mathematics

MDPI

Article

Federated Learning-Inspired Technique for Attack Classification in IoT Networks

Tariq Ahamed Ahanger [1,*], Abdulaziz Aldaej [1], Mohammed Atiquzzaman [2], Imdad Ullah [1] and Muhammad Yousufudin [1]

[1] College of Computer Engineering and Sciences, Prince Sattam Bin Abdulaziz University, Al-Kharj 11942, Saudi Arabia; a.aldaej@psau.edu.sa (A.A.); i.ullah@psau.edu.sa (I.U.); m.yousuf@psau.edu.sa (M.Y.)

[2] School of Computer Science, University of Oklahoma, Norman, OK 73019, USA; atiq@ou.edu

* Correspondence: t.ahanger@psau.edu.sa

Abstract: More than 10-billion physical items are being linked to the internet to conduct activities more independently and with less human involvement owing to the Internet of Things (IoT) technology. IoT networks are considered a source of identifiable data for vicious attackers to carry out criminal actions using automated processes. Machine learning (ML)-assisted methods for IoT security have gained much attention in recent years. However, the ML-training procedure incorporates large data which is transferable to the central server since data are created continually by IoT devices at the edge. In other words, conventional ML relies on a single server to store all of its data, which makes it a less desirable option for domains concerned about user privacy. The Federated Learning (FL)-based anomaly detection technique, which utilizes decentralized on-device data to identify IoT network intrusions, represents the proposed solution to the aforementioned problem. By exchanging updated weights with the centralized FL-server, the data are kept on local IoT devices while federating training cycles over GRUs (Gated Recurrent Units) models. The ensemble module of the technique assesses updates from several sources for improving the accuracy of the global ML technique. Experiments have shown that the proposed method surpasses the state-of-the-art techniques in protecting user data by registering enhanced performance measures of Statistical Analysis, Energy Efficiency, Memory Utilization, Attack Classification, and Client Accuracy Analysis for the identification of attacks.

Keywords: federated learning; security; DDoS attack; Internet of Things

MSC: 68P25; 68P30

Citation: Ahanger, T.A.; Aldaej, A.; Atiquzzaman, M.; Ullah, I.; Yousufudin, M. Federated Learning-Inspired Technique for Attack Classification in IoT Networks. *Mathematics* **2022**, *10*, 2141. https://doi.org/10.3390/math10122141

Academic Editors: Zhongyun Hua, Yushu Zhang, János Sztrik and Daniel-Ioan Curiac

Received: 29 April 2022
Accepted: 16 June 2022
Published: 20 June 2022

1. Introduction

Internet of Things (IoT) technology represents an inter-network of digitally interconnected sensors that can automate a wide range of functions. Smart solutions based on artificial intelligence (AI) are quickly becoming the norm in today's digital world. Several industrial machine learning (ML) solutions rely on IoT's micro-service architecture because it is an effective platform for application deployment. Incorporating ML into mini-compatible hardware architectures in IoT allows it to support AI-enabled services. Intelligent digital assistants, smart homes, and the Industrial IoT (IIoT) are all examples of how IoT devices are improving services in several industries. It has been demonstrated that IoT devices are excellent at delivering AI-empowered decision-making. Notably, however, IoT relies on sensitive end-user information. Because IoT devices must operate with little power consumption, traditional security firewalls can not be used to protect them, leaving them vulnerable to a variety of attacks. As described by Kolias et al. [1], malware bots such as Mirai can take advantage of IoT device vulnerabilities to gain control of their AI

functions. As a result, access to non-IoT systems inter-connected with IoT devices can be granted.

1.1. Research Domain

Unprotected IoT devices pose significant vulnerability to the entire network. Physical devices and the digital world must be connected via a network protocol in IoT networks. An investigation of IoT security vulnerabilities has been conducted by Neshenko et al. [2], Zhou et al. [3], and Panchal et al. [4] to discuss numerous attack-types and the corresponding impact on IoT sub-systems. There has been a rapid increase in vulnerable activities that exploit the security gaps of IoT networks [5]. Moreover, there is a huge demand for IoT devices owing to the popularity of microdevices that provide intelligent digital assistance and have been demonstrated to minimize manual labor. Many IoT devices have been produced with poor design choices, which has resulted in vulnerable hardware and extremely insecure IoT devices providing and exchanging digital information. It is difficult to set up an ML-based anomaly detection approach because of the wide variety of IoT sensors and regular training requirements to ensure performance optimality. Conspicuously, both developers and end-users are concerned about IoT security. IoT networks have been considered a core research domain by several authors in recent years. As a result, ML-based solutions for analyzing network breaches have become increasingly commonplace.

1.2. Research Motivation

Due to disadvantages (such as the requirement that all training data be stored on a centralized repository), security issues associated with transmitting acquired data from IoT sensors to the server, and computational costs associated with training large volumes of data on a unified server, ML-inspired decision-making is rarely preferred. The Federated Learning (FL) technique is one of the most promising and adaptable ways to address shortcomings of the ML-based approach. Decentralized ML model training in the FL technique preserves the data over the edge device and transfers the trained ML attributes to the centralized server. Comparative to conventional FL systems, the method of FL is proved to protect user data, making it the preferable option. Conspicuously, an ensemble-based solution to anomaly detection for IoT networks is proposed in the current study. It is possible to train ML models for anomaly detection on IoT networks using the presented technique, which does not require network data to be sent to a centralized server. Specifically, Long Short-Term Memory (LSTM) and Gated Recurrent Units-Neural Network (GRU-NN) models are employed for training ML effectively over the Modbus data to achieve effective outcomes in detecting IoT intrusion. In comparison with the state-of-the-art ML technique, the presented experimental findings show a reduced error in attack prediction with minimal false alarms. Some of the main contributions of the presented study are:

1. Federated Learning (FL)-based procedure has been proposed to ensure IoT-device security.
2. Attribute migration from a local node to a global node in the IoT framework has been proposed for effective attack identification.
3. Attack detection has been performed using the proposed FL mechanism based on the deep learning technique of Long Short-term memory (LSTM)
4. The performance of the proposed model has been assessed in terms of Statistical Analysis, Energy Efficiency, Memory Utilization, Attack Classification, and Client Accuracy Analysis.

Based on the aforementioned aspects, Figure 1 shows a generic overview of the proposed technique at the architectural level.

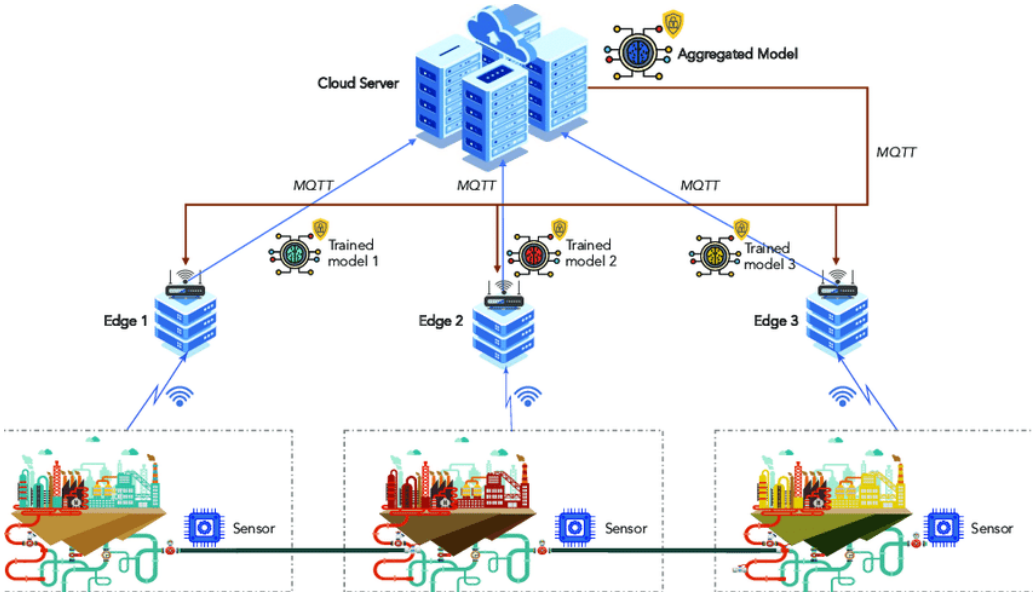


Figure 1. Conceptual View of IoT-Edge-Cloud for Anomaly Detection.

Paper Organization

Section 2 reviews some of the related studies in the current domain. Section 3 presents the proposed approach based on FL for attack classification. Section 4 summarizes the experimental results for validation purposes. Finally, Section 5 concludes the paper with future research directions.

2. Related Work

IoT microarchitecture design has been demonstrated to be successful in delivering ML solutions. As IoT has grown in popularity and usage, several research directions have been explored by researchers. The detection and classification of IoT network attacks represents one such research direction. Numerous research efforts have been presented to protect IoT networks from harmful attacks. In the current section, state-of-the-art research has been reviewed that proposes ML-inspired approaches to enhance the security of IoT networks. Waqas et al. [6] suggested that malicious Mirai-infected IoT devices can be detected by auto-learning sub-systems called BoTs, which are built on FL-inspired methods with an adaptive learning approach for IoT networks in the smart home. IoT security services and gateways are part of BoT's architectural framework. IoT relies on security gateways to connect IoT devices to the internet. The updated weights from IoT sensors are aggregated by a security module, which formulates device-specific storage for anomaly identification techniques. Based on the experimental simulation, enhanced accuracy and specificity were registered for the proposed model. Thom et al. [7] proposed a technique to monitor the IoT network's traffic using device-specific anomaly identification when new devices are added. A self-learning algorithm eliminates the need for an attack labeling since BoT learns the pattern associated with each attack type. False alarms in the detection of attacks are decreased, according to evaluation findings. However, there was no deep learning framework for FL, and the method was restricted to single (Mirai) attack types. Moreover, limited performance was registered for the presented approach in terms of battery power management. Deepfed, an FL-inspired intrusion identification system, was introduced by Li et al. [8] to identify risks in the internet-connected framework. GRU and Convolutional Neural Networks (CNN) are utilized to identify threats, while Paillier cryptosystems are employed to assure the privacy of both global and local models during learning. Based on experimental simulations, performance was comparatively better in terms of statistical measures. FedAGRU, an FL-based attention-gated recurrent unit, was proposed by Chen et al. [9]. To combat poisoning attacks, FedAGRU is designed to discover and discard minimal updates for an effective global model that minimizes communication costs. How-

ever, energy consumption was maximized for the presented approach in the simulation trials. An FL-based technique for wireless intrusion detection (WID) was also proposed by Cetin et al. [10] using a large data set. Two FL techniques were implemented using a mimic learning technique, which was combined with the ML-based intrusion detection system by Al-Athba et al. [11]. Experimental validation depicted enhanced performance for the proposed approach. The TensorFlow federated (TFF3) framework is another FL-based solution proposed in Rahman et al. [12]. However, the solution lacks in cost minimization and data loss. ML-inspired intrusion identification provided in Rouzbahani et al. [13] is comparable to the current work, which proposes a TensorFlow-based DL system for centralized anomaly detection. A threat detection algorithm employs six LSTMs as presented by Breux et al. [14]. Deep learning with PySyft by Ryffel et al. [15] implements FL and GRU, which further boosts the model's efficiency. Mothukuri et al. [16] proposed an ML-inspired anomaly identification method based on the detection of abnormalities in intelligent home sensors. To detect attacks, the authors recommended the logistic regression technique and the basic ANN classification algorithm. The authors focused on discovering data attack trends and utilizing a simple categorization technique that is non-adaptive to an ever-changing spectrum of IoT sensors. Significant performance was registered in comparison to the state-of-the-art decision-making models. To identify abnormalities in temporal data of industrial IoT applications, Li et al. [17] employed the attention-based CNN with LSTM. FL is implemented using the PySyft (Source: https://blog.openmined.org/install/, accessed on 28 April 2022) and PyTorch (Source: https://pytorch.org/, accessed on 28 April 2022). A gradient compression approach is presented to increase communication efficiency. Comprehensively, the state-of-the-art research on anomaly detection in IoT networks lacks an effective decentralized communication infrastructure. Henceforth, these constraints are incorporated to propose an FL-based method for IoT security threats. Moreover, Table 1 has been formulated to depict the comparative analysis with state-of-the-art research works in the current domain.

Table 1. State-of-the-art Comparison (Yes: Available, –: Not Available).

References	**[14]**	**[18]**	**[12]**	**[11]**	**[16]**	**[17]**	**[10]**	**Proposed**
Security	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
IoT	–	Yes	Yes	–	Yes	No	–	Yes
Quantification	Yes	Yes	Yes	Yes	Yes	–	Yes	Yes
Data Repository	Yes	Yes	Yes	Yes	–	–	Yes	Yes
Energy Efficiency	–	–	–	–	–	Yes	Yes	Yes
Federated Learning	–	Yes	Yes	–	Yes	–	–	Yes
Anomaly Prediction	–	Yes	–	–	–	–	–	Yes

3. Proposed Model

Figure 2 shows the conceptual architecture of the proposed model [16]. Several important components have been formulated for effective classifications of the attacks in the IoT scenarios. The detailed functionality of each component has been provided ahead in detail.

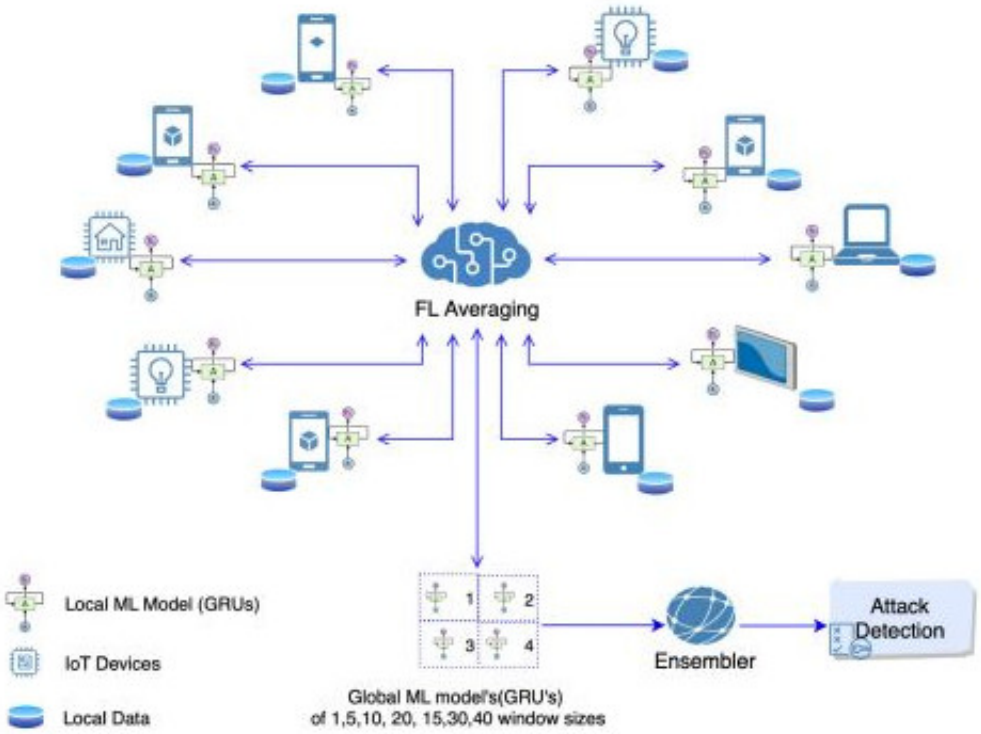


Figure 2. FL-based Attack Classification.

3.1. Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRUs)

The short-term memory/vanishing gradients problem can be addressed using the heterogeneous LSTM and GRUs models. Long-term dependencies can be learned by using LSTMs and GRUs, which include gates for monitoring data transmission and controlling the training procedure. Gates operate as switches, preserving both short-term and long-term data. LSTMs and GRUs have been used in a variety of real-time applications, including anomaly identification, voice detection, speech synthesis, and textual production. Both GRUs and LSTM models were tested during the assessment of the presented strategy, with GRUs models showing a greater accuracy rate and being less computationally expensive than LSTMs in the early FL training rounds. Some of the important aspects of GRU and LSTM are detailed ahead.

1. *Sigmoid Function*: Provides a means of determining whether or not any data should be retained or destroyed. Data in the network can be forgotten if a value close to 0 is generated, whereas data that must be maintained for future updates can only be generated when the value is greater than 1.
2. *Tangent Hyperbolic (tanh):* It is possible to obtain values between 0 and 1 by using an activation function. This assures that non-positive measures are mapped severely, whereas non-negative values are mapped from 0 to 1. As the mean value of the *tanh* function for the hidden layers makes learning considerably easier for subsequent layers, it is chosen for neural networks (NN).
3. *Cell State*: This is a representation of the data that have been stored in the LSTM's memory block. The respective cell state is represented by D_u, while the prior cell state is represented by D_{u-1}. LSTMs are presented as memory cells inspired by the human propensity to remember a similar pattern, where linked memory cells gather and store long-term reference information.
4. *LSTM's gates:* Memory cells are controlled by gates, which are used to regulate data storage, retrieval, and deletion. There are three gates in LSTMs including *input gate*, *forget gate*, and *output gate*. Each gate has several specific attributes. It is important to understand how the LSTM network discards information that does not contribute to the network's learning.

(a) *Forget Gate:* The data that are non-essential to the training process for one cell are analyzed by this gate. Mathematically,

$$g_u = \pi(X_g[g_{u-1}, y_u + c_g])$$

where g_u is the recent input measure of forget gate, y_u is the current input to memory cell X_g, c is bias value, and g_{u-1} is the last data value of the cell.

(b) *Input Gate:* Assists in the determination of how relevant current information is for storage in the cell state. The input gate's value is determined by multiplying it by the aggregate of the forget gate g_u and the last cell state d_{u-1}, as well as the input gate's current state.

$$j_u = \pi(X_j[i_{u-1}, y_u + c_j])$$

$$c_{cu} = \tanh\left(X_d[i_{u-1}, y_u + c_d]\right)$$

$$c_u = g_u * d_{u-1} + j_u * c_{cu}$$

d_u is an activating function that arises from the sigmoid layer. Information that is expected to be relevant for storage as future reference is calculated by using the current cell value in conjunction with d_{u-1}, the prior timestamp memory cell state established by the *tanh* layer.

(c) *Output Gate:* It is here that the network's ultimate output is determined. The *tanh* activation function is used to determine i_u from p_u.

$$p_u = \pi(X_p[i_{u-1}, y_u + c_p])$$

$$i_u = p_u * \tanh(d_u)$$

where i_u is the output measure of the current cell and p_u is the output gate measure, which is adjusted to a non-negative measure below 1 by utilizing the *sigmoid* function.

5. *GRU*: Compared to LSTMs, the construction of GRUs is simpler. Only two gates are needed to operate the memory cells: *Reset gate and Update gate.*

(a) *Reset Gate*: GRUs are more cost-effective and time-efficient to train. In the same way that LSTMs discard information that is irrelevant for future learning or reference, Reset Gates do the same.

$$s_u = \pi(X_s[i_{u-1}, y_u])$$

where the output of the *sigmoid* function for the memory cell of the reset gate is s_u, the information from the preceding memory cell is i_{u-1}, and the current input for the memory cell is y_u.

(b) *Update Gate:* GRUs utilize an update gate to determine whether information from the current state requirement is to be retained.

$$a_u = \pi(X_z[i_{u-1}, y_u])$$

$$i_{cu} = \tanh(X[s_u * i_{u-1}, y_u])$$

$$i_u = (1 - a_u * i_{u-1} + a_u * i_{cu})$$

where a_u is the output of the sigmoid layer, i_u is the vector obtained from the *tanh* function, and i_{u-1} is the preceding measure of the cell state.

The input of LSTM/GRU varies depending on the window size. The quantity of information varies for every window, which helps the ML model perform better, making window size selection critical. The training time is impacted by increasing the window size because the information kept in every storage cell of the NN also rises.

There is no established correlation between the window size and model performance that can be verified. Gonzalez et al. [19] suggested that the effect of the size of the window and LSTM layers is based on the amount of data.

3.2. Framework Design

To discover AI-enabled anomalies in IoT networks, an FL-based technique has been proposed. For a better-performing ML model, it is critical to select the appropriate window size because the number of data changes depending on the window size. The training time is affected by increasing the window size length since the amount of information stored in each neural network memory cell also grows [16]. In the current paper, seven different window sizes have been used. For each window size, a global DL model and an Ensemble technique composed of a Random Forest Decision Tree (RFDT) make up the approach's high-level architecture as shown in Figure 3. It comprises logical instances that represent networked sensors. A local DL model is also included for copying training data for every logical instance. In the current section, a detailed procedure is presented to implement the proposed strategy. In practice, there is no need to create virtual instances or pre-process recorded data on a central server because actual data are already available at the end-devices for training.

1. *Logical Instances:* PySyft is used to create virtual instances of the IoT network. The fl_n endpoints are virtualized, and a special instance called $fl_{average}$ is created to emulate the central server, allowing the learned ML model parameters to be shared between the fl_n endpoints and the central FL server. There are n virtual instances for each piece of data in the data collection.
2. *Preprocessing Acquired Data*: The *CICFlowmeter* (Source: https://github.com/ahlashkari/CICFlowMeter, accessed on 28 April 2022) utility is used to convert *.pcap* files to *.csv* files for pre-processing at every gateway that acts as a bridge between the sensing component and cloud. To remove aspects that are unnecessary for the learning process, the *.csv* file is transformed multiple times. Afterward, n pieces of the processed data are split up and dispersed across the IoT end-device's virtual instances.
3. *FL training*: IoT instances are accessible for FL training at any given time, and the training is performed asynchronously. $\mathrm{Fl_{average}}$ aggregating instance collects the weights from each node's trained local ML model and distributes them with the other nodes. $\mathrm{Fl_{average}}$ aggregation is used since it is an efficient algorithm for distributed training with an enormous number of clients [20]. Moreover, Ratio Loss is used as the representative method aiming to address the local imbalance by analyzing the local data distribution [21]. The ratio loss function is computed mathematically as

$$\text{Ratio-Loss} = (\beta + \alpha \mathrm{R}) * \mathrm{p} * (-\mathrm{p} * \log(\mathrm{s}))$$

where $(-\mathrm{p} * \log(\mathrm{s})$ is the cross-entropy loss function which has the true measure of p and probabilistic result S. β and α are the hyper-parameters. Furthermore, the utilization of the ratio loss function does not require users/clients to upload their overall sample quantities which ensures privacy. Additionally, the privacy protection in FL training is also guaranteed by secure aggregation protocols and differential privacy techniques [22]. Two types of GRUs have been employed in the current research as shown in Table 2. Training rounds in FL have been described in terms of the number of epochs executed by every terminal device. The FL training logic along with several stages of the proposed solution are presented in Algorithm 1 of Figure 4. An IoT end-device can be represented as an $\mathrm{fl_i}$ virtual instance. Finally, for each window size, $\mathrm{X_i}$ defines the model parameters GRU-ML for the network. GRU-ML is distributed to each $\mathrm{fl_i}$ virtual instance. $\mathrm{Fl_i}$'s local ML model updates are communicated with $\mathrm{fl_{average}}$ throughout each training cycle on GRU-ML using the secure socket layer protocol for data protection. Training grounds have been formulated on a multiprocessor to mimic the real-world situation. $\mathrm{Fl_i}$ training rounds are performed on different processors, and the learned local model weights ($\mathrm{mw_i}$) are

regularly shared with the average virtual instance. For example, the central server uses $fl_{average}$ virtual instance to collect and listen for local model updates (mw_i) from clients. Local ML model weights are combined to produce the global ML model (Mw_i) for each window size. Every terminal device should receive a copy of Mw_i.

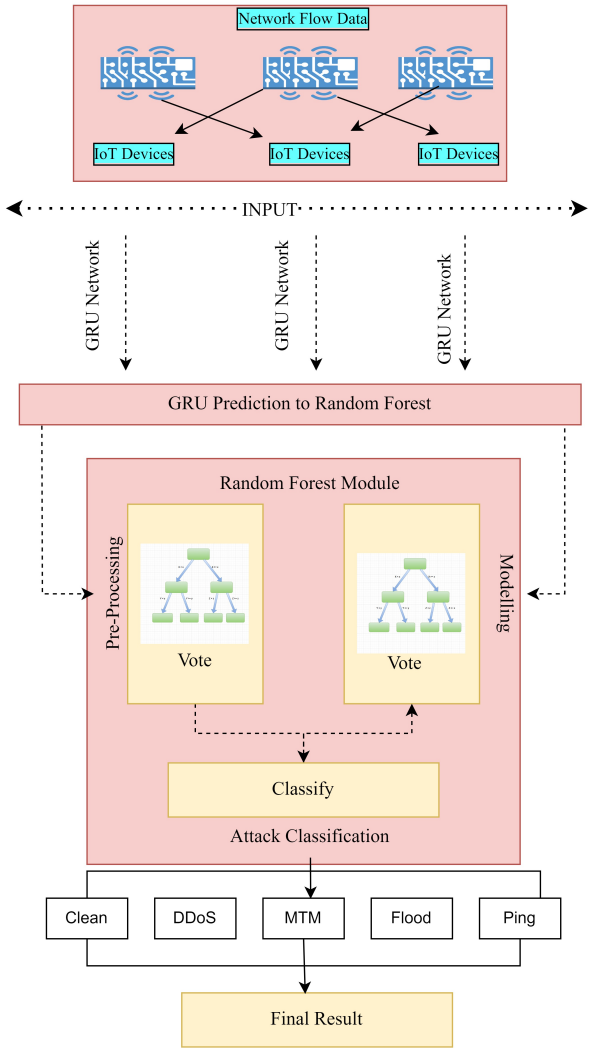


Figure 3. Illustration for Attack Classification.

4. *Ensemble Learning Technique*: An effective method for combining the results of many machine learning models is provided by Ensemble Learning [23]. It is typically linked to the well-established idea that integrating several ML models yields better results than using a single ML model. Ensembles of seven global ML models (Mw_i) are constructed using the random forest decision tree classifier. Each Mw_i forecasts the probability values $i_1, i_2, \ldots, i_n$ of each label Z for the given input Y, for example, $Y = Y_1, \ldots, Y_n$. To make an ensemble prediction function, the Ensemble uses the probability values of Mw_i to construct an ensemble prediction function g(y). Mathematically.

$$i_j = z_{cj}(N_{xj}(Y))$$

$$g(y) = \arg\max \sum_{k=1}^{K} J(z = i_k(y))$$

A total of seven ML models Mw_i were used to forecast each label $Z = Z_{Clean}, Z_{MITM}, \ldots Z_{pingDDos}$ which indicate the class of the attack in the data set, and the prediction probabilities of each model were compared. There are many machine-learning models in use, and each one employs its probabilities to cast its vote on the final label.

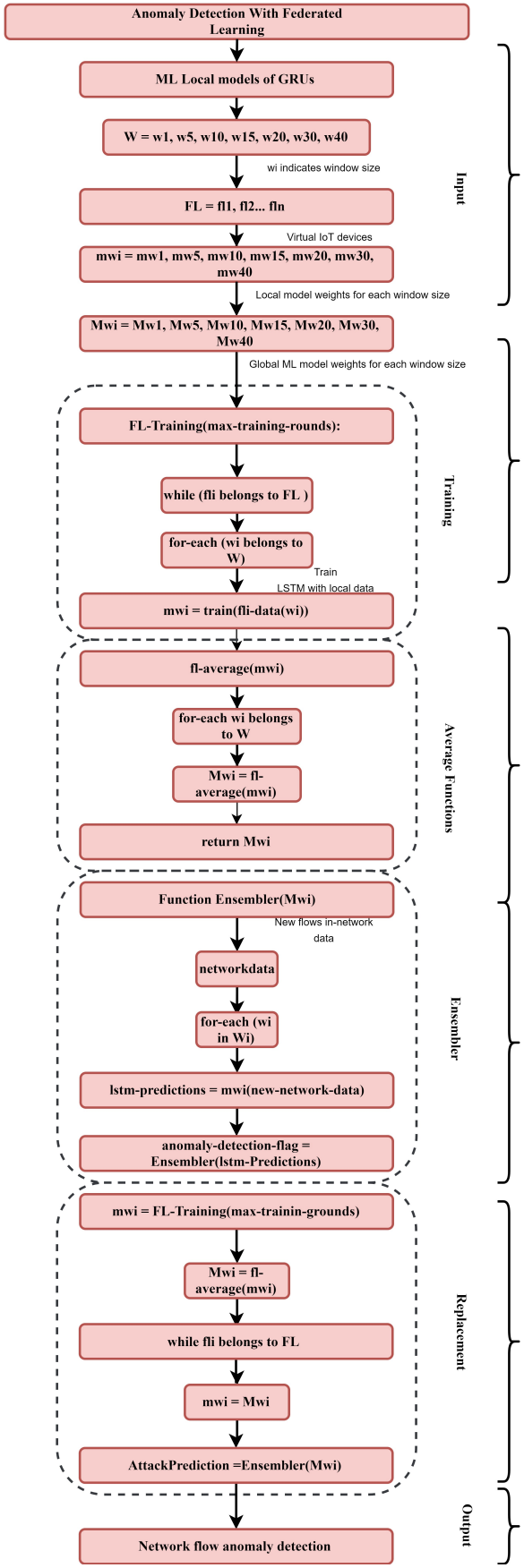


Figure 4. Algorithm 1: Proposed Technique of Attack Classification.

Table 2. GRUs used in the proposed model.

GRU Model	Layer	Dropout (in %)	Inter-Layer Size
1	3	2	128
2	1	1	256

4. Experimental Set-Up

The current section presents a data set and assessment measures that compare the performance of the proposed approach with state-of-the-art deep learning (DL) techniques. The HP laptop system (with a graphics processing unit bearing a 3.1 GHz clock cycle), running Ubuntu 18.1.0 LTS with 16 GB RAM and 1 TB hard disk is used for the federated environment. PySyft has been utilized for FL features and GRUs as the proposed ML neural network for DL. The Pytorch DL framework was utilized for the classic ML technique. FL uses Algorithm 1 with datasets for training. For comparative analysis, numerous state-of-the-art studies/techniques have been used. Specifically, three challenging deep learning studies/techniques have been utilized for performance assessment including Campos et al. [18] Ferrag et al. [24], and Friha et al. [25].

4.1. Data Set

TON_IOT datasets (Dataset 1) were used to test the performance of the proposed technique [26]. For Dataset 1, Train_Test_Dataset was used in CSV format for evaluating the efficiency of the proposed technique. The number of records includes normal and attack types for testing and training the ML algorithm. Moreover, several instances were bootstrapped to 60,259. Modbus is utilized to establish communication between IoT devices and server [27]. To abstract ML-readable CSV from collected networked data, *CICFlowmeter* is utilized. The following types of attacks have been considered for validating the performance assessment of the proposed model.

1. *Man in the Middle Attack*: As the term indicates, it involves a third-party entity (the attacker) that pretends to be either the sender or recipient during a conversation and attempts to steal information or execute activities as the sender or receiver to gain access to sensitive data. Having gained access to traffic control, the attacker can then generate bogus transactional data.
2. *Ping DDoS Flood Attack:* This is the most popular type of Distributed Denial-of-Service (DDoS) attack, in which the server is overwhelmed with pings from the attacker, forcing it to go offline and block any further connections.
3. *Query Flood Attack:* When an attacker delivers a flood of messages to overload an end device and prevents it from serving legitimate communications.
4. *SYN DDoS Attack*: It is possible to use a Syn DDoS attack to block the server from accepting any new connections by repeatedly sending *syn* packets to the server to begin a connection handshake and keep all ports busy. The SYN DDoS attack is often carried out by a bot that makes a large number of connection requests disguised as fake IP addresses.

4.2. Model Training

The proposed model was trained for more than 400 rounds with 2 epochs for each training round. It is a hyperparameter that determines how many epochs a learning algorithm will run over the complete training dataset for each client. During a single epoch, each training dataset sample is only used once to update the model's internal parameters. It is also implemented using the sci-kit-learn SGDClassifier to build the logistic regression algorithm. As mentioned earlier, $Fl_{average}$ is the aggregation function used in the current study. The data are normalized before the ML/DL is applied. In addition, the training and testing sets were divided into an 85/15 split.

4.3. Metrics for Assessment

It is common practice in ML to compare model predictions against actual values, to determine the proportion of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) in the data. The number of times the ML model's predictions match up with genuine labels or actual values is represented by TP and TN, whereas FP and FN show the number of times the ML model's predictions are erroneous. The following measures have been used to evaluate the proposed strategy and compare it with state-of-the-art techniques.

1. *Statistical Analysis*
   (a) Accuracy = $\frac{TP+TN}{TP+TN+FP+FN}$
   (b) Precision = $\frac{TP}{TP+FP}$
   (c) Recall = $\frac{TP}{TP+FN}$
   (d) F1 = $\frac{2*Precision*Recall}{Precision+Recall} = \frac{2*TP}{2*TP+FP+FN}$
2. *Energy Efficiency*: Amount of power utilized over time
3. *Memory Utilization*: Memory utilization over large datasets
4. *Attack Classification Analysis*
5. *Client Accuracy Analysis*
6. *Non-Independent and Identically Distributed (Non-IID) and Independent and Identically Distributed (IID) Analysis*.

To evaluate the trained ML models, Skorch (Source: https://github.com/skorch-dev/skorch, accessed on 28 April 2022) and Scikit-learn (Source: https://scikit-learn.org/stable/, accessed on 28 April 2022) have been utilized. As the proposed technique is based on the Pytorch DL framework, the skorch wrapper is used to access the Scikit-learn evaluation packages.

4.4. Results

The proposed model is validated based on the three types of GRU models as shown in Table 2. As mentioned earlier, numerous statistical parameters have been calculated in comparison to state-of-the-art studies of Logistic Regression (LR [18]), Recurrent Neural Network (RNN [24]), and Deep Neural Network (DNN [25]) to determine the performance enhancement of the proposed approach. The window size is considered for different scenarios using linear-quadratic-linear functions for better assessment of the proposed model. A detailed explanation of the results is described ahead. However, only the average of the results is depicted for various window sizes for effectiveness.

1. Statistical Results
   (a) *GRU-Model 1:* Figure 5 shows the results for comparison for GRU-Model 1. Statistical parameters of accuracy, precision, recall, and F1-measure have been calculated. It can be seen that the proposed FL-based attack classification is more accurate by registering an enhanced measure of 95.65% in comparison to LR (94.65%), RNN (92.36%), and DNN (90.26%). For precision analysis, the proposed technique of GRU-LSTM can outperform the deep learning techniques of LR (93.78%), RNN (92.01%), and DNN (89.16%) by registering an enhanced measure of 93.65%. Similarly, the f-measure and recall measure were analyzed for the proposed model. In the current scenario, the proposed model registered enhanced values of 94.65% and 93.26% for f-measure and recall, respectively. On the other hand reduced values for LR (92.65%, 93.54%), RNN (90.12%, 89.99%), and DNN (88.21%, 87.59%) were acquired. This shows

that the proposed model is more effective in statistical parameters as compared to other techniques.

(b) *GRU-Model 2:* Figure 6 shows the performance comparison for GRU-Model 2 for various window sizes. It can be seen that in the current scenario, the proposed FL-based model for attack detection is more effective and efficient. Specifically, statistical performance of 93.10% (accuracy), 93.12% (recall), 94.96% (precision), and 95.45% (F1-Measure) was registered for the proposed technique for variable window sizes. On the other hand, the LR model was able to register an average measure of 90.23% (accuracy), 89.15% (recall), 88.12% (precision), and 90.32% (F1-Measure). Similarly, for RNN and DNN techniques, 89.56%, and 88.23% values were acquired for accuracy, 87.78%, and 87.34% values were registered for recall, 86.89%, and 87.00% values were acquired for precision, and 87.14% and 87.96% values were registered for F1-measure. Henceforth, in the current scenario, the proposed model is better and more effective at classifying attacks in the IoT scenario using the GRU-LSTM technique.

2. *Energy Efficiency*: The energy efficiency pertains to the effectiveness of the proposed model in terms of power consumption. The proposed model was deployed over the computing system with monitored energy utilization. The comparative analysis was performed with different state-of-the-art techniques as shown in Figure 7. It can be seen that in the current scenario for GRU-Model 1, the proposed model can utilize a minimal average energy of 365.56J in comparison to LR (400.56J), RNN (456.59J), and DNN (499.58J) over the variable number of data instances. Similarly, for GRU-Model 2, the presented approach is more effective as it utilizes a minimal energy of 655.56J in comparison to LR (756.89J), and RNN (865.78J), and DNN (956.78J). This is because the proposed model incorporates the LSTM model that can compute results faster in comparison to other techniques. Henceforth, the proposed model is more energy-efficient and significantly better.
3. *Memory Utilization*: The memory utilization provides insight into CPU utilization of the system for execution of the DL model for attack classification. In the current study, numerous techniques were implemented and the corresponding utilization of processing units was analyzed. The results are shown in Figure 8. It can be seen that the average memory utilization for GRU-Model 1 is 25.56%. In comparison, LR (35.56%), RNN (46.58%), and DNN (65.58%) were able to utilize more memory for execution. A similar trend was observed for GRU-Model 2 in which the proposed model can register 35.69% memory in comparison to LR (49.58%), RNN (58.35%), and DNN (75.19%). Based on the results, it can be concluded that the presented model is much better than state-of-the-art techniques for attack classification.
4. *Attack Classification Analysis* Attack classification analysis depicts the performance of the proposed classification of attack types. In the current student, four types of attacks have been classified including Man-in-the-middle attack, DDoS, Query Flood Attack, and SYN DDoS attack. For assessment, different statistical measures have been identified. However, it is important to mention that during the attack classification, only the DL technique is altered while the remaining model is kept identical. The results for statistical analysis are shown in Table 3 (Man-in-the-middle attack), Table 4 (DDoS), Table 5 (Query Flood Attack), and Table 6 (SYN DDoS attack). Table 7 shows the overall generalized confusion matrix. It can be seen that in the current scenario, the proposed model can outperform state-of-the-art models in terms of Recall, Precision, Accuracy, Specificity, F1 Measure, and Log Loss. This is because the presented approach tends to learn and store the recent values in the local memory thereby reducing the loss of data. Moreover, the identification of specific attacks further enhances the overall accuracy of the presented approach.
5. *Client Accuracy Analysis*: Figure 9 depicts the user/IoT nodes distribution for the proposed technique and the comparative method in terms of final accuracy. An increase in the number of high-precision distribution users and a drop in the number of

low-precision users have been observed. More users can benefit from training with the proposed approach since it is more equitable. Federated Learning tends to favor particular users in the training process. In other words, some clients are unwilling or unable to increase their participation in the training process or the accuracy. To achieve a more equitable distribution of accuracy, the proposed strategy takes into account the frequency and precision with which users participate.

6. *Weight Divergence Analysis for Non-Independent and Identically Distributed (Non-IID) and Independent and Identically Distributed (IID) Data*: When moving from IID to non-IID, the weight divergence of all layers rises. As a result, it is hypothesized that there is a correlation between the weight divergence and the data skewness. Weight divergence measures the change in weights between two training procedures with the same weight initialization. Figure 10 shows the results for the divergence of weights based on the data analysis. It can be seen that the proposed model can register minimal divergence for weights during testing. Henceforth, it can be concluded that the proposed model is better and more efficient.

Table 3. Attack Classification Analysis: Man-in-the-Middle Attack.

Algorithms	Recall (%)	Precision (%)	Accuracy (%)	Specificity (%)	F1 Measure (%)	Log Loss (%)
Proposed	96	96	95.65	97.25	97.25	2.15
RNN	92	89	94	92	92	4.78
DNN	95	92	92	87	94.45	4.99
LR	96	95	95	96	95	3.54

Table 4. Attack Classification Analysis: Ping DDoS.

Algorithms	Recall (%)	Precision (%)	Accuracy (%)	Specificity (%)	F1 Measure (%)	Log Loss (%)
Proposed	95	94.23	94.65	95.25	95.25	3.15
RNN	90	88	84	85	88	5.78
DNN	91	93	92.2	89	92.45	4.89
LR	94	93	93	92	94	4.54

Table 5. Attack Classification Analysis: Query Flood Attack.

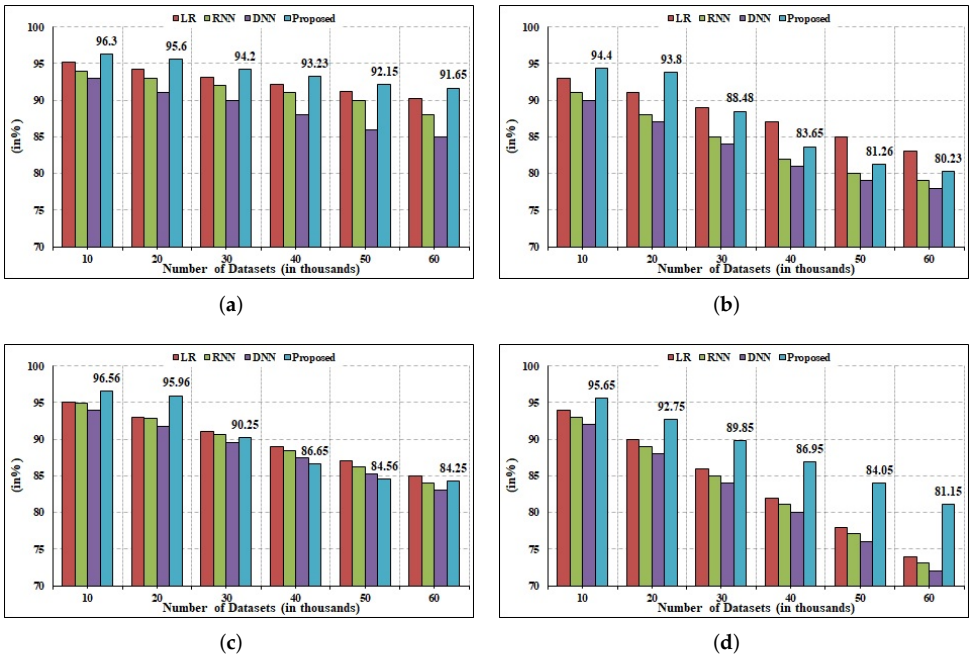
Algorithms	Recall (%)	Precision (%)	Accuracy (%)	Specificity (%)	F1 Measure (%)	Log Loss (%)
Proposed	94.45	94.98	94.35	96.25	96.25	2.67
RNN	92	86	92	91	91	4.34
DNN	92	91	91	83	92.45	5.89
LR	92	92	91	91	92	5.54

Table 6. Attack Classification Analysis: SYN DDoS Attack.

Algorithms	Recall (%)	Precision (%)	Accuracy (%)	Specificity (%)	F1 Measure (%)	Log Loss (%)
Proposed	95	94	93.65	94.25	93.25	2.90
RNN	90	86	90	90	93	5.34
DNN	91	90	87	89	90.45	5.15
LR	92	92	92	93	92	3.98

Table 7. Confusion Matrix.

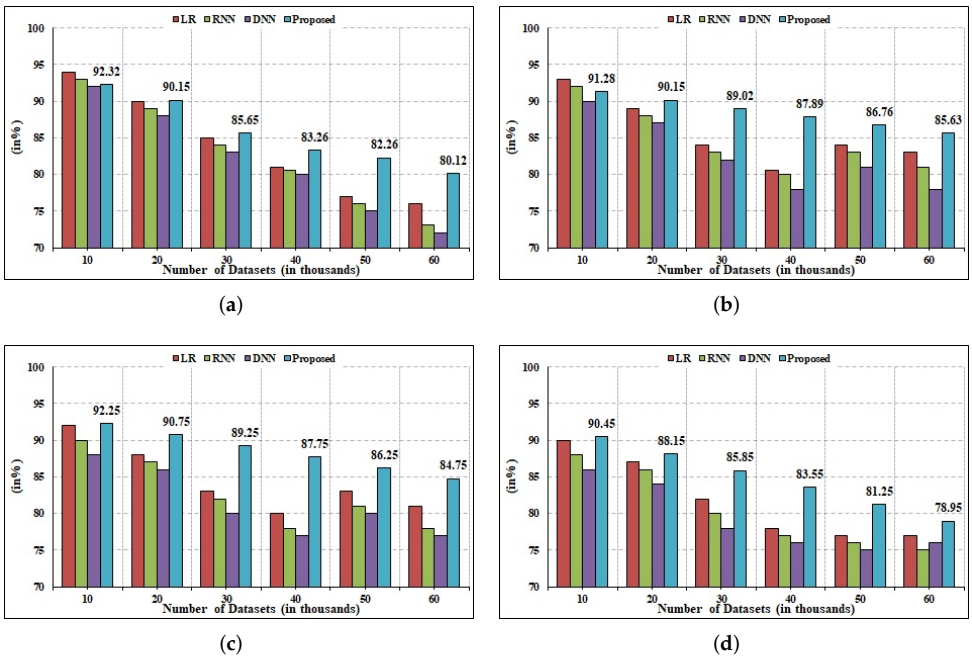
True	0 (No-Attack)	Attack 1	Attack 2	Attack 3	Attack 4
Attack 0 (no-Attack)	50	10	10	30	10
Attack 1	20	1142	10	10	10
Attack 2	10	10	44,541	30	30
Attack 3	10	10	20	55,785	40
Attack 4	10	10	10	145	57,688



(**a**) (**b**)

(**c**) (**d**)

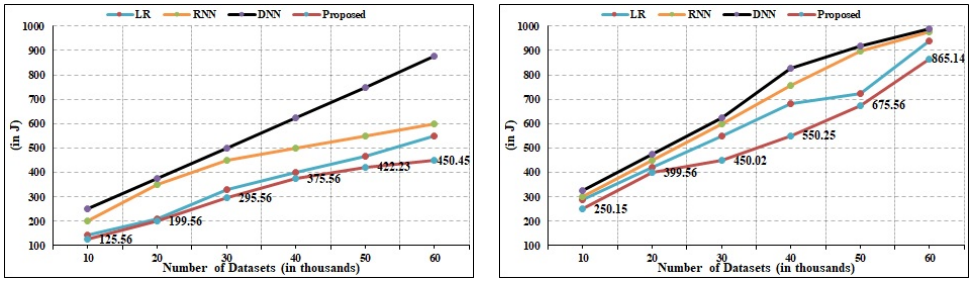
Figure 5. GRU-Model 1: Statistical Analysis. (**a**) Accuracy, (**b**) Precision, (**c**) Recall, (**d**) F1-measure.



(**a**) (**b**)

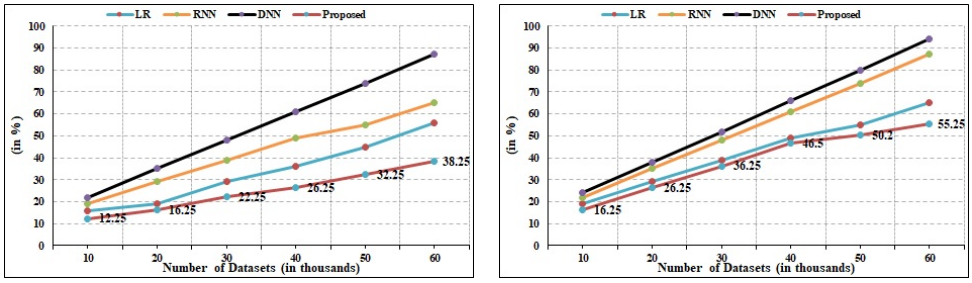
(**c**) (**d**)

Figure 6. GRU-Model 2: Statistical Analysis. (**a**) Accuracy, (**b**) Precision, (**c**) Recall, (**d**) F1-measure.



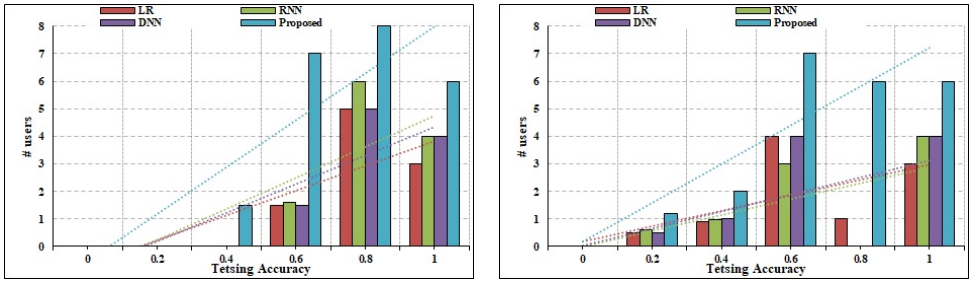
(**a**) (**b**)

Figure 7. Energy Consumption Analysis. (**a**) GRU-Model 1, (**b**) GRU-Model 2.



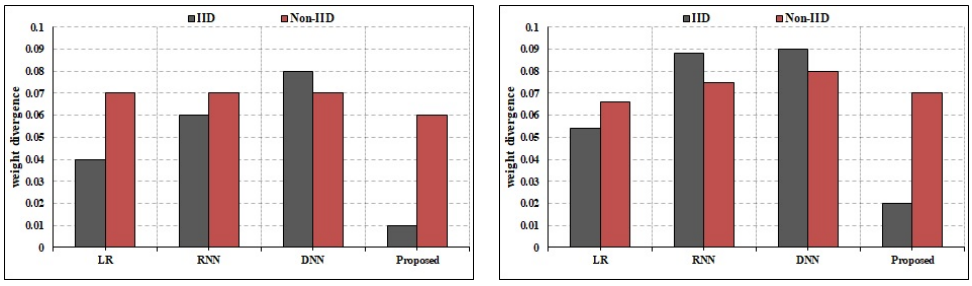
(**a**) (**b**)

Figure 8. Memory Consumption Analysis. (**a**) GRU-Model 1, (**b**) GRU-Model 2.



(**a**) (**b**)

Figure 9. User Testing Analysis. (**a**) GRU-Model 1, (**b**) GRU-Model 2.



(**a**) (**b**)

Figure 10. Weight Divergence Analysis. (**a**) GRU-Model 1, (**b**) GRU-Model 2.

Based on the aforementioned results, it can be concluded that in the current scenario, the proposed technique is more effective and efficient for the classification of IoT attacks.

5. Conclusions

The current research proposes that IoT network attacks can be accurately identified and classified using Federated Learning-based anomaly detection. Specifically, on-device

training and several layers of GRUs enable improved statistical performance rates of accuracy, precision, recall, and F1 measure in categorizing attacks. The ensemble technique, which integrates predictions from various GRU layers, greatly improves the performance. IoT devices become more dependable due to the FL advantages of user data privacy. The evaluation findings show that the suggested intrusion detection method surpasses the state-of-the-art deep learning techniques for attack detection. Using a challenging IoT testbed, the suggested technique can be further improved in the future by using real-time information concerning all known and unknown vulnerabilities in IoT devices.

Author Contributions: Data curation, T.A.A.; Formal analysis, T.A.A. and M.Y.; Funding acquisition, A.A.; Investigation, T.A.A. and A.A.; Methodology, M.A.; Project administration, A.A.; Resources, M.A.; Software, I.U. and M.Y.; Supervision, M.A.; Validation, I.U. All authors have read and agreed to the published version of the manuscript.

Funding: This research work was funded through the project number IF-PSAU-2021/01/17795.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors extend their appreciation to the Deputyship for Research and Innovation, Ministry of Education in Saudi Arabia for funding this research work through the project number IF-PSAU-2021/01/17795.

Conflicts of Interest: The authors declare no conflict of interests.

References

1. Kolias, C.; Kambourakis, G.; Stavrou, A.; Voas, J. DDoS in the IoT: Mirai and other botnets. *Computer* **2017**, *50*, 80–84. [CrossRef]
2. Neshenko, N.; Bou-Harb, E.; Crichigno, J.; Kaddoum, G.; Ghani, N. Demystifying IoT security: An exhaustive survey on IoT vulnerabilities and a first empirical look on Internet-scale IoT exploitations. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 2702–2733. [CrossRef]
3. Zhou, W.; Jia, Y.; Peng, A.; Zhang, Y.; Liu, P. The effect of iot new features on security and privacy: New threats, existing solutions, and challenges yet to be solved. *IEEE Internet Things J.* **2018**, *6*, 1606–1616. [CrossRef]
4. Panchal, A.C.; Khadse, V.M.; Mahalle, P.N. Security issues in IIoT: A comprehensive survey of attacks on IIoT and its countermeasures. In Proceedings of the 2018 IEEE Global Conference on Wireless Computing and Networking (GCWCN), Lonavala, India, 23–24 November 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 124–130.
5. Tushir, B.; Sehgal, H.; Nair, R.; Dezfouli, B.; Liu, Y. The impact of dos attacks onresource-constrained iot devices: A study on the mirai attack. *arXiv* **2021**, arXiv:2104.09041.
6. Waqas, M.; Kumar, K.; Laghari, A.A.; Saeed, U.; Rind, M.M.; Shaikh, A.A.; Hussain, F.; Rai, A.; Qazi, A.Q. Botnet attack detection in Internet of Things devices over cloud environment via machine learning. *Concurr. Comput. Pract. Exp.* **2022**, *34*, e6662. [CrossRef]
7. Thom, J.; Thom, N.; Sengupta, S.; Hand, E. Smart Recon: Network Traffic Fingerprinting for IoT Device Identification. In Proceedings of the 2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 26–29 January 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 72–79.
8. Li, B.; Wu, Y.; Song, J.; Lu, R.; Li, T.; Zhao, L. DeepFed: Federated deep learning for intrusion detection in industrial cyber–physical systems. *IEEE Trans. Ind. Inform.* **2020**, *17*, 5615–5624. [CrossRef]
9. Chen, Z.; Lv, N.; Liu, P.; Fang, Y.; Chen, K.; Pan, W. Intrusion detection for wireless edge networks based on federated learning. *IEEE Access* **2020**, *8*, 217463–217472. [CrossRef]
10. Cetin, B.; Lazar, A.; Kim, J.; Sim, A.; Wu, K. Federated wireless network intrusion detection. In Proceedings of the 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 9–12 December 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 6004–6006.
11. Al-Marri, N.A.A.A.; Ciftler, B.S.; Abdallah, M.M. Federated mimic learning for privacy preserving intrusion detection. In Proceedings of the 2020 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom), Odessa, Ukraine, 26–29 May 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–6.
12. Rahman, S.A.; Tout, H.; Talhi, C.; Mourad, A. Internet of things intrusion detection: Centralized, on-device, or federated learning? *IEEE Netw.* **2020**, *34*, 310–317. [CrossRef]
13. Rouzbahani, H.M.; Dehghantanha, A.; Choo, K.K.R. Big Data Analytics and Forensics: An Overview. In *Handbook of Big Data Analytics and Forensics*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 1–5.
14. Breux, V.; Boutet, J.; Goret, A.; Cattin, V. Anomaly Detection in a Data Center with a Reconstruction Method Using a Multi-Autoencoders Model. *Int. J. Mech. Ind. Eng.* **2022**, *16*, 46–53.

15. Ryffel, T.; Trask, A.; Dahl, M.; Wagner, B.; Mancuso, J.; Rueckert, D.; Passerat-Palmbach, J. A generic framework for privacy preserving deep learning. *arXiv* **2018**, arXiv:1811.04017.
16. Mothukuri, V.; Khare, P.; Parizi, R.M.; Pouriyeh, S.; Dehghantanha, A.; Srivastava, G. Federated learning-based anomaly detection for IoT security attacks. *IEEE Internet Things J.* **2021**, *9*, 2545–2554. [CrossRef]
17. Li, B.; Ma, S.; Deng, R.; Choo, K.K.R.; Yang, J. Federated Anomaly Detection on System Logs for the Internet of Things: A Customizable and Communication-Efficient Approach. *IEEE Trans. Netw. Serv. Manag.* **2022**, *19*, 1705–1716. [CrossRef]
18. Campos, E.M.; Saura, P.F.; González-Vidal, A.; Hernández-Ramos, J.L.; Bernabe, J.B.; Baldini, G.; Skarmeta, A. Evaluating Federated Learning for intrusion detection in Internet of Things: Review and challenges. In *Computer Networks*; Elsevier: Amsterdam, The Netherlands, 2021; p. 108661.
19. Gonzalez-Dominguez, J.; Lopez-Moreno, I.; Sak, H. Automatic language identification using long short-term memory recurrent neural networks. 2014.
20. Sun, T.; Li, D.; Wang, B. Decentralized federated averaging. *arXiv* **2021**, arXiv:2104.11375.
21. Wang, L.; Xu, S.; Wang, X.; Zhu, Q. Addressing class imbalance in federated learning. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual Conference, 2–9 February 2021; Volume 35, pp. 10165–10173.
22. Niu, C.; Wu, F.; Tang, S.; Hua, L.; Jia, R.; Lv, C.; Wu, Z.; Chen, G. Billion-scale federated learning on mobile clients: A submodel design with tunable privacy. In Proceedings of the 26th Annual International Conference on Mobile Computing and Networking, London, UK, 21–25 September 2020; pp. 1–14.
23. Zhou, X.; He, J.; Yang, C. An ensemble learning method based on deep neural network and group decision making. *Knowl.-Based Syst.* **2022**, *239*, 107801. [CrossRef]
24. Ferrag, M.A.; Friha, O.; Maglaras, L.; Janicke, H.; Shu, L. Federated deep learning for cyber security in the internet of things: Concepts, applications, and experimental analysis. *IEEE Access* **2021**, *9*, 138509–138542. [CrossRef]
25. Friha, O.; Ferrag, M.A.; Shu, L.; Maglaras, L.; Choo, K.K.R.; Nafaa, M. FELIDS: Federated learning-based intrusion detection system for agricultural Internet of Things. *J. Parallel Distrib. Comput.* **2022**, *165*, 17–31. [CrossRef]
26. Moustafa, N. *ToN-IoT Datasets*; IEEE: Piscataway, NJ, USA, 2019. [CrossRef]
27. Jaloudi, S. Communication protocols of an industrial internet of things environment: A comparative study. *Future Internet* **2019**, *11*, 66. [CrossRef]