network

Article

A Federated Learning-Based Approach for Improving Intrusion Detection in Industrial Internet of Things Networks

Md Mamunur Rashid [1], Shahriar Usman Khan [2], Fariha Eusufzai [3], Md. Azharuddin Redwan [3], Saifur Rahman Sabuj [3,*] and Mahmoud Elsharief [4]

[1] Department of Artificial Intelligence Convergence, Pukyong National University, Busan 48513, Republic of Korea
[2] Institute of Information Technology, Jahangirnagar University, Dhaka 1342, Bangladesh
[3] Department of Electrical and Electronic Engineering, Brac University, Dhaka 1212, Bangladesh
[4] Department of Electronic Engineering, Hanbat National University, Daejeon 34158, Republic of Korea
* Correspondence: s.r.sabuj@ieee.org

Abstract: The Internet of Things (IoT) is a network of electrical devices that are connected to the Internet wirelessly. This group of devices generates a large amount of data with information about users, which makes the whole system sensitive and prone to malicious attacks eventually. The rapidly growing IoT-connected devices under a centralized ML system could threaten data privacy. The popular centralized machine learning (ML)-assisted approaches are difficult to apply due to their requirement of enormous amounts of data in a central entity. Owing to the growing distribution of data over numerous networks of connected devices, decentralized ML solutions are needed. In this paper, we propose a Federated Learning (FL) method for detecting unwanted intrusions to guarantee the protection of IoT networks. This method ensures privacy and security by federated training of local IoT device data. Local IoT clients share only parameter updates with a central global server, which aggregates them and distributes an improved detection algorithm. After each round of FL training, each of the IoT clients receives an updated model from the global server and trains their local dataset, where IoT devices can keep their own privacy intact while optimizing the overall model. To evaluate the efficiency of the proposed method, we conducted exhaustive experiments on a new dataset named Edge-IIoTset. The performance evaluation demonstrates the reliability and effectiveness of the proposed intrusion detection model by achieving an accuracy (92.49%) close to that offered by the conventional centralized ML models' accuracy (93.92%) using the FL method.

Keywords: federated learning; intrusion detection; Internet of Things; machine learning; neural networks; privacy; security

check for updates

Citation: Rashid, M.M.; Khan, S.U.; Eusufzai, F.; Redwan, M.A.; Sabuj, S.R.; Elsharief, M. A Federated Learning-Based Approach for Improving Intrusion Detection in Industrial Internet of Things Networks. *Network* **2023**, *3*, 158–179. https://doi.org/10.3390/network3010008

Academic Editor: Xinrong Li

Received: 21 December 2022
Revised: 23 January 2023
Accepted: 27 January 2023
Published: 30 January 2023

1. Introduction

The development of the industrial Internet of Things (IIoT) has advanced significantly over the last few years as a result of the rapid development of wireless transmission and processing. A range of cutting-edge portable devices, such as smart phones, smart watches, and smart applications, have emerged on the IoT networks. Numerous businesses, including live gaming, smart manufacturing, navigational systems, smart cities, and smart healthcare, have extensively used these. The architecture of IoT networks still faces a number of important challenges due to their rapid proliferation. The creation of efficient and flexible control for IoT systems that can aid in energy savings, increase the number of applications, and be advantageous for potential future expansion is one of the main challenges. Along with guaranteeing security and privacy against unauthorized access, the IoT networks are cognitively demanding, time-efficient, and have a constant requirement for computing resources, which is another significant barrier. Due to the rapid development of digital technology and the growth in personal awareness, people are beginning to think about personal data security even more [1].

Distributed learning methods are needed so that devices can work together to create a single way to learn with local training. Federated learning (FL) is a decentralized platform for machine learning (ML) [2]. Unlike centralized learning frameworks, the FL framework automatically promotes confidentiality and privacy because data created on an end device does not leave the device. The data from the participating devices is used on the device itself to train the distributed learning model. A client device (e.g., a local Wi-Fi router) and a cloud server only share the settings that have been changed. Some of the benefits of using FL in wireless IoT networks are: (i) instead of exchanging huge amounts of training data, local ML system settings can save power and use less wireless bandwidth; (ii) locally calibrating the parameters of an ML model can greatly reduce transmission delay; (iii) FL can help protect data privacy because only the local learning model variables are sent and the training data stays on the edge devices. As shown in Figure 1, edge devices in FL work together to make a learning model by only sending locally learned designs to a global aggregation server and keeping the local training input at the device end [3].

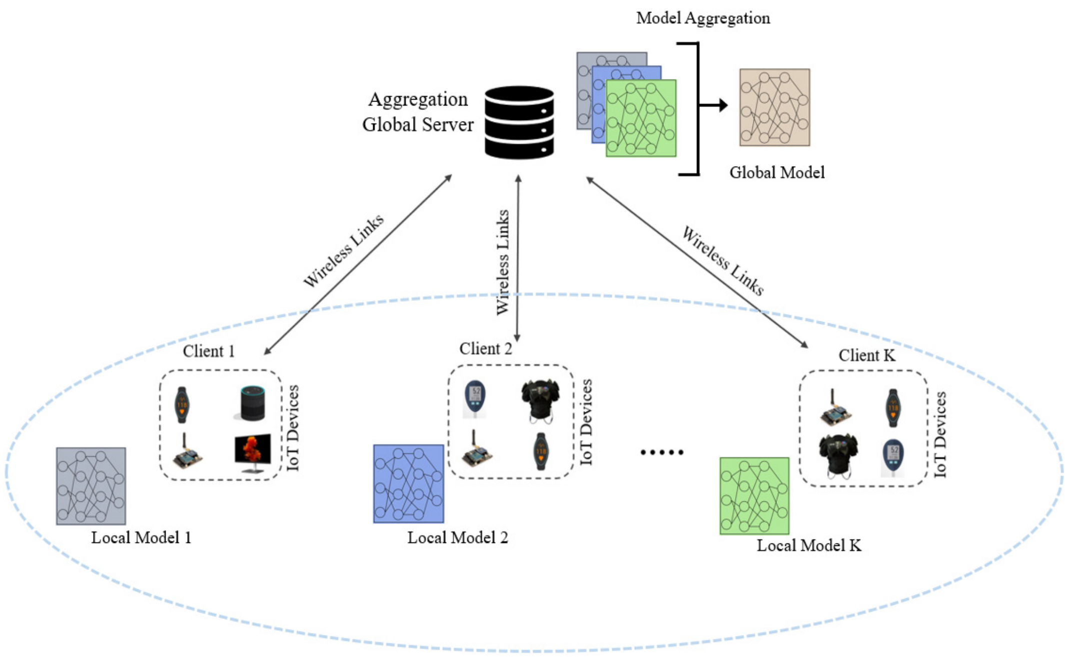

Figure 1. A generalized diagram of FL for wireless IoT devices.

The security of wireless IIoT devices is becoming an increasing concern for both manufacturers and consumers. IIoT devices are susceptible, as they lack the essential built-in security mechanisms to resist intrusions. The confined environment and low computational power of these devices are two of the primary reasons for this. The functions that can be performed on IoT devices are typically constrained by their low power consumption. Security measures subsequently keep failing as a result of that. Moreover, when individuals kept their data online without a password or with weak or default settings, the researchers uncovered one of the most critical security flaws. Frequently, IIoT devices are shipped with default, easy-to-remember passwords or no password at all. By gaining access to these devices, hackers may exploit this vulnerability with relative ease. Due to this vulnerability, attackers may exploit IIoT devices to perform serious assaults, putting the privacy of users at risk [4].

In order to prevent catastrophic harm, the research community is working on designing systems that can respond quickly and effectively to these assaults. An intrusion detection system (IDS) is a specialized security system that continuously examines network or computer system events for signs of an intrusion [5]. It examines the metadata found in network packets and employs pre-established rules to determine whether to allow or deny traffic. Intrusion detection methods can primarily be divided into two types: deployment-based techniques and detection-based techniques. Additionally, each of these categories can be grouped into a further two subcategories. Depending on how they are used, the intrusion detection systems can be of two types: host-based IDS (HIDS) and network-based IDS (NIDS). Depending on how intrusions are detected, IDS techniques can be categorized as either signature-based or anomaly-based systems. The first is established based on predefined behavioral patterns. As a result of that, a new or unknown threat cannot be detected with signature-based IDS. In order to detect any anomaly in the system's behavior and identify it as a potential threat, anomaly-based IDS employ specific network traffic characteristics. Currently, there are few ML-based solutions that are being used to produce a practical IDS to aid IIoT networks in discovering system irregularities. However, typical centralized systems are susceptible to a single point of failure (SPOF) [6], in addition to other limitations. One feasible solution to these issues is FL, which enables devices to train a common ML model without trading or receiving data.

This paper presents a method for accurately identifying unwanted intrusions in wireless IIoT networks in order to protect the privacy and security of IIoT data. We designed and tested an FL-based IIoT network capable of detecting network breaches and increasing security where ML takes place locally on local distributed clients rather than a centralized server. The contributions of our work can be found below:

- We have proposed an FL approach for IIoT intrusion detection in order to train the models at the local device end and accumulate their learning to ensure enhanced security against unwanted attacks.
- We have deployed two deep learning classifiers, namely, convolutional neural network (CNN) and recurrent neural network (RNN) for both centralized ML and federated learning (FL).
- We have utilized the Edge-IIoTset dataset [7] to demonstrate a thorough evaluation that contains real-world data about different types of attacks.
- We compared our proposed method to other works to show that it is superior.

The remainder of the article is organized as follows. Section 2 includes a review of the literature on the most current FL advancements related to IoT and intrusion detection. Section 3 discusses the proposed method in detail. Section 4 presents and thoroughly discusses the experimental data and evaluation. Finally, Section 5 summarizes our findings and suggests many potential future research topics.

2. Related Works

We have highlighted the contributions and limitations of existing related works focused on FL and ML adoption in IoT networks and intrusion detection.

In the most recent state of the art, there are two surveys about IoT intrusion detection [8,9]. Zarpelo et al. [8] gave an overview of IDS that are specific to the IoT and a taxonomy to organize them. Additionally, they presented a thorough comparison of the different IDS for IoT, taking things such as installation strategy, detection mechanism, and validation strategy into account. Benkhelifa et al. [9] were more concerned with enhancing IoT intrusion detection processes. They investigated the current state of the art, with a focus on IoT architecture. It provided a more comprehensive and critical analysis. In another paper, Ahmad et al. [2] presented the use of ML to defend massive IoT devices from various attacks. This work proved ML approaches and their efficacy by evaluating hundreds of research publications.

Samek et al. [10] focused on ML in IoT devices. Modern communication systems generate massive amounts of data, and ML optimizes resource allocation, saves energy, and improves performance. The research studied distributed learning to improve wireless device connectivity. Gunduz et al. discussed denial-of-service (DoS) attacks in wireless sensor networks (WSNs) [11]. DoS attacks can damage any layer of a WSN's architecture, compromising its security. Their study focused on five TCP/IP protocol levels and ML DoS solutions. In addition, they studied ML-based IDSs to solve the issue. Moreover, most conventional solutions lacked an ML-oriented approach; hence, the authors recommended ML

IDS to secure TCP/IP layers. Using a dimension reduction technique and a classifier, Zhao et al. [12] proposed a system that detects anomalies in IoT networks. Dimension reduction was performed with the help of principal component analysis to stop the performance of fault diagnostics from getting worse and to stop real-time demand threats from complex computations. This article is lacking an example of how to calculate memory size.

Vallathan et al. [13] suggest a novel deep learning-based technique for predicting the likelihood of abnormal events utilizing footage from connected surveillance systems and alerting users to those events in an IoT environment. A deep neural network, a multi-classifier, and kernel density functions make up the suggested solution. Here, abnormal behaviors are anticipated using the Random Forest Differential Evolution with Kernel Density (RFKD) method, and any abnormal activities that are identified result in signals being delivered to IoT devices using the MQTT (Message Queuing Telemetry Transport) protocol. The work does not, however, cover the tracking and detection of multiple anomalies in living environments.

Ferrag et al. [14] carried out a study that investigated a deep learning-based IDS for distributed DoS (DDoS) attacks that is built on three models: convolutional neural networks, deep neural networks, and recurrent neural networks. The performance of each model was investigated across two classification types (binary and multiclass) using two new actual traffic datasets, CIC-DDoS2019 and TON_IoT, which include various forms of DDoS attacks. The intrusion detection approach presented by Pajouh et al. [15] is built on a two-layer dimension reduction and two-tier classification module. This method is also reported to be able to identify User-to-Root (U2R) and Remote-to-Local (R2L) cyberattacks. Both linear discriminant analysis and component analysis have been used to minimize the number of dimensions. The Network Security Laboratory-Knowledge Discovery Dataset (NSL-KDD) dataset is used throughout the entire experiment. The two-tier classification module employs the Naive Bayes and certainty factor versions of KNN to identify anomalous behavior. An intrusion detection approach based on a two-step classification system was proposed by Pamukov et al. [16]. The first stage was to employ a negative selection approach that was resistant to difficult classification problems. The attack samples were then classified using a trained neural network. So, this strategy saved what little power and computing resources the end devices had by getting rid of the overhead of the training process. This work is limited to the creation of the Negative Selection Neural Network (NSNN) algorithm, and currently, there is no best way to implement online learning for it.

Khan et al. [17] focused on the application of decentralized ML technology to handle the massive amounts of data from the increasing IoT devices and subsequently discussed the challenges of FL. Their work developed a Stackelberg game-based methodology in order to create an FL incentive mechanism to improve the interaction between devices and edge servers. For their experiment, they utilized the Modified CoCoA framework and the MNIST dataset. The authors claimed that this approach was useful for big IoT networks since it enables customization of different ML computing characteristics based on the capabilities of connected devices. Tang et al. [18] suggested an FL-based approach to network intrusion detection. It supposedly addresses the issues of an inadequate network intrusion detection data set and privacy protection. For iterative training, this technique runs the GRU deep learning model locally. Network traffic data is stored locally, and the central server aggregates and averages the parameters, as is the case with all federated learning techniques. They employed the CICIDS2017 intrusion detection data set for their experiment, which is a popular intrusion detection data set but just a lab simulation data set.

Chen et al. [19] discussed the importance of distributed learning and the integration of FL in large IoT networks. However, achieving ML accuracy with the FL technique in a large IoT network requires regular updates to global algorithms, which cost a significant amount of data. They investigated the problem of maximizing resources and learning performance in wireless FL systems at the same time in order to reduce communication costs and improve learning performance. A Lagrange multiplier method is first used by decoupling variables, such as power variables, bandwidth variables, and transmission indicators, in order to maximize effective information flow via networks. Then, a power and bandwidth allocation mechanism based on linear search is established. In their work, Cao et al. [20] said that it was a major challenge for local differential privacy (LDP) in power IoTs to show how to find a balance between utility and privacy while still letting the native IoT terminal run. It was suggested to use an optimized framework that considered the trade-off between local differential privacy, data utility, and resource utilization. Additionally, users were divided into groups according to their level of requirement for privacy, and sensitive users received better privacy protection. The authors used Sparse Coding Randomized Aggregable Privacy-Preserving Ordinal Response (SCRAPPOR) and Factorial Hidden Markov Model (FHMM) algorithms and evaluated the Reference Energy Disaggregation Dataset (REDD) in the proposed method. This research was limited to LDP and power IoTs.

Attota et al. [21] offered the MV-FLID FL-based intrusion detection method, which trains on various IoT network data perspectives in a decentralized manner to identify, classify, and prevent intrusions. Maximizing the learning effectiveness of various kinds of attacks is facilitated by the multi-view ensemble learning component. The FL feature efficiently aggregates profiles through the use of peer learning, as the device's data is not shared with the server. However, they did not explore unsupervised and reinforcement ML systems, which can improve intrusion detection by detecting untrained attacks, in their work. In order to identify cyber threats in smart Internet of Things (IoT) systems, including smart homes, smart e-healthcare systems, and smart cities, Tabassum et al. [22] suggested a federated deep learning (DL) intrusion detection system utilizing GAN, named FEDGAN-IDS. In order to train the GAN network using augmented local data and serve as a classifier, they distributed it among IoT devices. They demonstrated that their model performed better and converged earlier than the most standalone IDS by comparing the model's convergence and accuracy.

Driss et al. [23] described a framework based on FL for detecting cyberattacks in vehicular sensor networks (VSNs). The proposed FL approach makes it possible to share computing resources and train with devices. For better performance in attack detection, a Random Forest (RF)-based ensemble unit is used in the suggested approach in conjunction with a group of Gated Recurrent Units (GRU). Du et al. [24] highlighted that vehicle IoT devices feature sensors that produce device-specific information that can affect device security if leaked. GPS, cameras, radar, etc., must be shared in cooperative driving. The authors proposed using FL to improve system security and performance by integrating massive IoT networks with various devices. Ghourabi et al. [25] proposed a new intrusion and virus detection system to secure the healthcare system's whole network. The proposed approach consists of two parts: an IDS for medical devices installed on the healthcare network and a malware detection system for data servers and medical staff devices. The goal was to protect the entire network, regardless of the devices and computers installed. The limitation is that it demands the installation of numerous systems across the healthcare network. Additionally, a correlation procedure is needed to compile the outcomes.

We have summarized the key contributions of the similar works in Table 1. As can be seen, we have included the publication year, experiment dataset, used classifiers/algorithms and key findings in the table.

Table 1. Summary of the related works.

Year	Reference	Dataset	Classifier/Algorithm/Framework	Findings
2017	[12]	KDD-CUP99	Softmax Regression, KNN	This paper proposed a system that detects anomalies in IoT networks. Dimension reduction was performed with the help of principal component analysis to prevent the performance of fault diagnostics from declining and to stop real-time demand threats from complex computations. This article lacked examples of how to calculate the memory size.

Table 1. *Cont.*

Year	Reference	Dataset	Classifier/Algorithm/Framework	Findings
2021	[13]	HHAR	RFKD	In this work, abnormal behaviors are predicted using the RFKD method, and any abnormal activities that are identified result in signals being delivered to IoT devices using the MQTT protocol. The work does not, however, cover the tracking and detection of multiple anomalies in living environments.
2021	[14]	CIC-DDoS2019, TON_IoT	CNN, DNN, RNN	This deep learning-based IDS was proposed for cybersecurity in agriculture 4.0 and identified DDoS attacks using three ML models: CNN, DNN, and RNN. The performance of each model was investigated across two classification types (binary and multiclass) using two real traffic datasets, CIC-DDoS2019 and TON_IoT, which include various forms of DDoS attacks.
2020	[15]	NSL-KDD	Naive Bayes, KNN	This proposed IDS for anomaly detection in IoT networks is based on a two-step classification system. This method is capable of identifying cyberattacks from U2R and R2L. The two-tier classification module employs the Naive Bayes and certainty factor versions of KNN to identify anomalous behavior.
2018	[16]	NSL-KDD	NSNN	An intrusion detection approach based on a two-step classification system was proposed in this paper using a negative selection algorithm and neural network. This strategy saved what little power and computing resources the end devices had by getting rid of the overhead of the training process. This work is limited to the creation of the NSNN algorithm, and currently, there is no best way to implement online learning for it.
2020	[17]	MNIST	Modified CoCoA framework	This research was focused on the application of FL technology for resource optimization and incentive mechanisms in edge networks. A Stackelberg game-based methodology was proposed in order to create an FL incentive mechanism to improve the interaction between devices and edge servers. This approach is useful for big IoT networks since it enables customization of different ML computing characteristics based on the capabilities of connected devices.
2022	[18]	CICIDS2017	GRU	This FL-based IDS supposedly addresses the issues of an inadequate network intrusion detection data set and privacy protection. This technique makes it possible for numerous ISPs or other organizations to carry out joint deep learning training under the premise of preserving local data. The privacy protection of the network traffic can be solved using this method, but no real-world scenario simulation was conducted to prove its feasibility.
2022	[19]	MNIST	CNN	This work investigated the problem of optimizing resources and learning performance in wireless FL systems at the same time in order to reduce communication costs and improve learning performance. A Lagrange multiplier method is used by decoupling variables, such as power variables, bandwidth variables, and transmission indicators, in order to maximize effective information flow via networks. A power and bandwidth allocation mechanism based on linear search is also established. The framework can successfully schedule clients based on wireless channel dynamics and learned model parameter features.
2020	[20]	REDD	SCRAPPOR, FHMM	This paper proposed an optimized framework that considered the trade-off between LDP, data utility, and resource utilization in power IoTs. Additionally, users were divided into groups according to their level of requirement for privacy, and sensitive users received better privacy protection. This work is limited to LDP and power IoTs.
2021	[21]	MQTT	RF	Maximizing the learning effectiveness of various kinds of attacks is facilitated by the multi-view ensemble learning component in this work. The FL feature efficiently aggregates profiles through the use of peer learning, as the device's data is not shared with the server. However, they did not explore unsupervised and reinforcement ML systems, which can improve intrusion detection by detecting untrained attacks.

Table 1. *Cont.*

Year	Reference	Dataset	Classifier/Algorithm/Framework	Findings
2022	[22]	NSL-KDD, KDD-CUP99, and UNSW-NB15	GAN	In order to identify cyber threats in smart IoT systems, including smart homes, smart e-healthcare systems, and smart cities, this federated DL intrusion detection system was proposed. In order to train the GAN network using augmented local data and serve as a classifier, they distributed it among IoT devices. By comparing the convergence and accuracy of the model, they showed that their model worked better and converged faster than most standalone IDS.
2022	[23]	Car Hacking: Attack & Defense Challenge 2020 Dataset	GRU with a RF-based ensemble unit	This approach makes it possible to share computing resources and train using vehicular sensor devices. An RF-based ensemble unit and a group of GRU are used in the suggested method to improve the performance of attack detection.
2022	[25]	TON_IoT, Edge_IIoTset, EMBER, and ECU-IoHT	LightGBM, BERT-based Transformer, and BiLSTM	This method consists of two parts: an IDS for medical devices installed on the healthcare network and a malware detection system for data servers and medical staff devices. The goal was to protect the entire network, regardless of the devices and computers installed. The limitation is that it demands the installation of numerous systems across the healthcare network. Additionally, a correlation procedure is needed to compile the outcomes.
2023	This paper	Edge_IIoTset	CNN, RNN, FedAvg Algorithm	We have proposed an FL approach for IIoT intrusion detection in order to train the models at the local device end and accumulate their learning to ensure enhanced security against unwanted attacks. We have deployed two deep learning classifiers, namely, CNN and RNN, for both centralized ML and FL. We have utilized the Edge-IIoTset dataset [7] (which contains real-world data about different types of attacks) to demonstrate a thorough evaluation and our method's performance.

In this paper, we propose a methodology for intrusion detection in IIoT networks in which data is securely stored at the end devices (often a collection of IIoT devices) and aggregated learnings are transmitted to the central server. We also conducted experiments and evaluations of our methodology using both centralized and FL on the Edge-IIoTset dataset, where our proposed FL model performed admirably, allowing us to confidently state that our method can accurately identify unwanted intrusions in IIoT networks.

3. Proposed Method

In conventional settings for anomaly detection, training data from the objects to be modeled is utilized to construct the model. However, the IIoT environment creates challenges for this method. As IIoT devices are primarily single-function devices with limited capabilities, they do not generate massive volumes of data. This makes it difficult to train a model using only data collected from a user's local network, as it may take some time to collect sufficient data for training a consistent model. This needs a method that combines training data from several users into a single local training dataset, thus accelerating the learning of a stable model. FL clients are those end devices (e.g., a local Wi-Fi router) that collect data from their respective connected IIoT devices. In a FL environment, each FL client trains a local model utilizing locally accessible training data, as opposed to transmitting data to a central entity as in a fully centralized learning architecture. After that, the learnings from those local trainings are transmitted back to the global server for aggregation. At the global server, the model is again improved by global training and distributed back to the local FL clients for the next FL iteration. By this way, both the global and local models get improved and can effectively classify the intrusions and benign traffic inside an IIoT dataset. This section describes and explains the operation of our proposed approach in detail.

3.1. System Architecture

The layout of the suggested FL technique for IIoT intrusion detection is shown in Figure 2, where a number of devices are installed in various locations and connected to the network. Our proposed model is divided into three parts:

- Local-end Learnings and Intelligence
- Learnings Distribution
- Accumulated Global Learnings

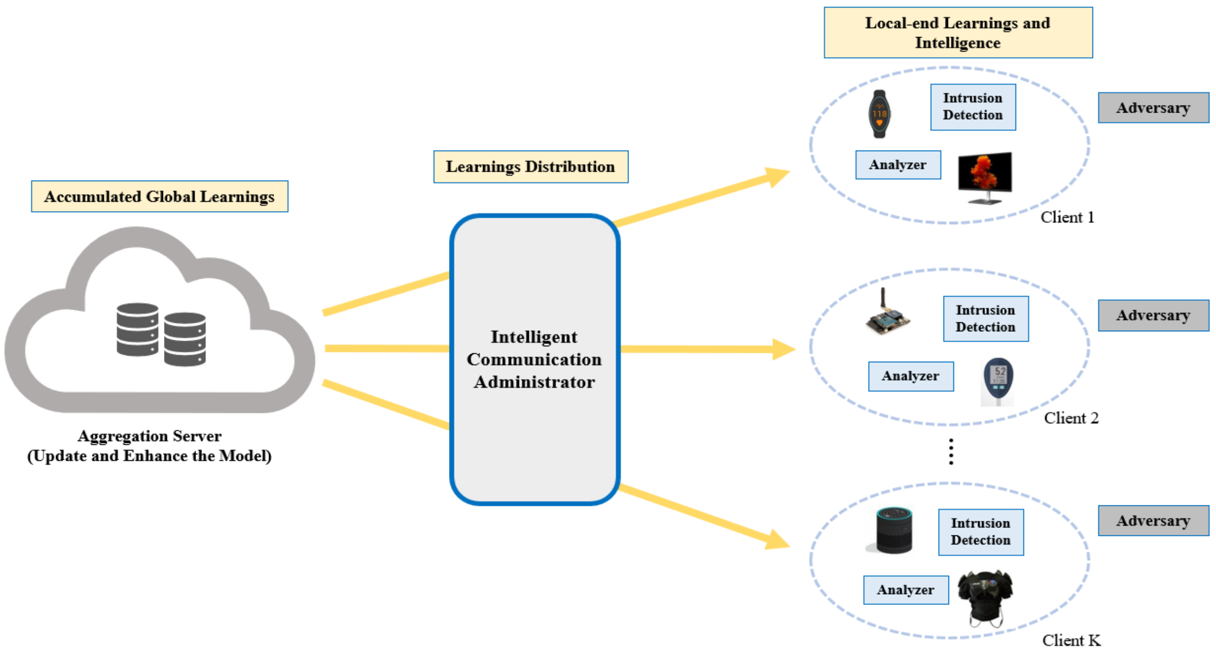

Figure 2. Proposed FL architecture for IIoT intrusion detection.

3.1.1. Local-End Learnings and Intelligence

In this part of the framework, each k client ($k \in [1, \dots, K]$) at the local end trains the data acquired from their separate IIoT devices with the local models shared by the server, while the IDS at the client end detects any unwanted attacks. Additionally, an analyzer is employed to keep track of their network data for subsequent analysis. This kind of smart learning on the device protects the independence of local intrusion detection by requiring local training, tweaking of parameters, and improving inference procedures.

3.1.2. Learnings Distribution

With the aim of integrating the models and developing a better intrusion detection system with optimum parameters, the clients exchange their trained learning with a server-based system for aggregation. The intelligent communication administrator (e.g., security gateway) is in charge of all interactions between clients and the aggregation server.

3.1.3. Accumulated Global Learnings

In order to obtain the efficiency of centralized ML methods, which somewhat contain global data learning, the detection models are exchanged with the server-based aggregation platform. The aggregation server is in charge of aggregating local learning and transforming it into global learning. The distributed clients receive the optimized model through the communication platform, which facilitates knowledge sharing. A client can detect intrusions using comparable behaviors obtained from several participating devices thanks to this sharing mechanism, which gradually improves learning.

3.2. Adversary Model and Assumptions

In an IIoT network, a threat or adversary M might be internal or external. An external adversary generally uses the Internet to launch cyberattacks, such as abusing digitally equipped systems, injecting malicious content into databases, stealing confidential information, and so on, or a member of an insider group who could remain inside the network, such as a compromised IIoT device or another networked device. IIoT malware can locate and exploit weak IIoT systems and devices by manipulating devices with lower security as a platform for cyberattacks. In our investigation, we made a few other assumptions. They are as follows:

- Trustworthy FL Aggregator: As aggregation servers are an essential part of the learning process, it is necessary that there always be some level of trust in the system that coordinates learning.
- No Malicious IIoT Device by Design: In some cases, security problems may already be present in a newly released IIoT product. These devices, however, must not be tainted or infected before they are put to their intended purpose. As a result, devices will only generate allowed interactions until an adversary M identifies and exploits any vulnerabilities, providing our model with valuable data from which to learn.
- Secure Clients: Since clients are essential components of FL learning in IIoT systems, we assume that they are secure. If such clients are compromised, the IIoT infrastructure is no longer safeguarded.

3.3. FL for Intrusion Detection

As previously stated, in our FL Model, all K clients' train a local model with the same shared global model, but it is trained on various local datasets rather than on a central server. Following that, they communicate the learning from these local trainings to an aggregation server via an SSL/TSL authenticated connection via the communication administrator (e.g., gRPC channel [26]). The aggregation server integrates all of them and produces an updated global model with optimal parameters. The letter w stands for the starting weights, and R represents the number of FL rounds, which will be repeated before reaching a convergence level. When each local client's weight is submitted to the aggregation server during communication round t, the following equation (Equation (1)) adapted from the FedAvg algorithm [27] can be used to update the model weights.

$$W_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^{k} \tag{1}$$

where n is the total size of all client datasets, and n_k represents the size of each client dataset. W_{t+1} is the updated global model after the iteration.

Figure 3 depicts a scenario illustrating the interconnections between the various FL IIoT intrusion detection system participants. The server initially selects clients who have connectivity with active IIoT devices that are powered on, charging, and connected to an unmetered Wi-Fi network to take part in the FL process. After that, the different parts of the system interact in the following ways to finish the whole process:

1. At t = 0, the server generates a neural network model from a global intrusion detection model. At this point, the number of hidden layers, neurons, epochs, etc. is counted. The symbol w denotes the model's initial weight.
2. Every k client (k ∈ [1, . . . , K]) need to utilize the global model download it, regardless of whether they contribute to the FL process or not. With their own private data, each of the K clients retrains the global model locally in parallel and creates a fresh set of local weights w_{t+1}^{k}.
3. The designated clients use the data collected from the IIoT devices under their control to improve the model under investigation while maintaining the privacy of their local data.

4. To protect the privacy of the clients, only the updated model parameters for the improved intrusion detection model are sent to the central server.
5. Once all the changes are received, the server combines the weights from the various node models to produce a new improved model (Equation (1)). The FedAvg method is used for the aggregation. In this method, the parameters are evaluated based on the size of the dataset at each node.
6. The updated model parameters are pushed back to the clients by the central server.
7. Each client uses the new model parameters and makes changes to them based on the new data.
8. For continuing model learning and improvement, steps 4, 5, 6, and 7 are repeated.

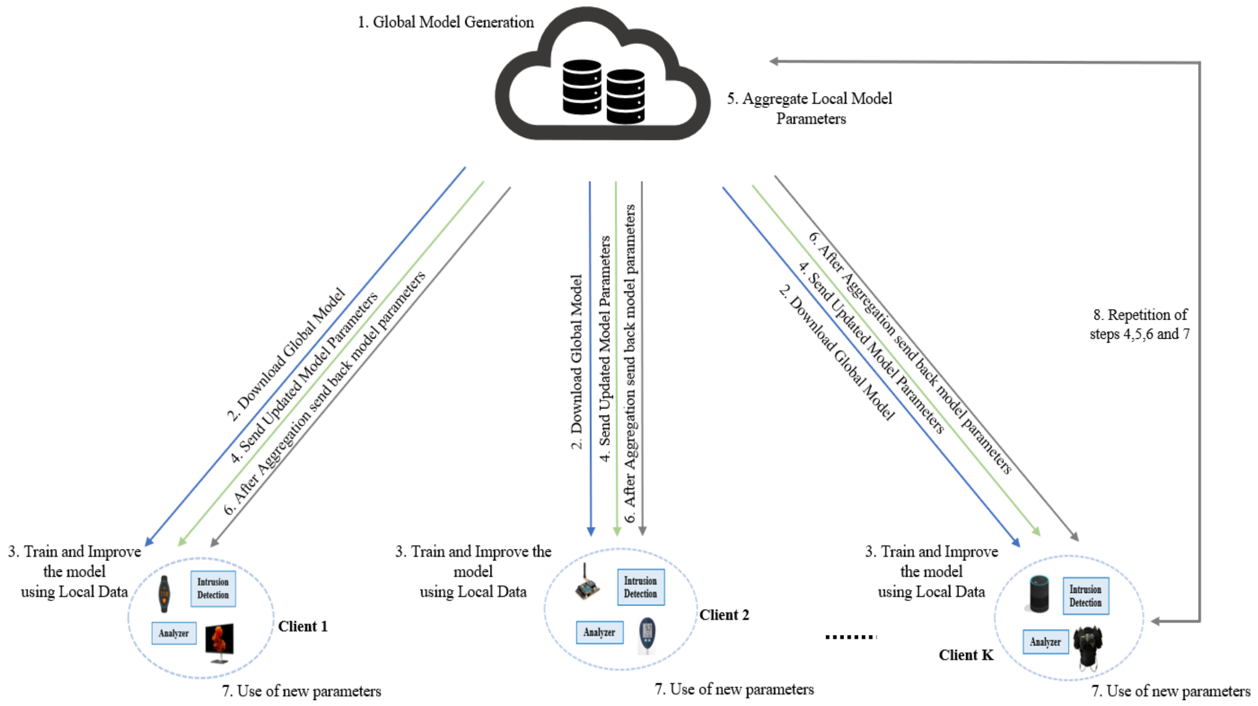
1. Global Model Generation
5. Aggregate Local Model Parameters
8. Repetition of steps 4,5,6 and 7
2. Download Global Model
4. Send Updated Model Parameters
6. After Aggregation send back model parameters
3. Train and Improve the model using Local Data
Intrusion Detection
Analyzer
Client 1
Client 2
Client K
7. Use of new parameters

Figure 3. Interactions among clients' participants of FL-based IDS.

3.4. Complexity of FL Approach

The FL parameters add complexity to FL over a conventional ML system because of the distinct scenario they represent. The skewness of the heterogeneous data spread among the client devices, for instance, is a crucial parameter. In a multi-class classification issue, a severely skewed data scenario can involve each client having just one label of data. The number of participating clients, as well as each client's communication and processing budget, are further important FL variables.

Provided that FL is supposed to function in the edge layer, it should have a low time complexity, as the software being run has a significant impact on the behavior of the associated processes. The time complexity of the global model depends on the time complexity of the clients, the time complexity of the aggregation server, and the complexity of the exchanged parameters, excluding transmission times, as these factors typically vary significantly across networks.

The local client's time complexity can be defined by: $O(E.n_k.(l_0.l_1 + l_1.l_2 + \ldots + l_{L-1}.l_L))$, where l_x represents layer x, E represents the number of local epochs, and n_k represents the size of each client dataset. Now, the global server time complexity depends on the total number of clients (K) and also the cumulation of all local model parameters (W). So, we can define the time complexity of the global server by: $O(K.W)$. Again, we will have many parameters exchanged between global and local clients. The complexity of the exchanged parameters we can define as: $O(W)$.

So, the total time complexity ($O_{(\text{total})}$) of our proposed FL approach can be represented by the following equation (Equation (2)).

$$O_{(\text{total})} = O(K.(E.n_k.(l_0.l_1 + l_1.l_2 + \ldots + l_{L-1}.l_L))) + O(K.W) + O(W) \tag{2}$$

3.5. ML Classifiers for Intrusion Detection

The intelligent IDS solution now has an entirely revolutionary path for growth thanks to the rapid advancements in ML techniques and applications. Neural network methods have proved to be very useful in extracting improved data representations for building effective models. While neural networks come in a variety of types, they all share these common basic components: neurons, weights, biases, and functions. In addition, neural networks frequently serve the same goal of connecting an input x and an output y so that y = f(x,θ), where θ is the parameters vector. For intrusion detection, we have kept the number of classifiers in a centralized setting very limited, since each classifier trains on the full dataset. We have used two different types of classifiers: the recurrent neural network (RNN) and the convolutional neural network (CNN).

3.5.1. Convolutional Neural Network

The purpose of CNN is to interpret data in the form of multiple arrays. The initial layers in this method consist of a collection of learnable filters subjected to convolutional feature extractors. Every bit of input data is traversed by a sliding window created by the applied filters. The outputs are referred to as "feature maps", and the overlapping distance is termed the "stride". A CNN layer that is employed to create distinct feature maps, is composed of convolutional kernels. In the feature map of the subsequent layer, a neuron is related to neighboring neuron areas. The kernel needs to be shared among all input spatial locations in order to produce a feature map. One or more fully connected layers are used to complete the classification after the convolutional and pooling layers are constructed [28]. In CNN architectures, the convolutional operation over the input feature maps and convolutional layers is shown by the following equation (Equation (3)):

$$h_j^{(n)} = \sum_{k=1}^{K} h_k^{(n-1)} * w_{kj}^{(n)} + b_{kj}^{(n)} \tag{3}$$

where $*$ is the convolution operator, and $h_j^{(n)}$ represents the jth feature map in the nth hidden layer. The kth channel of the $(n-1)$th hidden layer is denoted by the letters $h_k^{(n-1)}$, while the weights of the kth channel in the jth filter of the nth layer are represented by the letters $w_{kj}^{(n)}$, and the pertinent bias term is denoted by the letters $b_{kj}^{(n)}$.

3.5.2. Recurrent Neural Network

RNNs are improved feed-forward neural network models that have the capacity to memorize data at each step for subsequent outputs. In an RNN, neurons' output is linked to their own and other neurons' input. RNNs can therefore use their internal memory to represent data sequences and time series [29]. The following is a formalization of the standard RNN at time t:

$$\begin{cases} h_t = g\left(x_t W^{(\mathrm{in},h\mathrm{i})} + h_{t-1} W^{(hi,h\mathrm{i})} + b^{(h)}\right) \\ y_t = g\left(h_t W^{(hi,ou)} + b^{(y)}\right) \end{cases} \tag{4}$$

where x_t is the input sequence. The weight matrices for the input layer to the hidden layer, the hidden layer to the hidden layer, and the hidden layer to the output layer are denoted by $W^{(\mathrm{in},h\mathrm{i})}$, $W^{(hi,h\mathrm{i})}$, and $W^{(hi,ou)}$, respectively. In addition, $g(\cdot)$ denotes the activation function, and b stands for the bias.

The long short-term memory (LSTM) is a special type of RNN model for our model evaluation. The addition of the cell state to the LSTM network makes a difference compared to the regular RNN. The cell state will be sent to the subsequent cell with time t, which stands for long-term memory. The cell individually chooses whether or not to forget throughout the calculation of the cell sequence, allowing it to retain past information for a considerable amount of time [30]. The LSTM hidden layer is developed using the following formula (Equation (5)) at time step t:

$$\begin{cases} i_t = \sigma\left(x_t W^{(x,i)} + h_{t-1}W^{(h,i)} + c_{t-1}W^{(c,i)} + b^{(i)}\right) \\ f_t = \sigma\left(x_t W^{(x,f)} + h_{t-1}W^{(h,f)} + c_{t-1}W^{(c,f)} + b^{(f)}\right) \\ c_t = f_t * c_{t-1} + i_t * \tanh\left(x_t W^{(x,c)} + h_{t-1}W^{(h,c)} + b^{(c)}\right) \\ o_t = \sigma\left(x_t W^{(x,o)} + h_{t-1}W^{(h,o)} + c_t W^{(c,o)} + b^{(o)}\right) \\ h_t = o_t * \tanh(c_t) \\ y_t = \mathrm{SoftMax}(Wh_t + b) \end{cases} \tag{5}$$

where at time step t, i_t stands for the input gates, f_t stands for the forget gates, c_t stands for the memory cells, o_t stands for the hidden output, and y_t stands for the output layer.

4. Experiments, Results, and Discussion

This section includes experimental details for our proposed technique as well as a detailed discussion of performance.

4.1. Experimental Setup

We performed our experiments on Google Colaboratory using Python 3 as the programming language. In order to implement our approach, NumPy and other well-known libraries are employed, along with multi-dimensional arrays and matrices. Pandas offers powerful data-structure manipulation and analysis tools, which we also utilized. In addition, TensorFlow and Keras are used for ML and DL. Moreover, numerous supervised and unsupervised ML method implementations are available in Scikit-learn. Furthermore, we used SMOTE [31] to oversample minority classes in order to increase the overall model efficiency.

4.2. Dataset, Data Pre-Processing, and Feature Selection

Datasets are necessary in IIoT networks for both training and testing IDSs, so the selection of the appropriate dataset is crucial. There is now a new cybersecurity dataset called Edge-IIoTset [7] that was created for IIoT and IoT applications. Numerous IoT devices, such as heart rate sensors, flame sensors, temperature and humidity sensors, etc., generate the data. The testbed was subjected to 14 different types of attacks, including injection, malware, DDoS attacks, and man-in-the-middle (MITM) attacks. For FL-based tasks, the data distribution needs to be non-independently identically distributed (Non-IID), imbalanced, and reflect the elements of the real-world scenario. For the experimental purpose, we have divided our dataset (Edge-IIoTset) into several local datasets to train them as per requirement for FL. This was necessary because FL-specific datasets were unavailable.

We begin by grouping the data and removing duplicates and missing elements such as "NAN" (Not a Number) and "INF" (Infinite Value). Following that, we eliminate additional flow characteristics such as tcp.payload, tcp.options, tcp.len, tcp.flags.ack, tcp.flags, tcp.connection.rst, tcp.connection.fin, tcp.checksum, tcp.ack_raw, tcp.ack, arp.dst.proto_ipv4, ip.dst_host, ip.src_host, and frame.time. In the second step, we performed the data encoding step by using dummy encoding categorical data and by normalizing numeric data using the Z-score normalization defined by $(x - \mu)/\sigma$, where the feature value is x, the mean is μ, and the standard deviation is σ. Then, the oversampling was performed using the SMOTE.

We have applied the feature selection strategy to enhance the proposed model's performance and reduce its training and classification times. The feature selection approach looks for the most pertinent features and eliminates the irrelevant ones. Recursive Feature Elimination (RFE) has been applied in the suggested model. It is a wrapper method that repeatedly assesses how well a certain model performs with various feature combinations. In order to enhance accuracy, RFE recursively removes features from each feature set. The features are then ranked according to the order in which they were eliminated.

Table 2 displays the randomly selected data for ML models after data pre-processing (cleaning and splitting) and feature selection, as well as the train and test data sets that are generated.

Table 2. Data Distribution for Training and Testing.

Type	Total	Train	Test
Normal	24,301	19,368	4933
DDoS_UDP Attack	14,498	11,574	2924
DDoS_ICMP Attack	14,090	11,179	2911
Ransomware Attack	10,925	8634	2291
DDoS_HTTP Attack	10,561	8381	2180
SQL_injection Attack	10,311	8343	1968
Uploading Attack	10,269	8162	2107
DDoS_TCP Attack	10,247	8045	2202
Backdoor Attack	10,195	7967	2228
Vulnerability_scanner Attack	10,076	8089	1987
Port_Scanning Attack	10,071	8011	2060
XSS Attack	10,052	8017	2035
Password Attack	9989	7956	2033
MITM Attack	1214	976	238
Fingerprinting Attack	1001	798	203

We performed a series of tests to evaluate the efficacy of FL with varying client numbers (from 3 to 15) contributing to model training. We trained our model for a total of 50 epochs before reaching optimal output. We investigated the system's efficiency for various client numbers while developing the federated model. Each client received a piece of training data from the deployment dataset that was chosen at random. To examine this potential loss in accuracy, we generated three federated models by sharing the entire training dataset among 3, 9, and 15 clients, and we compared them to a centralized model.

4.3. Performance Metric

The following detection metrics were considered during model evaluation using test data:

- True Positive (TP): It refers to the number of attack samples out of the total number of samples that were accurately detected as attacks.
- False Positive (FP): It refers to the number of normal samples that were incorrectly identified as attacks.
- True Negative (TN): It refers to the number of benign samples that were accurately recognized as normal.
- False Negative (FN): It refers to the number of attack samples that were wrongly classified as normal.
- **Accuracy**: It represents the ratio of the number of correct classifications to the total number of inputs and is calculated as follows:

$$\mathrm{Acc} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \tag{6}$$

- **Precision**: It gives the ratio of successfully classifying attacks to the total number of expected attack outcomes, which can be calculated as follows:

$$\text{Pre} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{7}$$

- **Recall**: It offers the ratio of accurately categorized attacks to all anticipated attack outcomes, which is determined as follows:

$$\text{Rec} = \frac{\text{TP}}{\text{FP} + \text{FN}} \tag{8}$$

- **F_1-Score**: It gives the ratio of successfully classifying attacks to the total number of expected attack outcomes, which can be calculated as follows:

$$\text{F}_1 = 2 \cdot \frac{\text{Precision}\cdot\text{Recall}}{\text{Precision} + \text{Recall}} \tag{9}$$

- **True Positive Rate (TPR)**: A true positive state occurs when an activity is detected by the IDS as an attack and the activity truly represents an intrusion. TPR is sometimes also regarded as the detection rate and can be represented by the following expression:

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{10}$$

- **False Positive Rate (FPR)**: When the IDS detect an activity as an attack when it is actually benign or normal, the state is known as a false positive rate. A false positive could be regarded as a false alarm rate too, which can be calculated as follows:

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{11}$$

4.4. Performance Evaluation

This part contains the findings of the experiment based on centralized learning and our proposed FL-based model performance in intrusion detection using the Edge-IIoTset dataset.

4.4.1. Intrusion Detection Using Centralized Methods

To evaluate the effectiveness of the proposed model, we first applied two traditional centralized ML approaches for cyber-attack detection, namely, CNN and RNN.

The values that were used for the various classifiers' parameters in our suggested approach are listed in Table 3.

Table 4 shows the results of ML approaches for a centralized model in terms of Accuracy, Precision, Recall, and F1-score, which indicates how well the model detects benign and attack classes from the dataset. As shown in the table, the Accuracy and F1-Score values can reach as high as 94% and 93% for RNN and CNN approaches, while the Precision and Recall values can reach as high as 95% (RNN) and 94% (CNN) for identifying benign and malicious attacks, respectively. Despite the fact that centralized models performed pretty well, they are nevertheless susceptible to Single Point of Failure concerns.

4.4.2. Intrusion Detection using Federated Method

We deployed our model for FL experiments with three multiple client sets, K, with $K = 3$ (first set), $K = 9$ (second set), and $K = 15$ (third set). We utilized two cases to provide data to our various clients:

- Independent and Identically Distributed (IID): The distribution of data across the dataset corresponds to the distribution of data for each client.
- Non-Independent Identically Distributed (Non-IID): The distribution of data across the dataset is inconsistent with the distribution of data for each client.

Table 3. Settings for the ML classifiers used in the proposed solution.

Classifier	Parameter	Value
CNN	Convolutional layers	2–3 Conv1D
	Filters	16–74
	Kernel size	5
	Pooling layers	1 Global Average Pooling1D
	Hidden nodes	120–130
	Hidden layers	3–4
	Dropout	0.1–0.5
RNN	Hidden nodes	20–80
	Hidden LSTM layers	2
	Dropout	0.3
Global	Batch size	100
	Local epochs	20
	Global epochs	50
	Learning rate	0.01–0.5
	Regularization	L2
	Loss function	categorical_crossentropy
	Activation function	ReLu
	Classification function	SoftMax

Table 4. Evaluation of the Centralized Model in Intrusion Detection.

Class	Accuracy		Precision		Recall		F_1-Score	
	CNN	RNN	CNN	RNN	CNN	RNN	CNN	RNN
Normal	**0.93**	**0.94**	**0.94**	0.92	0.93	0.93	**0.93**	0.92
DDoS_UDP Attack	0.90	0.91	0.94	**0.95**	0.88	0.87	0.91	0.91
DDoS_ICMP Attack	0.82	0.81	0.81	0.78	0.84	0.82	0.82	0.80
Ransomware Attack	0.90	0.91	0.89	0.87	0.89	0.88	0.89	0.87
DDoS_HTTP Attack	0.64	0.54	0.59	0.57	0.68	0.63	0.63	0.6
SQL_injection Attack	0.64	0.68	0.66	0.67	0.61	0.69	0.63	0.68
Uploading Attack	0.72	0.78	0.77	0.82	0.62	0.77	0.69	0.79
DDoS_TCP Attack	0.91	0.92	0.92	0.92	0.90	0.92	0.91	0.92
Backdoor Attack	0.43	0.54	0.46	0.51	0.42	0.59	0.44	0.55
Vulnerability_scanner	0.67	0.62	0.63	0.56	0.67	0.54	0.65	0.55
Port_Scanning Attack	0.76	0.81	0.78	0.86	0.79	0.76	0.78	0.81
XSS Attack	0.52	0.69	0.56	0.72	0.44	0.58	0.49	0.64
Password Attack	0.77	0.68	0.73	0.65	0.86	0.79	0.79	0.71
MITM Attack	0.92	**0.94**	0.91	0.93	**0.93**	**0.95**	0.92	**0.94**
Fingerprinting Attack	0.63	0.65	0.61	0.71	0.66	0.68	0.63	0.69

Table 5 compares the accuracy outcomes of the global models, the best clients, and the worst clients after the 1st and 50th FL rounds using both the data distribution sets. As can be observed, the performance for all classes increased as the round increased. It is pretty common that in the case of IID, there is a smaller performance gap between the worst and best clients. The difference, however, is always quite large for Non-IID because some clients only have a few classes. As an illustration, for CNN with $K = 9$, the difference between the best and worst clients is quite large at the 1st FL round, but it becomes less for IID as we get closer to the 50th round. However, the gap for Non-IID is still big. Additionally, it can be seen that the detection accuracy is fairly competitive with the centralized learnings. We can therefore conclude that our FL model is extremely efficient and ensures increased privacy. Table 5 lists the Best Client Accuracy, Worst Client Accuracy, and Global Model Accuracy as letters B, W, and G, respectively.

Table 5. Evaluation of the FL Model in Intrusion Detection.

Classifier	Clients	1st Round						50th Round					
		IID			Non-IID			IID			Non-IID		
		B	W	G	B	W	G	B	W	G	B	W	G
CNN	*K* = 3	**63.23**	52.84	62.19	**59.89**	23.48	52.34	**91.34**	90.62	91.27	89.76	34.02	90.58
	K = 9	56.71	55.23	57.34	54.32	17.45	54.31	91.30	90.18	91.13	90.19	58.23	**90.73**
	K = 15	56.51	57.23	57.78	57.92	16.74	54.32	90.77	89.65	90.56	90.12	57.49	90.18
RNN	*K* = 3	**61.67**	54.87	61.28	**60.21**	24.64	53.37	**92.49**	92.08	92.37	91.26	72.98	**91.87**
	K = 9	58.43	53.67	56.84	53.68	19.42	55.79	92.41	92.01	92.28	91.13	69.37	91.53
	K = 15	59.65	52.69	58.92	57.97	17.16	54.72	92.19	91.98	92.02	91.06	74.71	91.07

As shown in Figure 4, we have displayed the learning process vs. accuracy graph for both centralized and FL techniques. The gap between the centralized approach (both CNN and RNN) and the FL method is rather substantial at the beginning. However, the margin shrinks as we move on to higher rounds, and by the 50th round, the results are quite competitive. This demonstrates that our proposed FL-method performs quite well in comparison to the centralized ML approaches and has further advantages that we will cover in the next sections.

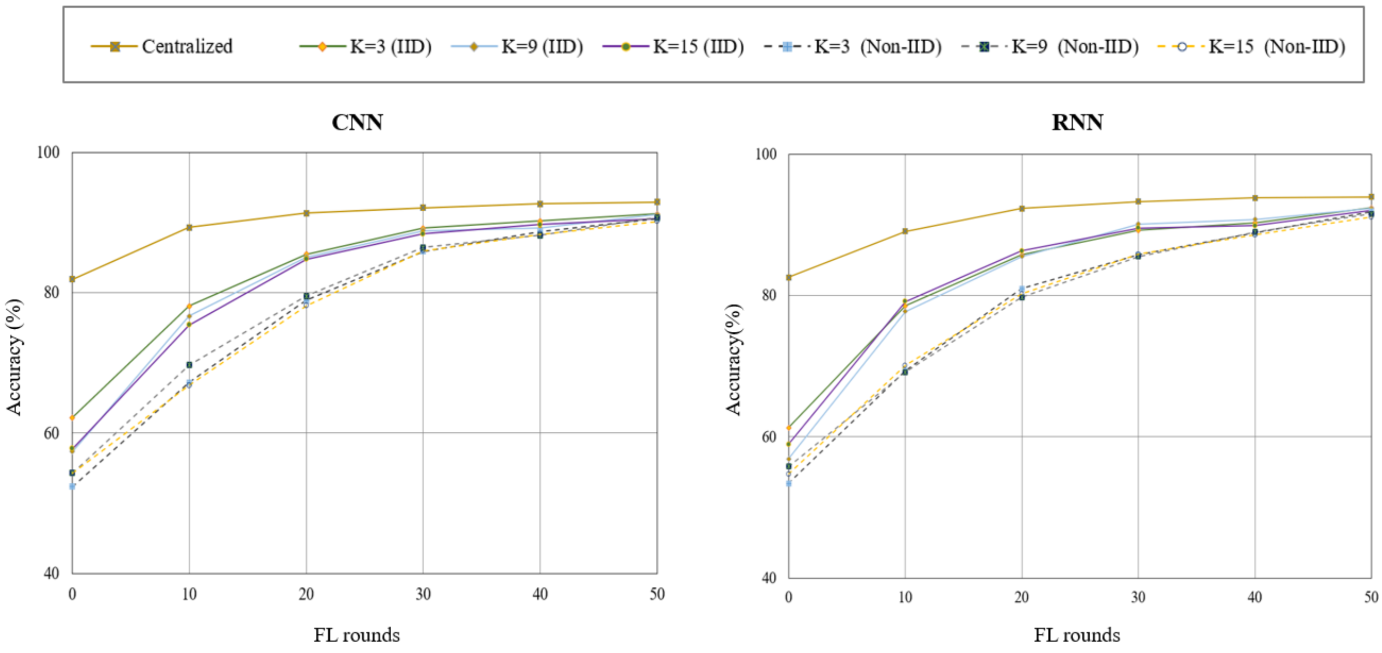

Figure 4. Learning vs. accuracy of centralized and FL.

Figure 5, which plots the true positive rate of intrusion detection versus the likelihood of a false positive, shows the Receiver Operating Characteristic (ROC) curve for the Edge-IIoTset dataset. Equations (10) and (11) represent the True Positive Rate and False Positive Rate, which can also be termed the Detection Rate and False Alarm Rate, respectively [32]. Given that all of the AUC (Area Under Curve) values fall between 0.91 and 0.95, we recognize that the performance of both ML approaches using our proposed FL method was quite satisfactory.

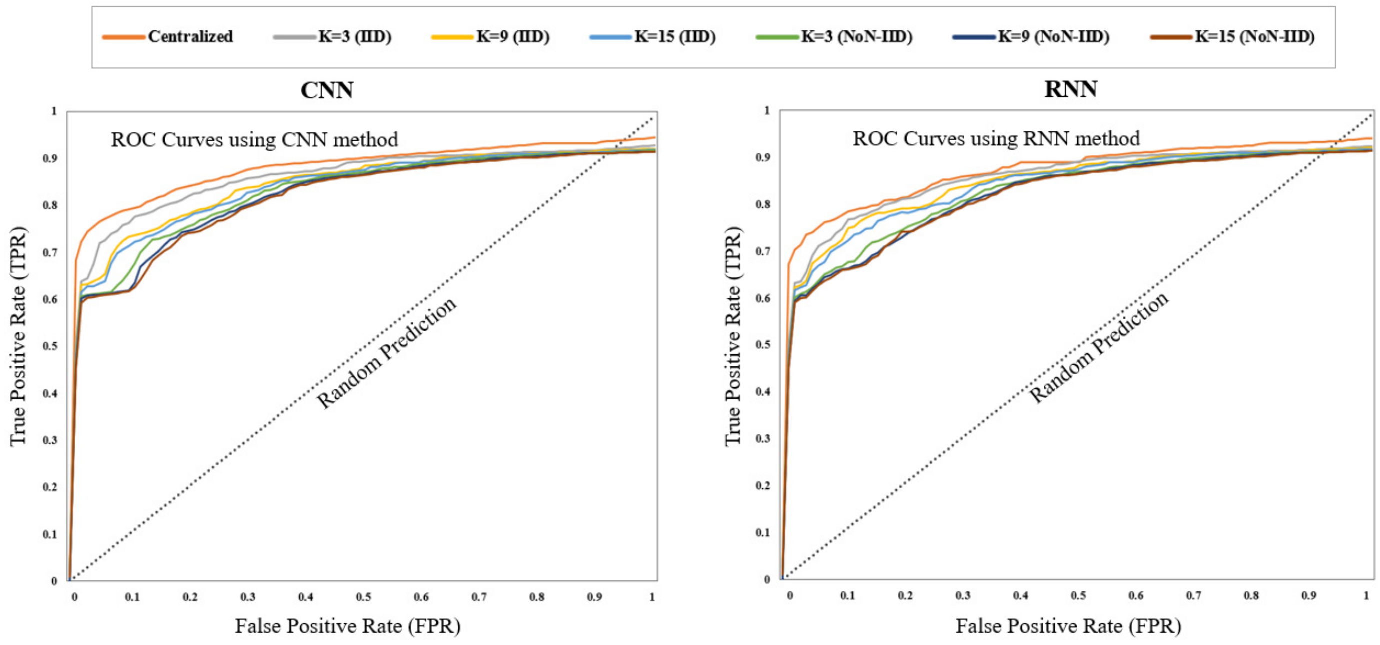

Figure 5. ROC curves for both CNN and RNN methods.

Figure 6 represents the training time of our proposed FL method after 50 FL rounds. We have demonstrated the time consumption for different numbers of FL clients (3, 9, and 15) for two different data distribution types (IID and Non-IID). As can be seen from the figure, the training time increases as the number of clients increases. It is evident from the figure that the training process is faster for the CNN method in comparison to RNN method. Additionally, it can be noticed that the different types of data distribution have no real significant impact on the time consumption, although in Non-IID cases, the time is slightly on the higher side. So, we can conclude that the training time for our proposed method varies depending only on the ML method and the number of clients used.

4.4.3. Comparison with Similar Works

Table 6 compares the efficacy of our work to that of comparable FL-based IDS approaches. The scope of the comparison covers the deployment year, datasets, ML classifiers, number of clients, and data distribution methods. As can be seen from the table, the proposed model is the only one that has addressed both IID and Non-IID data issues, and in previous section, we have demonstrated the performance of both of these data types. Ours is also the only one that evaluated the performance on the latest dataset (Edge-IIoTset).

Table 6. Comparison between Proposed Model and Relevant Works.

IoT IDS	Year	Dataset	Classifier	No. of Clients	IID	Non-IID
Nguyen et al. [33]	2019	Private Dataset	RNN-GRU	K = [5, 9, 15]	✗	✓
Zhao et al. [34]	2020	SEA	RNN-LSTM	K = 4	N/A	N/A
Li et al. [35]	2021	Gas Pipeline	CNN-GRU	K = [3, 5, 7]	✓	✗
Huong et al. [36]	2021	Bot-IoT	LocKedge	K = 4	✗	✓
Proposed Model	2022	Edge-IIoTset	CNN	K = [3, 9, 15]	✓	✓
			RNN	K = [3, 9, 15]	✓	✓

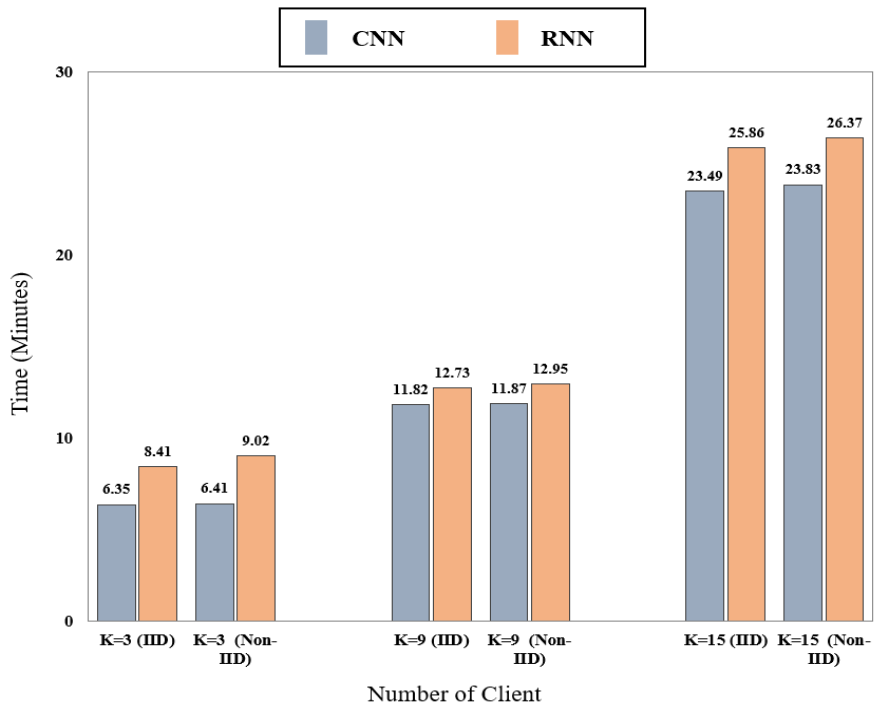

Figure 6. Number of client vs. training time for FL.

4.4.4. Discussion

Our proposed FL-based intrusion detection model is more effective for the following reasons.

- By using FL instead of conventional ML methods, IIoT devices are able to provide data that is both more secure and require less bandwidth for transmission. Less bandwidth is required as clients do not share whole data but only the learnings of their respective local models are shared.
- The massive amounts of private and secret information are no longer accessible through a single server. This ensures the privacy of the users and the security of the data.
- Due to the local representation of the models, devices are able to independently predict and recognize network anomalies even when they are disconnected. So, in case of any disconnection, local clients are still able to train their models and detect any kind of intrusions.
- As the number of FL rounds progressed, we achieved similar intrusion detection accuracy to centralized ML models. This is due to the fact that after each round, the models' performance gets enhanced by the learnings from all the client-end learnings, and they perform as accurately as a centralized model.

5. Conclusions and Future Scope

In this paper, we proposed an IoT intrusion detection system based on federated ML to increase security and privacy. Our primary objective was to identify unwanted intrusions so that IoT networks could be protected, and we performed our experiments on a recent dataset called Edge-IIoTset and ran experiments on both centralized and federated systems using two popular ML models called CNN and RNN. The experimental outcome demonstrated that with our proposed FL approach, we can achieve fairly competitive results in intrusion detection. In addition, we compared our technique to other FL-based IDS systems in both IID and non-IID scenarios. The experiments presented in this paper illustrate its applicability and usefulness, and have significant effects for the use of FL in the context of IoT networks.

In our future work, we intend to make the model more reliable when there might be malicious edge nodes on the network. Additionally, we will concentrate on a mechanism that uses an outlier detection filtering technique to prevent poisoning attempts that are injected gradually. The majority of IoT devices are capable of using several types of energy and processing power (CPU cycles per second). Hence, innovative FL protocols are required that offer criteria for choosing a group of local devices with enough resources. The devices must be chosen based on long-lasting backup power, sufficient memory, accurate data, and increased processing capability. If there are dishonest clients and servers, the usual FL method may potentially pose privacy issues. Therefore, further research is necessary on the issue of how to achieve a more reliable FL by removing all possible threats. Additionally, FL has some limitations that can affect the accuracy of the global model, such as devices that stop working in the middle of an operation, slow upload and model update times, clients that do not have much relevant data, etc. For future research, it is important to solve these problems, which make the global model much less accurate.

Author Contributions: Conceptualization, M.M.R.; funding acquisition, M.E. and S.R.S.; investigation, M.M.R. and S.U.K.; methodology, M.M.R.; project administration, M.M.R., S.U.K., F.E., M.A.R., M.E. and S.R.S.; software, M.M.R., S.U.K., F.E. and M.A.R.; supervision, M.E. and S.R.S.; validation, M.M.R., M.E. and S.R.S.; writing—original draft, M.M.R., S.U.K., F.E. and M.A.R.; writing—review and editing, M.M.R., S.U.K., F.E., M.A.R., M.E. and S.R.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not Applicable.

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

Abbreviation

AMQP	Advanced Message Queuing Protocol
ANN	Artificial Neural Networks
BERT	Bidirectional Encoder Representations from Transformers
BiLSTM	Bidirectional Long Short-Term Memory
CNN	Convolutional Neural Network
DoS	Denial of Service
FHMM	Factorial Hidden Markov Model
FL	Federated Learning
FN	False Negative
FP	False Positive
FPR	False Positive Rate
GA	Genetic Algorithms
GAN	Generative Adversarial Network
GPS	Global Positioning System
GRPC	Google Remote Procedure Call
GRU	Gated Recurrent Units (GRUs)
HIDS	Host-Based IDS
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IID	Independent Identically Distributed
INF	Infinite Value
IoMT	Internet of Medical Things
IoT	Internet of Things
KDD	Knowledge Discovery Dataset
KNN	K-Nearest Neighbor

LDP	Local Differential Privacy
LSTM	Long Short-Term Memory
MITM	Man-in-the-Middle
ML	Machine Learning
MQTT	Message Queuing Telemetry Transport
NAN	Not a Number
NIDS	Network-Based IDS
Non-IID	Non-Independent Identically Distributed
NSL-KDD	Network Security Laboratory-Knowledge Discovery Dataset
NSNN	Negative Selection Neural Network
PCA	Principle Component Analysis
R2L	Remote to Local
REDD	Reference Energy Disaggregation Dataset
RF	Random Forest
RFE	Recursive Feature Elimination
RFKD	Random Forest Differential Evolution with Kernel Density
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
SCRAPPOR	Sparse Coding Randomized Aggregable Privacy-Preserving Ordinal Response
SMOTE	Synthetic Minority Oversampling Technique
SPOF	Single Point of Failure
SQL	Structured Query Language
SSL	Secure Sockets Layer
SVM	Support Vector Machines
TCP	Transmission Control Protocol
TLS	Transport Layer Security
TN	True Negative
TP	True Positive
TPR	True Positive Rate
U2R	User to Root
UDP	User Datagram Protocol
VSN	Vehicle Sensor Networks
XGBoost	Enhanced Gradient Tree Boosting System

References

1. Guo, Y.; Zhao, Z.; He, K.; Lai, S.; Xia, J.; Fan, L. Efficient and flexible management for industrial Internet of Things: A federated learning approach. *Comput. Netw.* **2021**, *192*, 108122. [CrossRef]
2. Bag, S. Federated Learning—A Beginners Guide. 15 May 2021. Available online: https://www.analyticsvidhya.com/blog/2021/05/federated-learning-a-beginners-guide/ (accessed on 12 August 2022).
3. Yang, Z.; Chen, M.; Wong, K.-K.; Poor, H.V.; Cui, S. Federated Learning for 6G: Applications, Challenges, and Opportunities. *Engineering* **2022**, *8*, 33–41. [CrossRef]
4. Ahmad, R.; Alsmadi, I. Machine Learning Approaches to IoT Security: A Systematic Literature Review. *Internet Things* **2021**, *14*, 100365. [CrossRef]
5. Liao, H.-J.; Lin, C.-H.R.; Lin, Y.-C.; Tung, K.-Y. Intrusion detection system: A comprehensive review. *J. Netw. Comput. Appl.* **2013**, *36*, 16–24. [CrossRef]
6. Kirvan, P. Single Point of Failure (SPOF). Available online: https://www.techtarget.com/searchdatacenter/definition/Single-point-of-failure-SPOF (accessed on 13 August 2022).
7. Ferrag, M.A.; Friha, O.; Hamouda, D.; Maglaras, L.; Janicke, H. Edge-IIoTset: A New Comprehensive Realistic Cyber Security Dataset of IoT and IIoT Applications for Centralized and Federated Learning. *IEEE Access* **2022**, *10*, 40281–40306. [CrossRef]
8. Zarpelão, B.B.; Rodrigo, S.M.; Cláudio, T.K.; Sean, C.A. A survey of intrusion detection in Internet of Things. *J. Netw. Comput. Appl.* **2017**, *84*, 25–37. [CrossRef]
9. Benkhelifa, E.; Welsh, T.; Hamouda, W. A critical review of practices and challenges in intrusion detection systems for IoT: Toward universal and resilient systems. *IEEE Commun. Surv. Tutor.* **2018**, *20*, 3496–3509. [CrossRef]

10. Samek, W.; Stanczak, S.; Wiegand, T. The convergence of machine learning and communications. *arXiv* **2017**, arXiv:1708.08299. [CrossRef]
11. Gunduz, S.; Arslan, B.; Demirci, M. A Review of Machine Learning Solutions to Denial-of-Services Attacks in Wireless Sensor Networks. In Proceedings of the 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA), Miami, FL, USA, 9–11 December 2015.
12. Zhao, S.; Li, W.; Zia, T.; Zomaya, A.Y. A Dimension Reduction Model and Classifier for Anomaly-Based Intrusion Detection in Internet of Things. In Proceedings of the 2017 IEEE 15th International Conference on Dependable, Autonomic and Secure Computing, 15th International Conference on Pervasive Intelligence and Computing, 3rd International Conference on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech), Orlando, FL, USA, 6–10 November 2017.
13. Vallathan, G.; John, A.; Thirumalai, C.; Mohan, S.; Srivastava, G.; Lin, J.C.-W. Suspicious activity detection using deep learning in secure assisted living IoT environments. *J. Supercomput.* **2021**, *77*, 3242–3260. [CrossRef]
14. Ferrag, M.A.; Shu, L.; Djallel, H.; Choo, K.-K.R. Deep learning-based intrusion detection for distributed denial of service attack in Agriculture 4.0. *Electronics* **2021**, *10*, 1257. [CrossRef]
15. Pajouh, H.H.; Javidan, R.; Khayami, R.; Dehghantanha, A.; Choo, K.-K.R. A two-layer dimension reduction and two-tier classification model for anomaly-based intrusion detection in IoT backbone networks. *IEEE Trans. Emerg. Top. Comput.* **2016**, *7*, 314–323. [CrossRef]
16. Pamukov, M.E.; Poulkov, V.K.; Shterev, V.A. Negative Selection and Neural Network Based Algorithm for Intrusion Detection in IoT. In Proceedings of the 2018 41st International Conference on Telecommunications and Signal Processing (TSP), Athens, Greece, 4–6 July 2018.
17. Khan, L.U.; Pandey, S.R.; Tran, N.H.; Saad, W.; Han, Z.; Nguyen, M.N.H.; Hong, C.S. Federated Learning for Edge Networks: Resource Optimization and Incentive Mechanism. *IEEE Commun. Mag.* **2020**, *58*, 88–93. [CrossRef]
18. Tang, Z.; Hu, H.; Xu, C. A federated learning method for network intrusion detection. *Concurr. Comput. Pract. Exp.* **2022**, *34*, e6812. [CrossRef]
19. Chen, H.; Huang, S.; Zhang, D.; Xiao, M.; Skoglund, M.; Poor, H.V. Federated Learning over Wireless IoT Networks with Optimized Communication and Resources. *IEEE Internet Things J.* **2022**, *9*, 16592–16605. [CrossRef]
20. Cao, H.; Liu, S.; Zhao, R.; Xiong, X. IFed: A novel federated learning framework for local differential privacy in Power Internet of Things. *Int. J. Distrib. Sens. Netw.* **2020**, *16*, 155014772091969. [CrossRef]
21. Attota, D.C.; Mothukuri, V.; Parizi, R.M.; Pouriyeh, S. An ensemble multi-view federated learning intrusion detection for IoT. *IEEE Access* **2021**, *9*, 117734–117745. [CrossRef]
22. Tabassum, A.; Erbad, A.; Lebda, W.; Mohamed, A.; Guizani, M. Fedgan-ids: Privacy-preserving ids using gan and federated learning. *Comput. Commun.* **2022**, *192*, 299–310. [CrossRef]
23. Driss, M.; Almomani, I.; Huma, Z.; Ahmad, J. A federated learning framework for cyberattack detection in vehicular sensor networks. *Complex Intell. Syst.* **2022**, *8*, 4221–4235. [CrossRef]
24. Du, Z.; Wu, C.; Yoshinaga, T.; Yau, K.-L.A.; Ji, Y.; Li, J. Federated Learning for Vehicular Internet of Things: Recent Advances and Open Issues. *IEEE Open J. Comput. Soc.* **2020**, *1*, 45–61. [CrossRef]
25. Ghourabi, A. A Security Model Based on Light GBM and Transformer to Protect Healthcare Systems from Cyberattacks. *IEEE Access* **2022**, *10*, 48890–48903. [CrossRef]
26. An Introduction to Key gRPC Concepts, with an Overview of gRPC Architecture and RPC Life Cycle. Available online: https://grpc.io/docs/what-is-grpc/core-concepts/ (accessed on 10 September 2022).
27. McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; y Arcas, B.A. Communication-efficient learning of deep networks from decentralized data. *arXiv* **2016**, arXiv:1602.05629.
28. Zhang, Q.; Zhang, M.; Chen, T.; Sun, Z.; Ma, Y.; Yu, B. Recent advances in convolutional neural network acceleration. *Neurocomputing* **2019**, *323*, 37–51. [CrossRef]
29. Yu, Y.; Si, X.; Hu, C.; Zhang, J. A review of recurrent neural networks: LSTM cells and network architectures. *Neural Comput.* **2019**, *31*, 1235–1270. [CrossRef]
30. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
31. Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: Synthetic minority over-sampling technique. *J. Artif. Intell. Res.* **2002**, *16*, 321–357. [CrossRef]
32. Chan, H. *Introduction to Probability for Data Science*; Michigan Publishing: Ann Arbor, MI, USA, 2021.
33. Nguyen, T.D.; Marchal, S.; Miettinen, M.; Fereidooni, H.; Asokan, N.; Sadeghi, A.-R. DÏoT: A Federated Self-Learning Anomaly Detection System for IoT. In Proceedings of the 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), Dallas, TX, USA, 7–10 July 2019.
34. Zhao, R.; Yin, Y.; Shi, Y.; Xue, Z. Intelligent intrusion detection based on federated learning aided long short-term memory. *Phys. Commun.* **2020**, *42*, 101157. [CrossRef]

35. 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]
36. Huong, T.T.; Bac, T.P.; Long, D.M.; Thang, B.D.; Binh, N.T.; Luong, T.D.; Phuc, T.K. Lockedge: Low-complexity cyberattack detection in iot edge computing. *IEEE Access* **2021**, *9*, 29696–29710. [CrossRef]