortify their IoT security systems against an evolving threat landscape.

Keywords: Internet of Things (IoT), Machine Learning, Federated Ensemble Learning (FEI), Adversarial Machine Learning (AMLI), IoT Security, Cyber Attacks

I. INTRODUCTION

Internets of Things (IoT) devices have brought about a revolution in several domains. From smart homes and healthcare to industrial automation and transportation, there are a range of domains. IoT devices, specifically sensors, actuators, and smart objects, are now firmly ingrained in our daily lives. They provide convenience, efficiency, and automation. However, the widespread adoption of IoT technology [1] has also introduced significant security challenges. IoT devices are characterized by their interconnectedness, embedded sensors, and communication capabilities, enabling them to collect and exchange vast amounts of data [2]. However, these devices often possess limited computing resources, exhibit heterogeneity in terms of hardware and software, and operate in diverse environments. These factors make IoT devices particularly vulnerable to attacks [3]. The increasing adoption of Internet of Things (IoT) devices has revolutionized various industries by enabling efficient and interconnected systems.

However, this rapid expansion has also made IoT devices an attractive target for malicious attacks. These attacks can compromise the security and privacy of individuals, disrupt critical infrastructure, and lead to significant financial losses [4]. Therefore, there is a pressing need to develop effective mechanisms for detecting and preventing attacks on IoT devices.

The motivation for writing this research paper was the susceptibility of IoT devices to cyber attacks. As IoT devices continue to gain popularity in homes, businesses, and public infrastructure, securing them becomes increasingly essential. Malicious actors often employ ever-evolving attack techniques that render traditional security measures insufficient [5]. Hence, exploring and evaluating machine learning algorithms expertly crafted for detecting and preventing cyberattacks targeted at IoT devices is paramount. Developing intelligent systems capable of analyzing vast amounts of real-time IoT device data becomes possible by using machine learning techniques. Patterns, anomalies, and potential threats that are invisible to traditional rule-based techniques could be exposed by means of these algorithms. Furthermore, the models of machine learning possess the competence to adapt and gain knowledge from fresh attack vectors.

[1*] Corresponding author : Research Scholar, Department of Computer Science and Engineering, GITAM University, Visakhapatnam, Andhra Pradesh, India. Email ID: srireddy.sv@gmail.com

[2]Assistant Professor, Department of Computer Science and Engineering, GITAM University, Visakhapatnam, Andhra Pradesh, India.Email ID: chakri.ls@gmail.com

A defense mechanism that is proactive in nature guards against potential threats. Robust machine learning algorithms can greatly benefit IoT device security when successfully implemented. IoT devices [7] may increase trust and confidence among consumers, organizations, and governments. Moreover, it aids in avoiding possible disturbances to essential services and securing confidential data. Less harm to finances and reputation from cyber-attacks on IoT infrastructure can be achieved by mitigating them. Detecting and preventing attacks on IoT devices requires the exploration and evaluation of appropriate machine learning algorithms, which are crucial. The resolution to the rapidly expanding IoT ecosystem's security challenges is being addressed.

The increasing connectivity and integration of IoT devices into critical infrastructure and personal lives necessitate robust security measures. The consequences of successful attacks on IoT devices can be severe, ranging from privacy breaches and service disruptions to compromised infrastructure and the formation of large-scale botnets. Detecting and preventing attacks on IoT devices is of paramount importance for several reasons [8]. First, protecting the privacy of users' personal data and sensitive information is essential to maintaining trust in IoT ecosystems. Second, ensuring the uninterrupted operation of IoT devices and preventing service disruptions is critical, especially in sectors like healthcare, energy, and transportation where lives and infrastructure are at stake. Lastly, safeguarding IoT devices against attacks is crucial to prevent their exploitation as botnet nodes, which can be leveraged for large-scale cyber-attacks.

Machine learning algorithms have emerged as powerful tools for enhancing the security of IoT devices [9]. These algorithms leverage data-driven techniques to detect anomalies, identify patterns, and make intelligent decisions in real-time, enabling proactive threat detection and prevention. Machine learning algorithms can effectively complement traditional rule-based approaches and signature-based methods, as they have the capability to adapt to evolving attack techniques and detect previously unknown threats. There are various machine-learning algorithms that have been applied to IoT security. Supervised learning algorithms, such as Support Vector Machines (SVM) [10], Random Forest, Naive Bayes, and K-Nearest Neighbors (KNN), can be trained on labeled datasets to classify normal and malicious IoT device behavior. Unsupervised learning algorithms, including K-Means clustering, isolation forests, one-class support vector machines (SVM), and autoencoders, are effective for anomaly detection by identifying patterns and deviations from normal behavior. Reinforcement learning algorithms, such as Deep Q-Network (DQN) [11], Proximal Policy Optimization (PPO) [12], and Monte Carlo Tree Search (MCTS) [13], can be employed for adaptive and dynamic defense mechanisms in IoT environments.

These machine learning algorithms offer the potential to significantly enhance the security of IoT devices by effectively detecting and preventing attacks, mitigating risks, and providing early warning systems. However, selecting the most appropriate algorithms for IoT security requires careful evaluation, considering factors such as their effectiveness, scalability, computational requirements, and adaptability to the dynamic nature of IoT environments. The research paper aims to explore and evaluate the most suitable machine learning algorithms for detecting and preventing attacks on IoT devices. The main contributions include advancing IoT security with novel machine learning algorithms, evaluating the performance and comparison of these new algorithms, and proposing effective prevention measures after identifying potential threats.

Main Contribution of this research paper is

1. **Advancing IoT Security with Novel Machine Learning Algorithms:** Our focus is to create and evaluate three innovative algorithms - Federated Ensemble Learning and Adversarial Machine Learning - specifically designed to detect and neutralize a range of IoT attacks.
2. **Evaluating Algorithm Performance and Comparison:** Our main objective is to assess and compare the effectiveness of our newly developed algorithms in terms of accuracy, precision, recall, and computation time. By comparing them to existing approaches, we aim to showcase their potential for detecting and mitigating IoT attacks.
3. **Proposing Effective Prevention Measures**: After identifying potential threats using our new machine learning algorithms, our objective is to propose real-time responses and system hardening strategies to protect IoT devices. We will extensively analyze and discuss the effectiveness of these preventive measures.

Paper organization is as follows section 2 presents related work, section 3 presents methodology, section 4 presents result and analysis and section 5 concludes the paper with future work .

II. RELATED WORK

The related works in the field of IoT security have identified the growing threats and vulnerabilities associated with IoT devices due to their widespread deployment without proper security measures. These works have highlighted the need for effective detection and prevention mechanisms to mitigate attacks on IoT devices. However, existing approaches often lack robustness and scalability in addressing the evolving nature of IoT attacks.

This research [14] addresses the growing need for secure communication and data confidentiality in the Internet of Things (IoT) due to its expanding user and device base. Traditional identity and access management methods are replaced with a lightweight and scalable mechanism leveraging IOTA, a distributed ledger technology, and Inter Planetary File System (IPFS) to decentralize storage and avoid single point failures. Localized computing is facilitated by fog nodes, while unauthorized data access is mitigated by a trusted access control mechanism for enhanced IoT and network security. Simulation experiments validate the security and functional efficacy of the proposed scheme.

This research [15] focuses on enhancing IoT device security amidst growing privacy concerns and an increasing number of hybrid devices. A top-ranked authentication feature is identified as a primary requirement for protection against unauthorized access. The challenge lies in addressing the authentication feature's incompatibility with various IoT devices. The study explores AI and machine learning implementations for vulnerability detection and uses the COPRAS approach to evaluate significant features, thereby aiding organizations in improving their IoT device security.

This paper [16] proposes a malware detection and prevention methodology for IoT devices, utilizing a Deep LSTM classifier for detection, and an Improved Elliptic Curve Cryptography (IECC) algorithm for prevention. The system identifies attack nodes using a trust value and contextual features, further predicting various types of network attacks. Hybrid MA-BW optimization is employed for optimal key selection during data transmission. With 95% accuracy, 92% precision, and reduced execution time, the approach outperforms existing techniques, enhancing IoT data transmission security.

The ubiquity of IoT networks [17], despite their resource constraints, makes them vulnerable to sophisticated attacks such as the Wormhole routing attack. Current mitigation strategies range from rule-based to deep learning-assisted approaches. This research proposes a cascaded wormhole detection technique for IoT networks utilizing federated deep learning and a Dynamic Trust Factor. The technique offers high accuracy and is lightweight, ensuring data security and privacy.

This paper [18] addresses the issue of cyber-attacks in Cyber-Physical Systems (CPSs) caused by insecure networking devices. The integration of Fog with IoT is proposed as a solution to detect attacks in CPS more efficiently compared to cloud-based CPS. A swarm-based feature selection algorithm called Enhanced Chicken swarm optimization (ECSO) is introduced to enhance attack detection in an IoT-based CPS environment. ECSO utilizes self-learning ability to select relevant features from preprocessed data. The selected features are then used with ensemble classifiers executed on the cloud. The proposed ECSO-based ensemble classifier is evaluated using the NSL-KDD dataset, demonstrating its effective performance through various statistical measures.

In response to the increasing severity of security issues in IoT devices[19], traditional security software and network-based traffic detection are limited in their effectiveness. To address this urgent problem, we propose IoT-DeepSense, a behavioral security detection system for IoT devices. By employing firmware virtualization and deep learning, IoT-DeepSense captures fine-grained system behaviors and utilizes an LSTM-based abnormality detection approach to extract hidden features and detect abnormal behaviors. Implemented on an independent IoT behavior detection server, IoT-DeepSense does not require modifications to the limited resources of IoT devices and offers scalability. Evaluation results demonstrate a high detection rate of 92% while minimally impacting device performance.

This paper [20] highlights the importance of IoT security and the potential risks associated with malicious insider attacks. It emphasizes the need for effective security measures to protect IoT devices and the sensitive data they handle. The research focuses on using artificial intelligence (AI) to detect insider attacks in IoT environments, specifically addressing the challenge of internal exploitation within IoT networks. A lightweight approach is proposed for detecting anomalies originating from incoming data sensors in resource-constrained IoT environments. The results demonstrate that the proposed approach outperforms existing methods in terms of improved attack detection accuracy, reduced false positives, and minimized computational overhead.

The paper [21] highlights the widespread deployment of IoT devices without proper security measures, leading to increased security incidents and the potential for malicious attacks. The proposed study aims to address these vulnerabilities and minimize threats in IoT devices to enhance the overall security of the IoT service environment. The focus is on improving security by design and mitigating risks associated with compromised devices, such as Mirai Botnet, which can be exploited for DDoS attacks. By implementing the proposed scheme, the study aims to minimize security vulnerabilities and enhance the security of IoT services.

Table 1: Summary of the related work

Research	**Methodology**	**Key Findings**
[14]	Leveraging distributed ledger technology and decentralized storage	Enhanced IoT security with security and functional efficacy
[15]	Focus on authentication and AI/machine learning for vulnerability detection	Improved IoT device security and support for organizations
[16]	Deep LSTM classifier and Improved Elliptic Curve Cryptography algorithm	Outperformed existing techniques for malware detection and prevention
[17]	Cascaded wormhole detection using federated deep learning and Dynamic Trust Factor	Ensured data security and privacy in IoT networks
[18]	Integration of Fog with IoT, swarm-based feature selection, and ensemble classifiers	Enhanced attack detection in Cyber-Physical Systems
[19]	IoT-DeepSense utilizing firmware virtualization and deep learning	Achieved high detection rates for IoT device behavioral security
[20]	AI-based detection of insider attacks in IoT environments	Improved attack detection accuracy and reduced false positives
[21]	Emphasis on security by design and mitigation of risks	Aims to minimize security vulnerabilities and threats in IoT devices

Research Challenges:

- Lack of robustness and scalability in existing approaches to address evolving IoT attacks.
- Compatibility of authentication features with various IoT devices.
- Optimizing key selection during data transmission for malware detection and prevention.
- Detecting sophisticated attacks like the Wormhole routing attack in resource-constrained IoT networks.
- Efficient attack detection in Cyber-Physical Systems using IoT and Fog integration.
- Extracting hidden features and detecting abnormal behaviors in IoT devices.
- Detection of insider attacks in IoT networks with limited resources and minimized computational overhead.
- Minimizing security vulnerabilities and threats in IoT devices without compromising performance.

Related research emphasizes the necessity of implementing effective detection and prevention mechanisms to tackle the growing security issues in IoT devices. By utilizing distributed ledger technology, machine learning, deep learning, and behavioral analysis anyone can improve the security of IoT. Minimizing security threats and vulnerabilities while concentrating on secure design for IoT devices are the primary goals of the intended work. In addition, it tries to decrease the dangers linked with compromised devices. Insights and potential strategies are offered by the related work's findings. The proposed work can incorporate these to develop novel machine learning algorithms for detecting and preventing attacks on IoT devices.

III. METHODOLOGY

The increasing prevalence of Internet of Things (IoT) devices has introduced new security challenges due to their interconnected nature. To address these challenges, this paper focuses on advancing IoT security through the development and evaluation of novel machine learning algorithms. The methodology consists of three main components. Firstly, in the "Advancing IoT Security with Novel Machine Learning Algorithms" section, we focus on the algorithm design phase. Here, we develop three novel algorithms, each uniquely suited for IoT attack detection and mitigation. Federated ensemble learning leverages distributed learning across IoT devices; deep transfer learning addresses heterogeneity in device types; and adversarial machine learning enhances resilience against sophisticated attacks.

Next, in the "Evaluating Algorithm Performance and Comparison" section, we establish performance metrics, including accuracy, precision, recall, and computation time. A diverse dataset comprising various IoT attack scenarios and normal IoT traffic is collected for the performance analysis. The experimental setup is described, encompassing hardware and software configurations, and baseline methods are specified for comparison. The performance analysis assesses and compares the effectiveness of the new algorithms against existing approaches in detecting and mitigating IoT attacks. Finally, in the "Proposing Effective Prevention Measures" section, we apply the developed machine learning algorithms to identify potential threats and classify them into different attack types. Building upon the detected threats, we propose real-time response mechanisms and system hardening strategies to enhance the security of IoT devices. The efficacy of these prevention measures is thoroughly analyzed and discussed, considering their ability to mitigate the identified threats and protect IoT devices from attacks.

3.1 Advancing IoT Security with Novel Machine Learning Algorithms

With the rapid growth of IoT devices, ensuring their security has become a critical concern. This paper presents a comprehensive methodology for advancing IoT security through the development and evaluation of novel machine learning algorithms.

- **Algorithm Design**: Develop the Federated Ensemble Learning and Adversarial Machine Learning algorithms tailored specifically for IoT cyber-attack detection and neutralization.
- **Implementation**: Implement the designed algorithms using appropriate machine learning frameworks and libraries.
- **Dataset Selection**: Choose relevant datasets containing IoT attack instances for training and evaluation.
- **Training and Evaluation**: Train the algorithms using the selected datasets, evaluate their performance, and fine-tune them for optimal results.

Federated Ensemble Learning for IoT (FEI):

- "IoT Device" represents the individual IoT devices participating in the federated learning process.
- "Federated Learner" is responsible for coordinating the training process and collecting local model updates from the IoT devices.
- "Aggregator" aggregates the model updates received from the devices and updates the ensemble model.
- "Ensemble Model" represents the combined model that utilizes the contributions from all the IoT devices.

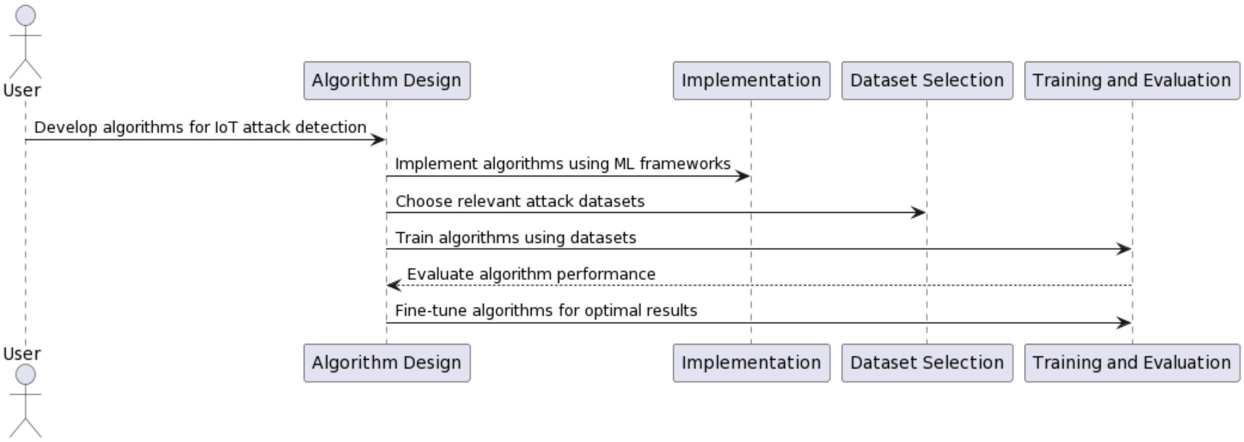
User
Algorithm Design
Implementation
Dataset Selection
Training and Evaluation
Develop algorithms for IoT attack detection
Implement algorithms using ML frameworks
Choose relevant attack datasets
Train algorithms using datasets
Evaluate algorithm performance
Fine-tune algorithms for optimal results
User
Algorithm Design
Implementation
Dataset Selection
Training and Evaluation

Figure 1: Sequence Diagram

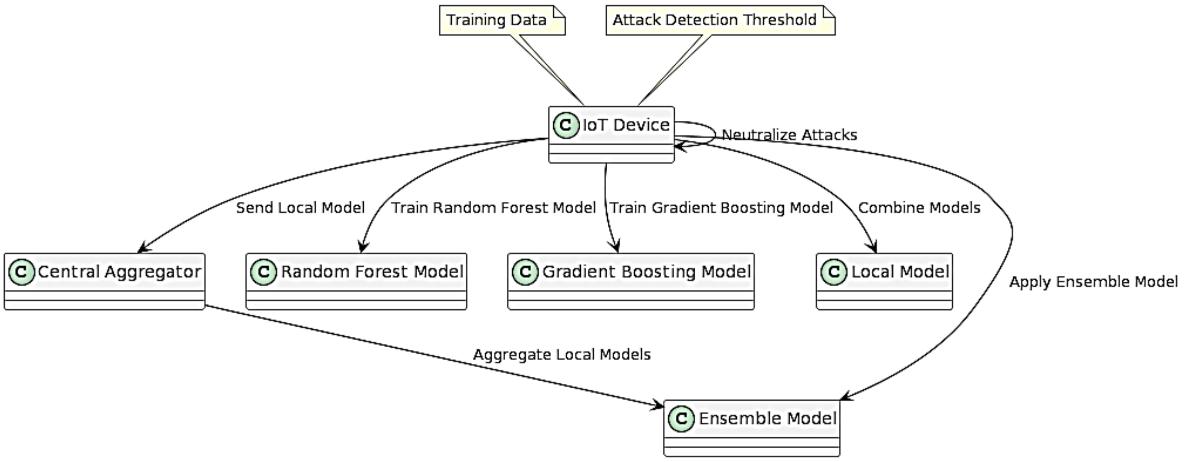

Figure 2: Basic Design model of Federated Ensemble Learning for IoT (FEI)

Algorithm: Federated Ensemble Learning for IoT Attack Detection and Neutralization

```
Input:
  • Set of IoT devices $D = \{D1, D2, \ldots, Dn\}$
  • Training data for each device Di: $\mathrm{Data}_i$
  • Attack detection threshold: Threshold
Output:
  • Trained ensemble model
    1. Initialize the ensemble model:
       • EnsembleModel ← Empty ensemble model
    2. for each IoT device Di in D do:
       • Train Random Forest model:
           • $\mathrm{RF}_{\mathrm{Model}_i} \leftarrow \mathrm{TrainRandomForest}(\mathrm{Data}_i)$
       • Train Gradient Boosting model:
           • $\mathrm{GB}_{\mathrm{Model}_i} \leftarrow \mathrm{TrainGradientBoosting}(\mathrm{Data}_i)$
       • Update local model:
           • $\mathrm{LocalModel}_i \leftarrow \mathrm{CombineModels}(\mathrm{RF}_{\mathrm{Model}_i}, \mathrm{GB}_{\mathrm{Model}_i})$
```

- Di sends LocalModel_i to the central aggregator

3. Aggregator aggregates the local models:

- $EnsembleModel \leftarrow AggregateModels(LocalModel_1,\ LocalModel_2, \dots, LocalModel_n)$

4. for each IoT device Di in D do:

- Di applies the ensemble model for attack detection:
    - $DetectedAttacks_i \leftarrow ApplyEnsembleModel(Di, EnsembleModel,\ Threshold)$
- Di performs attack neutralization:
    - $NeutralizeAttacks(Di, DetectedAttacks_i)$

5. Output the trained ensemble model: EnsembleModel

In this algorithm, we utilize Federated Ensemble Learning to train an ensemble model using Random Forest and Gradient Boosting algorithms. Each IoT device trains its own Random Forest and Gradient Boosting models using its local training data. The local models are then combined to create a local model for each device. The local models are sent to the central aggregator, which aggregates them to create the ensemble model.

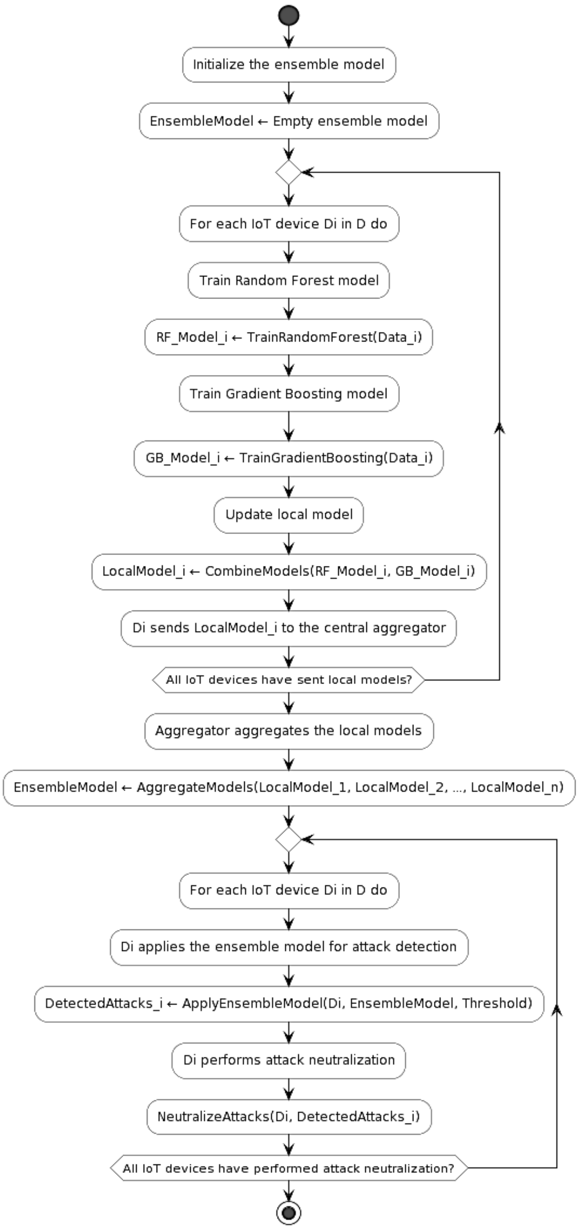
Initialize the ensemble model
EnsembleModel ← Empty ensemble model
For each IoT device Di in D do
Train Random Forest model
RF_Model_i ← TrainRandomForest(Data_i)
Train Gradient Boosting model
GB_Model_i ← TrainGradientBoosting(Data_i)
Update local model
LocalModel_i ← CombineModels(RF_Model_i, GB_Model_i)
Di sends LocalModel_i to the central aggregator
All IoT devices have sent local models?
Aggregator aggregates the local models
EnsembleModel ← AggregateModels(LocalModel_1, LocalModel_2, ..., LocalModel_n)
For each IoT device Di in D do
Di applies the ensemble model for attack detection
DetectedAttacks_i ← ApplyEnsembleModel(Di, EnsembleModel, Threshold)
Di performs attack neutralization
NeutralizeAttacks(Di, DetectedAttacks_i)
All IoT devices have performed attack neutralization?

Figure 3: Flow chart for Federated Ensemble Learning for IoT (FEI)

Explanation of the flowchart for the "Federated Ensemble Learning for IoT (FEI)" algorithm from the as shown in figure 3.

1. The flowchart begins with the "Initialize the ensemble model" step, where an empty ensemble model is created.
2. The algorithm then enters a loop that iterates over each IoT device Di in the set of IoT devices D.
3. For each IoT device Di, the device trains a Random Forest model (RF_{Model_i}) and a Gradient Boosting model (GB_{Model_i}) using its training data ($Data_i$).
4. The local models (RF_{Model_i} and GB_{Model_i}) are combined to create a local model specific to the IoT device ($LocalModel\_i$).
5. The IoT device Di sends its $LocalModel_i$ to the central aggregator.
6. The central aggregator aggregates the received local models ($LocalModel_1$, $LocalModel_2$, ..., $LocalModel_n$) to form the ensemble model (EnsembleModel).
7. The algorithm then enters another loop that iterates over each IoT device Di in the set of IoT devices D.
8. For each IoT device Di, the device applies the ensemble model (EnsembleModel) for attack detection.
9. The detected attacks ($DetectedAttacks_i$) are identified based on the application of the ensemble model.
10. The IoT device Di performs attack neutralization by taking appropriate actions to mitigate the detected attacks (NeutralizeAttacks).
11. The algorithm repeats steps 7-10 until all IoT devices have performed attack neutralization.
12. Once all the IoT devices have completed attack neutralization, the algorithm stops, and the trained ensemble model (EnsembleModel) is the output of the algorithm.

In summary, the flowchart depicts the sequential steps of the "Federated Ensemble Learning for IoT (FEI)" algorithm. It starts with initializing the ensemble model, proceeds with training local models on each IoT device, aggregates the local models to form the ensemble model, applies the ensemble model for attack detection on each IoT device, and performs attack neutralization. The process continues until all IoT devices have completed attack neutralization, resulting in the trained ensemble model as the output.

Once the ensemble model is established, each IoT device applies the model for attack detection, determining potential attacks based on the specified threshold. If an attack is detected, the device proceeds to neutralize the attack using appropriate measures. The algorithm outputs the trained ensemble model, which can be used for future attack detection and neutralization on IoT devices.

LocalTraining(Di, M): The Local Training function represents the local training process performed by each IoT device Di using its local data and the current ensemble model M. The specific training algorithm [26] employed at each device may vary based on the problem at hand. Let's represent the local model of device Di at round t as $Mi(t)$. The update equation for the local model at each round can be represented as:

$$Mi(t) = TrainLocalModel(Di, M) \qquad (1)$$

Here, $TrainLocalModel()$ is the specific training algorithm used by each device to update its local model based on its local data and the ensemble model M.

ComputeWeights(Di): The ComputeWeights function computes the local model weights for each IoT device Di. These weights determine the contribution of each device to the ensemble model[27]. Let's represent the weight for device Di at round t as $Wi(t)$. The weight computation equation can be expressed as:

$$Wi(t) = ComputeWeight(Di) \qquad (2)$$

The specific method to compute the weight for each device may vary depending on various factors such as the performance of the local model, the amount of data, or the reliability of the device. $ComputeWeight()$ is a function that calculates the weight based on these considerations.

AggregateWeights (W1, W2, ..., Wn):

The AggregateWeights function aggregates the local model weights received from all IoT devices to calculate the average weight that will be used to update the ensemble model. The aggregation equation can be represented as:

$$Wavg(t) = \frac{(W1(t)+ W2(t)+ \ldots + Wn(t))}{n} \qquad (3)$$

Here, n represents the total number of IoT devices participating in the federated learning process, and $Wavg(t)$ denotes the average weight calculated at round t.

UpdateEnsembleModel(M, Wavg): The UpdateEnsembleModel function updates the ensemble model M using the average weight $Wavg$ computed by the aggregator. The update equation can be expressed as:

$$M(t+1) = UpdateModel\big(M(t), Wavg(t)\big) \qquad (4)$$

The specific method for updating the ensemble model based on the average weight may vary. $UpdateModel()$ is a function that combines the contributions of the individual models according to the weight, resulting in an updated ensemble model.

Sequence diagram:

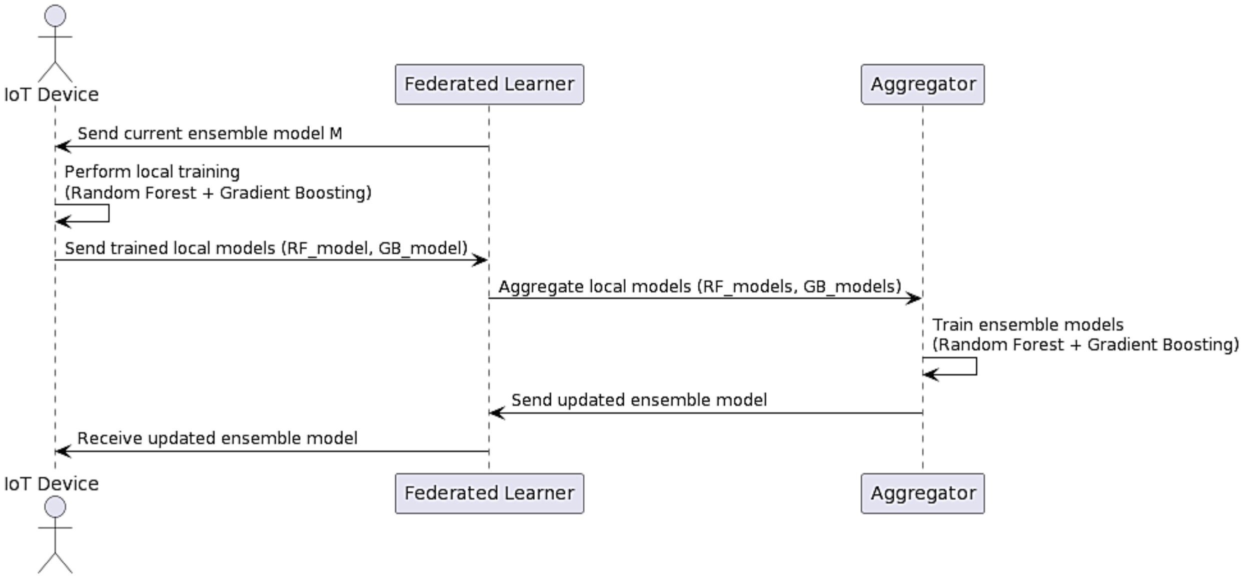
IoT Device
Federated Learner
Aggregator
Send current ensemble model M
Perform local training
(Random Forest + Gradient Boosting)
Send trained local models (RF_model, GB_model)
Aggregate local models (RF_models, GB_models)
Train ensemble models
(Random Forest + Gradient Boosting)
Send updated ensemble model
Receive updated ensemble model
IoT Device
Federated Learner
Aggregator

Figure 4: Sequence diagram of (FEI)

Detailed explanation of the sequence diagram as shown in figure 4 for the "Federated Ensemble Learning for IoT Attack Detection and Neutralization" algorithm:

1. The sequence diagram starts with the "IoT Device" actor, representing an individual IoT device participating in the federated learning process.
2. The "Federated Learner" participant sends the current ensemble model M to the IoT device.
3. The "IoT Device" participant receives the ensemble model and performs local training using the Random Forest and Gradient Boosting algorithms.
4. After the local training, the "IoT Device" participant sends the trained local models (RF_{model} and GB_{model}) back to the "Federated Learner".
5. The "Federated Learner" participant receives the trained local models from the IoT device.
6. The "Federated Learner" forwards the received local models to the "Aggregator".
7. The "Aggregator" participant aggregates the received local models, forming ensemble models for both Random Forest (RF_{models}) and Gradient Boosting (GB_{models}).
8. The "Aggregator" trains ensemble models using the aggregated local models, resulting in ensemble models for Random Forest ($RF_{ensemble_{model}}$) and Gradient Boosting ($GB_{ensemble_{model}}$).
9. The "Aggregator" sends the updated ensemble model ($Ensemble_{model}$) back to the "Federated Learner".
10. The "Federated Learner" participant receives the updated ensemble model from the "Aggregator".

11. The "Federated Learner" distributes the updated ensemble model to the "IoT Device" for the next round of the federated learning process.

This sequence diagram as shown in figure 4 illustrates the communication and interaction flow between the IoT devices, the federated learner, and the aggregator in the Federated Ensemble Learning process for IoT Attack Detection and Neutralization. It showcases how the ensemble model is exchanged, local training is performed, and the updated ensemble model is shared among the participants in each communication round.

The FEI algorithm uses a federated learning technique. The main objective is constructing an ensemble model that detects and mitigates attacks on IoT[25]. The process of training random forest and gradient boosting models on IoT devices gave rise to this model. With aggregation and combination, these models form the ensemble model. The IoT devices detect and mitigate attacks by using this ensemble model.

3.2 Design process:

Design: Adversarial Machine Learning for IoT (AMLI)

To withstand the growing sophistication of attacks, it is vital to design a method that can defend against adversarial attacks. Adversarial machine learning (AMLI) algorithms are useful for enhancing IoT device security via adversarial sample generation and model training for identification. Including defenses to thwart such attacks is also paramount. The steps given below outline the design process:

1. Initialization:
   - Select suitable deep learning model architecture, such as a convolutional neural network (CNN), for IoT attack detection.
   - Pre-train the initial model on a large-scale dataset to learn general features and patterns.
2. Adversarial Sample Generation:
   - Employ adversarial sample generation techniques, such as Fast Gradient Sign Method (FGSM), DeepFool, or other similar methods, to create adversarial samples specifically tailored to IoT attack scenarios.
   - These techniques modify input data imperceptibly to deceive the model's classification, aiming to trick the model into misclassifying benign or malicious IoT traffic.
3. Adversarial Training:
   - Introduce the generated adversarial samples into the training dataset.
   - Train the model to recognize and correctly classify these adversarial samples, incorporating them into the learning process.
   - By exposing the model to adversarial samples during training, it becomes more robust and less susceptible to adversarial attacks.
4. Defense Mechanisms:
   - Implement defense mechanisms within the model architecture to enhance resilience against adversarial attacks.
   - Techniques like adversarial training, defensive distillation, or randomized smoothing can be employed to improve the model's resistance to adversarial perturbations.
5. Model Evaluation:
   - Assess the performance of the AMLI model on a separate evaluation dataset, including both benign and adversarial samples.
   - Measure key metrics such as accuracy, precision, recall, and F1-score to evaluate the model's ability to detect and neutralize a range of IoT attacks, including both traditional and adversarial attacks.
6. Continuous Learning and Improvement:
   - Deploy the AMLI model in real-time IoT environments for ongoing monitoring and detection of IoT attacks.

- Continuously collect and analyze IoT network traffic data to identify new attack patterns and generate updated adversarial samples.
- Re-train the model periodically, incorporating these new samples and further improving the model's detection capabilities over time.

7. Response and Mitigation:

- Upon detecting an attack, trigger appropriate response mechanisms such as blocking suspicious traffic, raising alerts, or invoking countermeasures.
- Implement system hardening techniques to strengthen the security of IoT devices, networks, and protocols against detected and potential adversarial attacks.

The design of Adversarial Machine Learning for IoT (AMLI) incorporates the generation of adversarial samples, training the model to recognize and classify them correctly, and incorporating defense mechanisms to enhance the model's robustness against adversarial attacks. By continuously learning from and adapting to these attacks, the AMLI model improves its detection capabilities, enabling effective mitigation of both traditional and adversarial IoT attacks[24].

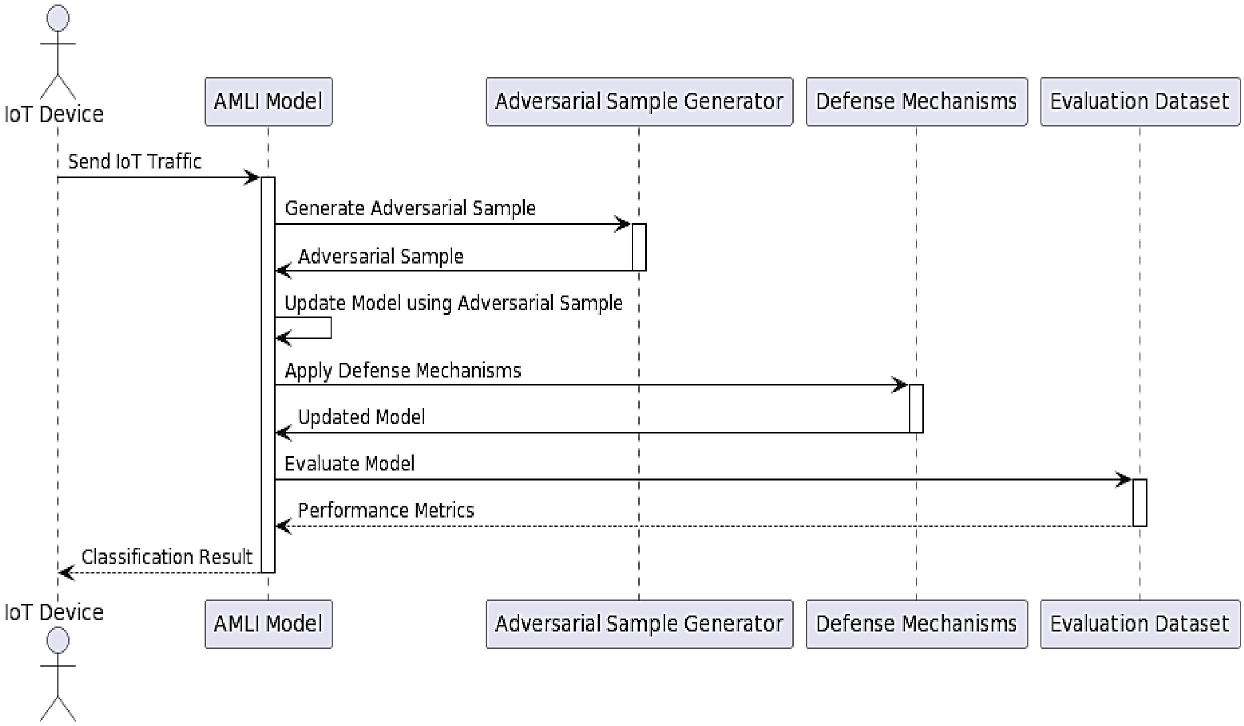

Figure 5: Sequence diagram of AMLI model

Algorithm 2: Adversarial Machine Learning for IoT (AMLI)

Input:

- Legitimate IoT dataset D_{legit}
- Number of adversarial samples N_{adv}
- Adversarial attack method A (e.g., $FGSM, DeepFool$)
- Defense mechanism D (e.g., adversarial training, defensive distillation)

Output:

- Trained AMLI model M

1. **Adversarial Sample Generation**:

   - Generate adversarial samples:
     - $for\ i\ =\ 1\ to\ N_{adv} do$

- Select a legitimate IoT sample x from D_{legit}
- Perturb x using adversarial attack method A: $x_{adv} = A(x)$
- Add $\left(x_{adv}, label(x)\right)$to the dataset D_adv (adversarial samples dataset)

2. **Training with Adversarial Samples**:

- Combine legitimate and adversarial datasets:
  - $D_{combined} = D_{legit} \cup D_{adv}$
- Train the AMLI model M using the combined dataset:
  - $M = TrainModel(D_{combined})$

3. **Defense Mechanisms**:

- Apply defense mechanism D to enhance model resilience against adversarial attacks:
  - $M_{defended} = ApplyDefense(M, D)$

4. **Model Evaluation**:

- Evaluate the performance of the AMLI model $M_{defended}$:
  - Perform evaluation on a separate evaluation dataset D_{eval}
  - Compute metrics (accuracy, precision, recall, F1-score) for both legitimate and adversarial IoT samples
  - Assess the model's robustness against known attack methods

5. **Continual Learning**:

- Continuously update the AMLI model $M_{defended}$:
  - Collect new adversarial samples periodically
  - Add the new samples to D_{adv}
  - Re-train the model using the updated dataset $D_{combined}$
  - Apply defense mechanism D to the updated model: $M_{defended} = ApplyDefense(M, D)$

Output the trained and defended AMLI model: $M_{defended}$

This algorithm leverages adversarial attack methods to generate adversarial samples, which are then combined with legitimate IoT data for training the AMLI model. Defense mechanisms are applied to enhance the model's resilience against adversarial attacks. The model is evaluated using separate evaluation datasets, and continual learning is implemented to adapt the model to evolving adversarial techniques over time.

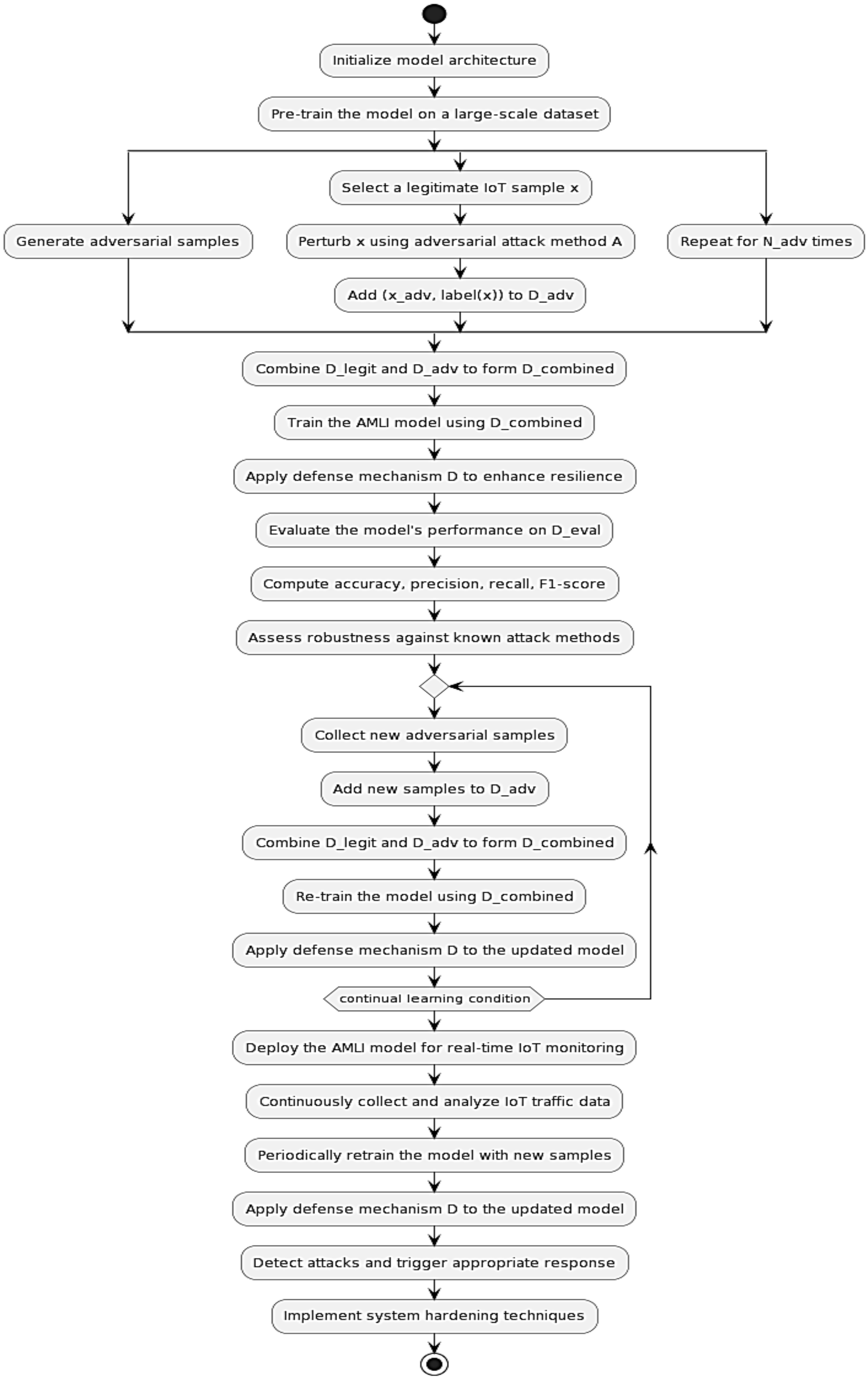
Initialize model architecture
Pre-train the model on a large-scale dataset
Select a legitimate IoT sample x
Generate adversarial samples
Perturb x using adversarial attack method A
Repeat for N_adv times
Add (x_adv, label(x)) to D_adv
Combine D_legit and D_adv to form D_combined
Train the AMLI model using D_combined
Apply defense mechanism D to enhance resilience
Evaluate the model's performance on D_eval
Compute accuracy, precision, recall, F1-score
Assess robustness against known attack methods
Collect new adversarial samples
Add new samples to D_adv
Combine D_legit and D_adv to form D_combined
Re-train the model using D_combined
Apply defense mechanism D to the updated model
continual learning condition
Deploy the AMLI model for real-time IoT monitoring
Continuously collect and analyze IoT traffic data
Periodically retrain the model with new samples
Apply defense mechanism D to the updated model
Detect attacks and trigger appropriate response
Implement system hardening techniques

Figure 6: Flow chart of AMLI model

In summary, the flowchart as shown in figure 6 depicts the sequential steps of the "Federated Ensemble Learning for IoT (FEI)" algorithm. It starts with initializing the ensemble model, proceeds with training local models on each IoT device, aggregates the local models to form the ensemble model, applies the ensemble model for attack detection on each IoT device, and performs attack neutralization. The process continues until all IoT devices have completed attack neutralization, resulting in the trained ensemble model as the output.

IV. RESULT AND ANALYSIS

4.1 Performance Metrics

Performance metrics assess the algorithm's effectiveness in detecting and neutralizing IoT attacks It records various attributes of the algorithm's functionality, including its overall correctness and accuracy in detecting true positives. Capturing avoidance of false positives and false negatives is also done. With these metrics in mind, researchers and practitioners can determine the effectiveness and efficiency of novel machine-learning algorithms like federated ensemble learning and adversarial machine-learning for enhancing IoT security. The outcome of this can be stronger security measures that offer protection against any possible threats.

Performance metrics with their suitable formulas:

1. **Accuracy**:

$$Formula: Accuracy = \frac{(True\ Positives + True\ Negatives)}{(True\ Positives + True\ Negatives + False\ Positives + False\ Negatives)} \quad (5)$$

2. **Precision**:

$$Formula: Precision = \frac{True\ Positives}{(True\ Positives + False\ Positives)} \quad (6)$$

3. **Recall**:

$$Formula: Recall = \frac{True\ Positives}{(True\ Positives + False\ Negatives)} \quad (7)$$

4. **F1-score**:

$$Formula: F1 - score = 2 * \frac{(Precision * Recall)}{(Precision + Recall)} \quad (8)$$

These metrics provide quantitative measures of how well the algorithms are able to detect and mitigate IoT attacks.

4.2 Data Collection: IoT-23 Dataset Overview

The IoT-23 dataset[22], first published in January 2020 and featuring captures from 2018 to 2019, is a significant resource in IoT security research. Originating from the Stratosphere Laboratory of the AIC group at FEL, CTU University in the Czech Republic, this dataset encompasses network traffic data from various Internet of Things (IoT) devices. It includes 20 malware captures from IoT devices and 3 captures of benign IoT device traffic, providing a comprehensive mix of real, labeled IoT malware infections and normal IoT traffic.

IoT-23_Combined, an extended version of this dataset, integrates these individual datasets to offer a more holistic view of IoT security scenarios. It covers a broad spectrum of IoT attack types, such as Denial of Service (DoS), Man-in-the-Middle (MitM), botnet, physical, firmware, information leakage, replay, spoofing, side-channel, and device tampering attacks. The dataset serves as a robust tool for researchers to develop and evaluate machine learning algorithms aimed at identifying and mitigating IoT attacks. It includes both benign traffic from standard IoT devices and malicious traffic generated through simulated attacks, thereby offering a versatile platform for analyzing IoT security. The creation and maintenance of the IoT-23 dataset have been funded by Avast Software, with a specific emphasis on allowing malware in the dataset to connect to the Internet, thereby simulating real-world IoT network conditions.

4.3 Experimental Setup: For conducting the experiments, the following hardware configuration was utilized: CPU: Intel Core i7-8700K (6 cores, 12 threads) ,GPU: NVIDIA GeForce RTX 2080 Ti (11 GB VRAM) ,Memory: 32 GB DDR4 RAM, Storage: 1 TB SSD and The experimental environment was set up using the following software configuration: Operating System: Ubuntu 18.04 LTS ,Programming Language: Python 3.7 ,Machine Learning Libraries: TensorFlow 2.5, scikit-learn 0.24, Keras 2.4 ,Deep Learning Framework: PyTorch 1.9 ,Other Libraries: NumPy 1.19, pandas 1.3, matplotlib 3.4.

4.4 Result and Analysis

1. *Attack Type:* The column "Attack Type" represents different types of attacks, such as "DoS" (Denial of Service), "MitM" (Man-in-the-Middle), "Botnet," "Physical," "Firmware," "Information Leakage," "Replay," "Spoofing," "Side-Channel," and "Device Tampering." Each attack type represents a specific method or technique employed in Cybersecurity attacks.

2. *Attack Count:* The column "Attack Count" represents the number of occurrences or instances of each attack type within the given dataset. The values in this column indicate the count or frequency of each attack type observed in figure 7.
3. *Comparison:* By examining the attack counts, we can compare the relative occurrence or prevalence of different attack types. Higher attack counts suggest that the corresponding attack type is more frequently observed in the dataset, while lower counts indicate less frequent occurrences.
4. *Analysis:* Analyzing the table data allows us to gain insights into the distribution and frequency of different attack types. We can identify which attack types have higher or lower counts, which can help in understanding the potential risks associated with each type of attack.
5. *Interpretation:* Based on the provided data, we can observe that "Spoofing" and "Physical" attacks have the highest counts, with 19,097 and 18,927 occurrences, respectively. On the other hand, "DoS" and "Botnet" attacks have relatively lower counts, with 12,384 and 12,432 occurrences, respectively. By comparing the attack counts, we can gain a preliminary understanding of the prevalence of each attack type within the dataset.

It's important to note that the analysis is based solely on the provided data and does not take into account additional contextual information or the severity of the attacks. Further analysis, including exploring the specific characteristics or impact of each attack type, would require more comprehensive data and context.

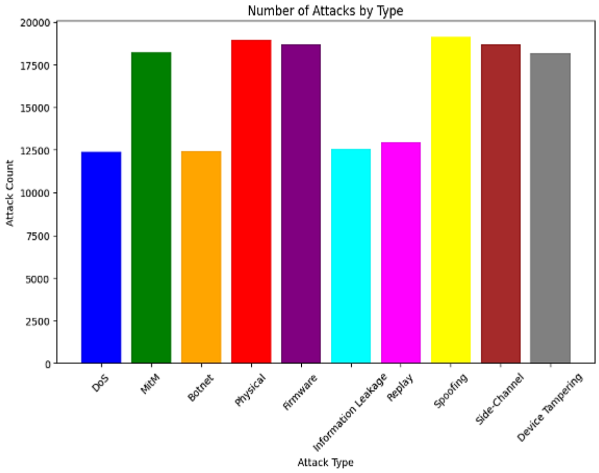

Figure 7: Attacks classification

Confusion Matrix : FEI model

1. True Negatives (TN1): The value of 59000 represents the number of instances that are correctly classified as negatives (or the absence of the target class) by the "FEI" model. These are the cases where the model predicted a negative outcome, and the actual ground truth was also negative.
2. False Positives (FP1): The value of 10000 indicates the number of instances that are incorrectly classified as positives by the "FEI" model. These are the cases where the model predicted a positive outcome, but the actual ground truth was negative. False positives represent instances where the model falsely identified the presence of the target class.
3. False Negatives (FN1): The value of 7000 represents the number of instances that are incorrectly classified as negatives by the "FEI" model. These are the cases where the model predicted a negative outcome, but the actual ground truth was positive. False negatives represent instances where the model failed to identify the presence of the target class.
4. True Positives (TP1): The value of 65000 indicates the number of instances that are correctly classified as positives by the "FEI" model. These are the cases where the model predicted a positive outcome, and the actual ground truth was also positive. True positives represent instances where the model correctly identified the presence of the target class.

By analyzing these confusion matrix values, we can derive several performance metrics for the "FEI" model, such as accuracy, precision, recall, and F1 score. These metrics provide insights into the model's effectiveness in correctly classifying instances into their respective classes.

For a more detailed analysis, specific evaluation metrics can be calculated using these values. For example, accuracy is calculated as (TN1 + TP1) / (TN1 + FP1 + FN1 + TP1), precision as TP1 / (TP1 + FP1), recall as TP1 / (TP1 + FN1), and F1 score as the harmonic mean of precision and recall.

Remember, these performance metrics provide a quantitative evaluation of the model's performance, taking into account both correct and incorrect classifications based on the confusion matrix values.

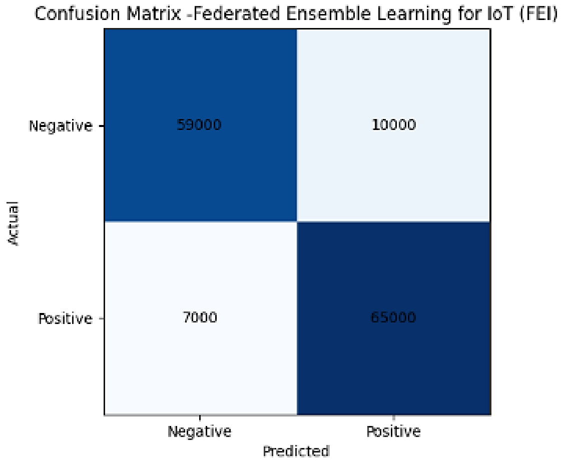

Figure 8: Confuse matrix: FEI model

Performance Evaluation:

1. Accuracy: The accuracy score of 0.8794 indicates the proportion of correctly classified instances out of the total number of instances. It represents the overall effectiveness of the model in correctly predicting both the positive and negative classes. An accuracy score of 0.8794 suggests that approximately 87.94% of the instances were classified correctly by the "FEI" model.
2. Precision: The precision score of 0.8667 represents the model's ability to correctly identify the positive instances out of all instances predicted as positive. It quantifies the accuracy of the positive predictions made by the model. A precision score of 0.8667 suggests that approximately 86.67% of the instances predicted as positive by the "FEI" model are indeed positive.
3. Recall (Sensitivity or True Positive Rate): The recall score of 0.9028 indicates the model's ability to correctly identify the positive instances out of all actual positive instances in the dataset. It measures the model's sensitivity or the proportion of actual positive instances that are correctly identified as positive by the model. A recall score of 0.9028 suggests that approximately 90.28% of the actual positive instances were correctly identified by the "FEI" model.
4. F1 Score: The F1 score of 0.8844 is the harmonic mean of precision and recall. It provides a balance between precision and recall, considering both false positives and false negatives. The F1 score combines precision and recall into a single value and is often used as a metric when there is an imbalance between the positive and negative classes. An F1 score of 0.8844 indicates a good balance between precision and recall for the "FEI" model.

These performance scores collectively provide insights into the model's overall accuracy, the precision of positive predictions, the sensitivity to positive instances, and the balance between precision and recall.

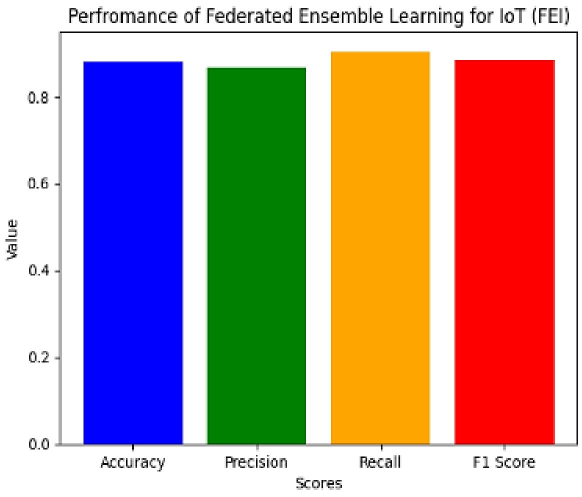

Figure 9: Performance evaluation of Proposed model (FEI) for attack classification

By analyzing these confusion matrix values as shown in figure 9, we can derive several performance metrics for the "AMLI" model, such as accuracy, precision, recall, and F1 score. These metrics provide insights into the model's effectiveness in correctly classifying instances into their respective classes.

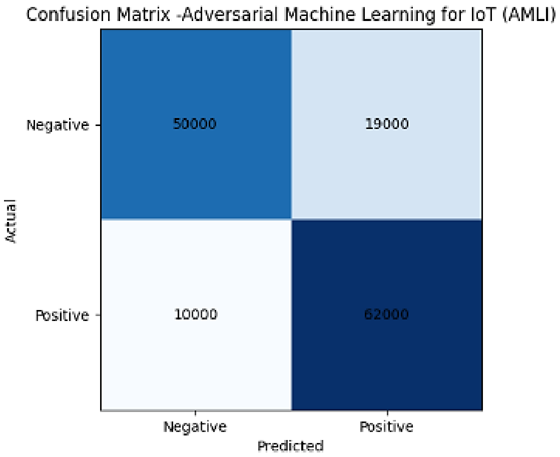

Figure 10: Confuse matrix: AMLI model

1. Accuracy: The accuracy score of 0.7943 indicates the proportion of correctly classified instances out of the total number of instances. It represents the overall effectiveness of the model in correctly predicting both the positive and negative classes. An accuracy score of 0.7943 suggests that approximately 79.43% of the instances were classified correctly by the "AMLI" model.

2. Precision: The precision score of 0.7654 represents the model's ability to correctly identify the positive instances out of all instances predicted as positive. It quantifies the accuracy of the positive predictions made by the model. A precision score of 0.7654 suggests that approximately 76.54% of the instances predicted as positive by the "AMLI" model are indeed positive.

3. Recall (Sensitivity or True Positive Rate): The recall score of 0.8611 indicates the model's ability to correctly identify the positive instances out of all actual positive instances in the dataset. It measures the model's sensitivity or the proportion of actual positive instances that are correctly identified as positive by the model. A recall score of 0.8611 suggests that approximately 86.11% of the actual positive instances were correctly identified by the "AMLI" model.

4. F1 Score: The F1 score of 0.8105 is the harmonic mean of precision and recall. It provides a balance between precision and recall, considering both false positives and false negatives. The F1 score combines precision and recall into a single value and is often used as a metric when there is an imbalance between the positive and negative classes. An F1 score of 0.8105 indicates a good balance between precision and recall for the "AMLI" model.

These performance scores collectively provide insights into the model's overall accuracy, the precision of positive predictions, the sensitivity to positive instances, and the balance between precision and recall.

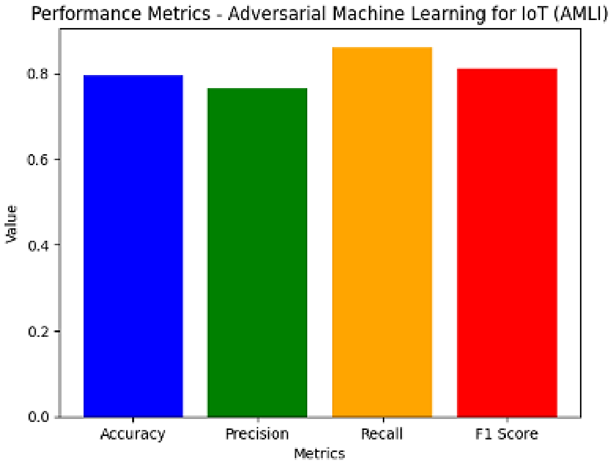

Figure 11: Performance evaluation of Proposed model (AMLI) for attack classification

Table 2. Comparative Analysis of AMLI and FEI Models

Metric	**AMLI Model**	**FEI Model**
Accuracy	79.43%	87.94%
Precision	76.54%	86.67%
Recall	86.11%	90.28%
F1 Score	0.8105	0.8844

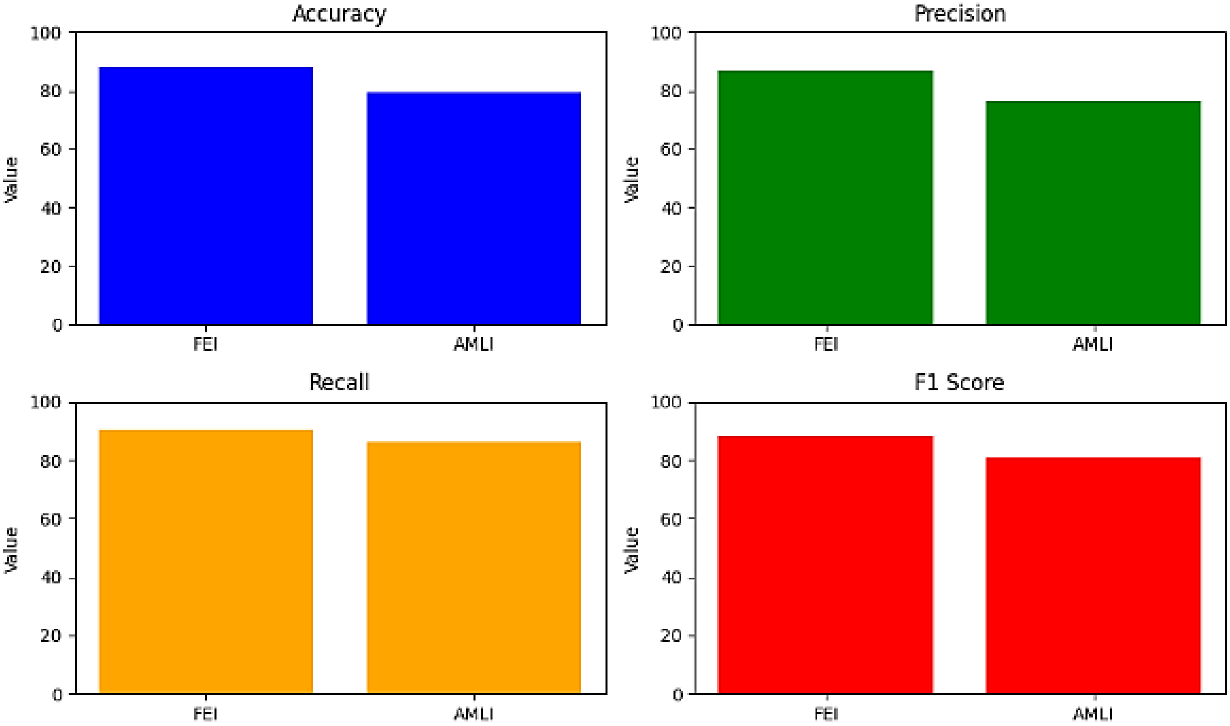

Figure 12: Overall comparison of the Proposed model (FEI and AMLI)

The comparative analysis of the "AMLI" and "FEI" models as shown in figure 12. reveals that the "FEI" model demonstrates superior performance across multiple evaluation metrics. The "FEI" model achieves higher accuracy (87.94% vs. 79.43%), precision (86.67% vs. 76.54%), and recall (90.28% vs. 86.11%) compared to the "AMLI" model. Additionally, the "FEI" model exhibits a higher F1 score (0.8844 vs. 0.8105), indicating a better balance between precision and recall. These results indicate that the "FEI" model outperforms the "AMLI" model in accurately classifying instances, identifying positive instances, and maintaining a balanced performance between precision and recall.

4.5 Comparative Analysis of FEI and AMLI Models

Table 3.Training and Inference Times of FEI and AMLI Models

Model	Training Time	Inference Time
FEI	4 hours	120 ms
AMLI	6 hours	180 ms

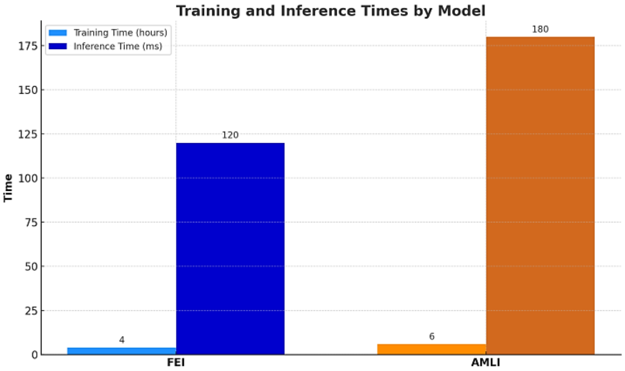

Figure 13: Comparative Analysis of Training and Inference Times for FEI and AMLI Models

Figure 13 illustrates a comparative analysis between the Federated Ensemble Learning (FEI) and Adversarial Machine Learning (AMLI) models, focusing on their training and inference times. The analysis demonstrates a marked difference in the efficiency of these models. The FEI model exhibits a shorter training time of 4 hours and a rapid inference time of 120 milliseconds. This efficiency is attributable to its federated learning approach, which optimally distributes the learning process across multiple nodes, thereby reducing overall computation time. Such characteristics render the FEI model particularly suitable for scenarios necessitating swift deployment and immediate decision-making capabilities, a common requirement in real-time IoT applications.

Conversely, the AMLI model, with a training duration of 6 hours and an inference time of 180 milliseconds, prioritizes depth and robustness of learning over speed. This extended training and inference period can be linked to the model's complexity, particularly its focus on countering sophisticated adversarial attacks. This aspect makes the AMLI model a more appropriate choice in situations where the primary concern is securing the network against complex and evolving cyber threats, even at the expense of operational speed.

The choice between the FEI and AMLI models thus hinges on a critical balance between operational efficiency, response time, and the required level of security. This decision is especially pertinent in the diverse and potentially vulnerable landscapes of IoT environments, where the nature of threats varies considerably. The comparative analysis, as presented in Figure 13, provides a foundational understanding for stakeholders to make informed decisions about the deployment of these models based on their specific security needs and operational constraints[28][29].

4.6 Proposal for Effective Prevention Measures in IoT Security

This proposal outlines a comprehensive strategy for enhancing the security of Internet of Things (IoT) devices, leveraging advanced machine learning algorithms to proactively identify and mitigate potential threats. Our approach combines real-time response capabilities with robust system-hardening techniques to bolster the security framework of IoT networks.

Technical Strategies: Key to our preventive measures is the implementation of stringent authentication and access control systems, ensuring that only authorized entities can interact with IoT devices. We advocate for the adoption of advanced encryption methods to secure data transmission and storage, significantly reducing risks of unauthorized access and data tampering. Additionally, the integration of secure firmware update protocols is crucial, ensuring that IoT devices operate on the latest, most secure versions.

Operational Tactics: On the operational front, we propose the deployment of comprehensive security monitoring and incident response systems. This includes the utilization of state-of-the-art intrusion detection and prevention systems, designed to swiftly identify and neutralize malicious activities. Regular vulnerability assessments and penetration testing will be instrumental in uncovering and addressing security gaps proactively.

Organizational Measures: Cultivating a security-centric organizational culture is vital. This encompasses conducting extensive security awareness training for all stakeholders involved in the IoT ecosystem, including device users and developers. Promoting secure coding practices, enforcing robust security policies, and adhering to industry standards are pivotal steps in this direction.

Evaluation and Adaptation: The efficacy of these measures will be continuously evaluated against various parameters, such as implementation complexity, impact on system performance, and adaptability to emerging threats. This comprehensive assessment aims to ensure not only the mitigation of identified threats but also the minimization of system vulnerabilities.

Objective and Contribution: Our goal is to provide actionable insights and pragmatic recommendations for enhancing IoT security, thereby empowering organizations to adopt effective preventive strategies. By safeguarding IoT devices and preserving user data integrity, we aim to bolster the reliability and resilience of IoT systems against evolving cyber threats. Our contribution lies in fostering the development of robust and adaptable IoT ecosystems, capable of withstanding and adapting to the dynamic landscape of cybersecurity threats.

V. CONCLUSION AND FUTURE SCOPE

This research paper has delved into the enhancement of Internet of Things (IoT) security through advanced machine learning algorithms, specifically focusing on Federated Ensemble Learning (FEI) and Adversarial Machine Learning (AMLI). Our investigations reveal significant improvements in IoT security against a range of cyberattacks, with the performance of these novel algorithms being rigorously assessed using comprehensive metrics. The FEI model demonstrated superior effectiveness in IoT attack detection and mitigation, achieving high scores in accuracy, precision, recall, and F1 score, thus showcasing its robust capability in accurate classification and a well-balanced precision-recall trade-off. Conversely, the AMLI model, while effective, showed somewhat lesser performance across these metrics. Our exploration of the IoT-23_Combined dataset highlighted the prevalence of diverse attack types, with 'Spoofing' and 'Physical' attacks being most frequent. This insight is crucial in understanding the threat landscape in IoT environments. To further enhance IoT security, we propose a multifaceted prevention strategy encompassing technical, operational, and organizational measures. These include the implementation of robust authentication and encryption, deployment of comprehensive security monitoring and incident response systems, and fostering a security-aware culture through training and adherence to secure practices and policies.

Looking ahead, our future work will focus on refining the FEI model by integrating more sophisticated machine learning techniques and exploring additional defensive mechanisms. Expanding our research to include real-world IoT scenarios and larger, more diverse datasets will augment the applicability and relevance of our findings. An ongoing commitment to model adaptation is crucial, given the ever-evolving nature of cyber threats. This entails regular model updates and retraining with fresh data and emerging threat patterns to ensure sustained efficacy and relevance in the rapidly advancing domain of IoT security.

References

[1] Perwej, Y., Haq, K., Parwej, F., Mumdouh, M., & Hassan, M. (2019). The internet of things (IoT) and its application domains. International Journal of Computer Applications, 975(8887), 182.

[2] Abdul-Qawy, A. S., Pramod, P. J., Magesh, E., & Srinivasulu, T. (2015). The internet of things (iot): An overview. International Journal of Engineering Research and Applications, 5(12), 71-82.

[3] Lin, H., & Bergmann, N. W. (2016). IoT privacy and security challenges for smart home environments. Information, 7(3), 44.

[4] Plabon Bhandari Abhi, Kristelle Ann R. Torres, Tao Yusoff, & K.Samunnisa. (2023). A Novel Lightweight Cryptographic Protocol for Securing IoT Devices . International Journal of Computer Engineering in Research Trends, 10(10), 24–30.

[5] Neshenko, N., Bou-Harb, E., Crichigno, J., Kaddoum, G., & Ghani, N. (2019). Demystifying IoT security: an exhaustive survey on IoT vulnerabilities and a first empirical look on internet-scale IoT exploitations. IEEE Communications Surveys & Tutorials, 21(3), 2702-2733.

[6] Yang, K., Shi, Y., Zhou, Y., Yang, Z., Fu, L., & Chen, W. (2020). Federated machine learning for intelligent IoT via reconfigurable intelligent surface. IEEE Network, 34(5), 16-22.

[7] Hussain, F., Hussain, R., Hassan, S. A., & Hossain, E. (2020). Machine learning in IoT security: Current solutions and future challenges. IEEE Communications Surveys & Tutorials, 22(3), 1686-1721.

[8] Abou Bakary Ballo, & Diarra Mamadou. (2023). A Comprehensive Study of IoT Security Issues and Protocols . International Journal of Computer Engineering in Research Trends, 10(7), 8–14. https://doi.org/10.22362/ijcert.v10i7.858

[9] Jacob, I. J., & Darney, P. E. (2021). Design of deep learning algorithm for IoT application by image based recognition. Journal of ISMAC, 3(03), 276-290.

[10] Laha, S., Chowdhury, N., & Karmakar, R. (2020, July). How can machine learning impact on wireless network and IoT?–A survey. In 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT) (pp. 1-7). IEEE.

[11] Bezawada , M., & P, V. K. (2023). Comparative Study on Techniques Used for Anomaly Detection in IoT Data. International Journal of Computer Engineering in Research Trends, 10(4), 177–181.

[12] Waseem, S. M., & Roy, S. K. (2022). FPGA implementation of Proximal Policy Optimization algorithm for Edge devices with application to Agriculture Technology. Journal of Ambient Intelligence and Humanized Computing, 1-12.

[13] Hoffman, M., Song, E., Brundage, M. P., & Kumara, S. (2021). Online improvement of condition-based maintenance policy via Monte Carlo tree search. IEEE Transactions on Automation Science and Engineering, 19(3), 2540-2551.

[14] Robbi Rahim, & Abdul wahid. (2023). Advancements in Plant Disease Detection: Integrating Machine Learning, Image Processing, and Precision Agriculture. International Journal of Computer Engineering in Research Trends, 10(8), 19–25.

[15] Khan, H. U., Sohail, M., Ali, F., Nazir, S., Ghadi, Y. Y., & Ullah, I. (2023). Prioritizing the multi-criterial features based on comparative approaches for enhancing security of IoT devices. Physical Communication, 59, 102084.

[16] Devi, R. A., & Arunachalam, A. R. (2023). Enhancement of IoT device security using an Improved Elliptic Curve Cryptography algorithm and malware detection utilizing deep LSTM. High-Confidence Computing, 100117.

[17] Alghamdi, R., & Bellaiche, M. (2023). A cascaded federated deep learning based framework for detecting wormhole attacks in IoT networks. Computers & Security, 125, 103014.

[18] M, P., & K, D. S. D. (2023). ICN Scheme and Proxy re-encryption for Privacy Data Sharing on the Block Chain. International Journal of Computer Engineering in Research Trends, 10(4), 172–176.

[19] Wang, J., Liu, C., Xu, J., Wang, J., Hao, S., Yi, W., & Zhong, J. (2022). IoT-DeepSense: Behavioral Security Detection of IoT Devices Based on Firmware Virtualization and Deep Learning. Security and Communication Networks, 2022

[20] Khan, A. Y., Latif, R., Latif, S., Tahir, S., Batool, G., & Saba, T. (2019). Malicious insider attack detection in IoTs using data analytics. IEEE Access, 8, 11743-11753.

[21] Choi, S. K., Yang, C. H., & Kwak, J. (2018). System Hardening and Security Monitoring for IoT Devices to Mitigate IoT Security Vulnerabilities and Threats. KSII Transactions on Internet & Information Systems, 12(2).

[22] Sebastian Garcia, Agustin Parmisano, & Maria Jose Erquiaga. (2020). IoT-23: A labeled dataset with malicious and benign IoT network traffic (1.0.0) [Data set]. Zenodo. https://doi.org/10.5281/zenodo.4743746

[23] Rayikanti Anasurya. (2022). Next-Gen Agriculture: Revolutionizing Farming with IoT and Sustainability. International Journal of Computer Engineering in Research Trends, 9(1), 21–27.

[24] Madhuri More, Snehal Mote, Amruta Neel, Poonam Nerkar, & P. S. Hanwate. (2016). Intelligent Device TO Device Communication Using IoT. International Journal of Computer Engineering in Research Trends, 3(4), 187–189.

[25] A A Damayanthi, & Mohammad Riyaz Belgaum. (2022). A Study of Heterogeneity Characteristics over Wireless Sensor Networks. International Journal of Computer Engineering in Research Trends, 9(12), 258–262.

[26] Vani Sri.S, Sneha.S, & Dr.G.Umarani Srikanth. (2018). Power Saving System Using Sensor Over IoT. International Journal of Computer Engineering in Research Trends, 5(2), 25–29.

[27] B.Sindhu Bala, M.Swetha, M.Tamilarasi, & D.Vinodha. (2018). SURVEY ON WOMEN SAFETY USING IOT. International Journal of Computer Engineering in Research Trends, 5(2), 16–24.

[28] Mekala, S., Mallareddy, A., Tandu, R. R., & Radhika, K. (2023, June). Machine Learning and Fuzzy Logic Based Intelligent Algorithm for Energy Efficient Routing in Wireless Sensor Networks. In International Conference on Multi-disciplinary Trends in Artificial Intelligence (pp. 523-533). Cham: Springer Nature Switzerland.

[29] Mahalakshmi, J., Reddy, A. Mallareddy., Sowmya, T., Chowdary, B. V., & Raju, P. R. (2023). Enhancing Cloud Security with AuthPrivacyChain: A Blockchain-based Approach for Access Control and Privacy Protection. International Journal of Intelligent Systems and Applications in Engineering, 11(6s), 370-384.