electronics

MDPI

Article

Asynchronous Privacy-Preservation Federated Learning Method for Mobile Edge Network in Industrial Internet of Things Ecosystem

John Owoicho Odeh [1], Xiaolong Yang [1,*], Cosmas Ifeanyi Nwakanma [2] and Sahraoui Dhelim [3]

[1] Computer and Communication Engineering, University of Science and Technology, Beijing 100083, China; b20190668@xs.ustb.edu.cn
[2] ICT-Convergence Research Center, Kumoh National Institute of Technology, Gumi 39177, Republic of Korea; cosmas.ifeanyi@kumoh.ac.kr
[3] School of Computer Science, University College Dublin, Belfield, D04V1w8 Dublin, Ireland; sahraoui.dhelim@ucd.ie
* Correspondence: yangxl@ustb.edu.cn

Abstract: The typical industrial Internet of Things (IIoT) network system relies on a real-time data upload for timely processing. However, the incidence of device heterogeneity, high network latency, or a malicious central server during transmission has a propensity for privacy leakage or loss of model accuracy. Federated learning comes in handy, as the edge server requires less time and enables local data processing to reduce the delay to the data upload. It allows neighboring edge nodes to share data while maintaining data privacy and confidentiality. However, this can be challenged by a network disruption making edge nodes or sensors go offline or experience an alteration in the learning process, thereby exposing the already transmitted model to a malicious server that eavesdrops on the channel, intercepts the model in transit, and gleans the information, evading the privacy of the model within the network. To mitigate this effect, this paper proposes asynchronous privacy-preservation federated learning for mobile edge networks in the IIoT ecosystem (APPFL-MEN) that incorporates the iteration model design update strategy (IMDUS) scheme, enabling the edge server to share more real-time model updates with online nodes and less data sharing with offline nodes, without exposing the privacy of the data to a malicious node or a hack. In addition, it adopts a double-weight modification strategy during communication between the edge node and the edge server or gateway for an enhanced model training process. Furthermore, it allows a convergence boosting process, resulting in a less error-prone, secured global model. The performance evaluation with numerical results shows good accuracy, efficiency, and lower bandwidth usage by APPFL-MEN while preserving model privacy compared to state-of-the-art methods.

Keywords: asynchronous privacy preservation; Internet of Things; industrial Internet of Things ecosystem; iteration model design update strategy; double-weight modification; convergence boosting process

Citation: Odeh, J.O.; Yang, X.; Nwakanma, C.I.; Dhelim, S. Asynchronous Privacy-Preservation Federated Learning Method for Mobile Edge Network in Industrial Internet of Things Ecosystem. *Electronics* **2024**, *13*, 1610. https://doi.org/10.3390/electronics13091610

Academic Editor: Hung-Yu Chien

Received: 27 March 2024
Revised: 17 April 2024
Accepted: 22 April 2024
Published: 23 April 2024

1. Introduction

The rise and growth of machine learning has rapidly changed the digital landscape enabling computers to imitate human behavior, with the most recent use of digital twins. This disruptive technology has led to the advancement of text, image, or speech recognition technology [1]. However, the gains come with the need for lots of data for learning purposes. In addition, there is an unprecedented surge in the number of connected devices and the evolution of intelligence paradigms with large-scale connectivity [2]. Thus, the Internet of Things (IoT) accounts for 1.68 devices per human as far back as 2020 to be about 9.76 billion. The IoT is estimated to have increased to about 15.72 billion devices and is likely to increase to about 29.42 billion devices by 2030 [3]. The most common use case for the industrial IoT

(IIoT) devices in the edge network segment is internet and media devices which include smartphones, laptops, etc., which are used as mobile tracking devices. This is depicted as shown in the result of a statistical survey reflecting the number of connected and forecasted IIoT edge devices globally, between 2019 and 2030, in Figure 1.

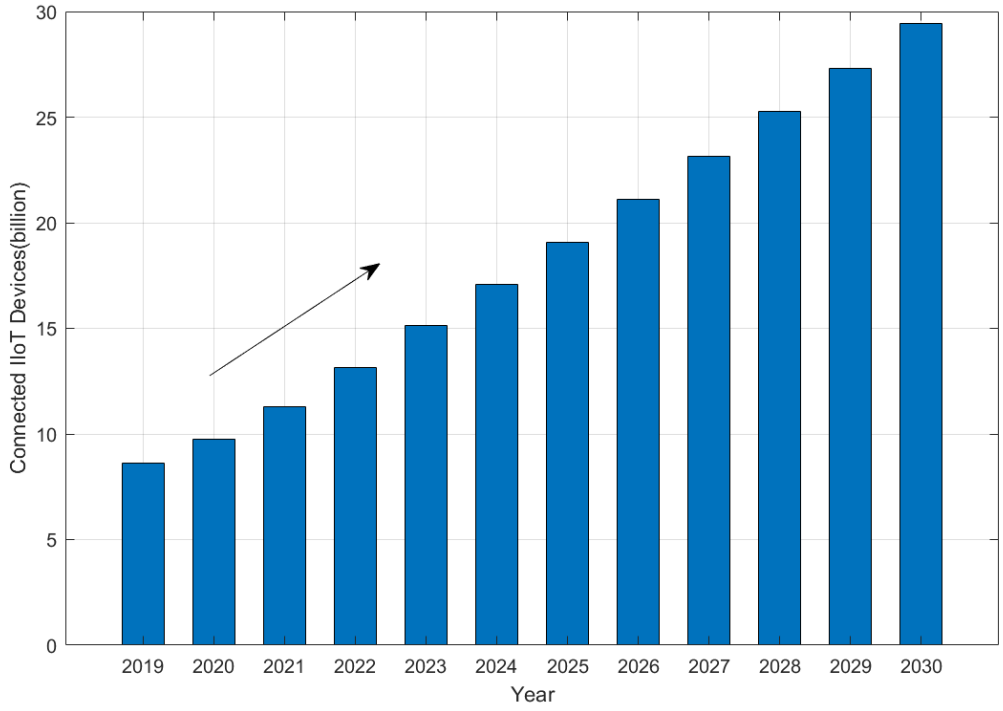

Figure 1. Global estimation of connected industrial IoT edge devices globally to 2030 [3].

With the advances in technologies such as edge network computing, the IoT, digital twin, and the IIoT as seen in the fourth industrial revolution and the current Society 5.0, there is exponential growth and development of the mobile edge networks [4]. The implication is the increase in data generation by these sensors or edge infrastructures. Thus, in this mobile network, constraints such as limitations in computing resources, high bandwidth usage, latency in network connectivity, heterogeneity, and threats of malicious cloning of devices have impacted the processing of the needed data, as well as creating loopholes for affecting the privacy of data [5]. To effectively control and handle the increased data, they are processed and transmitted to the centralized data centers which cause congestion, delayed processing, and possible collision attacks. This known challenge has given rise to multi-edge computing [6], which is within a line-of-sight technique or near data source communication platform that combines the task of computing, storing, networking, and other securing capabilities within the decentralized edge network. The result is an effective reduction in bandwidth usage and quicker response time. To enhance the effective data processing in the mobile network as experienced in the IIoT network, the improved federated learning technique introduces a distributed learning algorithm that transmits only locally learned models to the mobile transmission unit within the edge network [7]. The federated learning edge devices collaborate to locally build and optimize a learning model that is locally kept by interacting with the edge server [8]. This ensures data privacy protection as the data is not shared with neighboring edge nodes, but only trained through interaction with the federated learning server. Privacy preservation is thus achieved in terms of anonymity, unlinkability, undetectability, and identity management [9]. Even the transmitting and learning of data is carried out at a rate less than the secrecy capacity.

2. Related Works

In securing model updates and training, it is noted that federated learning operation in mobile edge computing (MEC) is stimulated by privacy protection and security features. Adversaries can target and retrieve vital details about an edge device from the shared model updates. In addition, malicious devices can join in the model training process to infuse fake model updates and negatively affect the trained model's accuracy. To have a secured model transmission, a cryptographic protocol called secured multi-party computation [10,11] is aimed at obfuscating the personal information and ensuring zero knowledge between the nodes involved. This guarantees the distribution of computation amongst the multi-edge devices without divulging any private information. The models involved are aggregated and masked by the addition of random numbers. Then, the encoded segments are distributed to the connected nodes for computation to ensure data privacy and trust. The secured multi-party computation scheme allows the nodes to connect, without knowing each other's confidential details [12]. This helps to handle complex issues like mobile crowdsourcing and traffic congestion.

A similar approach is the anonymous differential privacy (DP) [13] that uses the artificial noise added to a locally trained model before being transmitted and aggregated. This is to stop adversaries from accessing the leaked model parameters, even as the enhanced security measure comes at the expense of model accuracy and computation. A combined optimization will help strike a balance between computation overhead, accuracy, and privacy protection. The main technique is the addition of Gaussian noise as highlighted by [14,15] for the primary aim of protecting the model parameter. Homomorphic encryption, a token-based privacy mechanism that performs calculations on encrypted data, was proposed [16]. It was designed in the context of federated learning where edge clients or devices generate private and public tokens, for the decryption and encryption of locally trained models. This method further aggregates all the encrypted model updates at the edge server.

However, the operation of this homomorphic encryption added computational complexities on the edge devices, in terms of bandwidth, time and power, and large data size. Another technique proposed is the batch method [17], also known as BatchCrypt. It reduces the transmission and encryption cost to the barest minimum by ensuring that edge devices arrange the gradients into quantum (low-bit integer format) and transmit the encrypted batch of encoded gradients [18]. In this format, the total size and other overhead are greatly reduced. An asynchronous federate learning architecture, a differentially private asynchronous federated learning (DP-AFL) scheme [19], was designed by leveraging a distributed peer-to-peer update scheme instead of the centralized update to mitigate the privacy threat posed by the centralized server. The privacy of updated models in federated learning was enhanced by co-opting local differential privacy into a gradient descent training scheme and adding Gaussian noise. However, this is prone to model disruption and error due to noise addition. Furthermore, ref. [20] proposed a method called the federated learning-based proactive content caching scheme based on hierarchical architectural design, consisting of user and edge servers for the industrial network. This decentralized, local storage, local training, and resource-constrained tolerant feature has made the federated learning for MEC in urban informatics a good approach [21,22]. This advancement in the operations of the distributed federated learning on MEC-powered edge devices was challenged due to privacy and security concerns and resource and deployment constraints.

Some privacy threats targeted at the multi-edge network can disrupt the IIoT architecture; these threats may include the compromised unsecured federated learning algorithm, where a hacker can read, modify, and even flood the network modified updates, thereby breaching the privacy and causing a delay in the model update [23]. Furthermore, a man-in-the-middle attack can cause an interception, relays, and potential alteration of edge communication, thereby ceasing control over one or more edge nodes. The falsification of the local model by a malicious server threatens the authenticity and integrity of both the model update and the network resulting in loss and accuracy of the local dataset [24]. A se-

cured network and an authentication/verification mechanism in the multi-edge network will bring reprieve to the system. Furthermore, ensuring the use of up-to-date firmware of the various heterogeneous nodes will suffice [5]. A comparison of these related works is shown in Table 1 as follows.

Table 1. Comparison of related works.

Privacy Preservation	IMDUS	Edge Network	Convergence Process	Research	Reference
✓	×	×	×	Micro-LED as a promising candidate for high-speed Visible Light Communication	[11]
✓	×	×	×	Visible Light Communication in 6G: Advances, Challenges, and Prospects	[12]
✓	×	×	×	Learning Differentially Private Recurrent Language Models	[13]
✓	×	×	×	Practical Secure Aggregation for Privacy-Preserving Machine Learning	[14]
✓	✓	×	×	Secure Multi-Party Computation: Theory, Practice and Applications	[15]
✓	✓	✓	×	Update or Wait: how to keep Data Your Fresh	[16]
✓	✓	×	×	BatchCrypt: Efficient Homomorphic Encryption for (Cross-Silo) Federated learning	[17]
✓	×	×	×	Learning in the Air: Secure Federated Learning for UAV-Assisted Crowdsensing	[18]
✓	✓	×	×	Federated Learning Based Proactive Content Caching in Edge Computing	[19]
✓	✓	✓	×	Task Allocation Algorithm for Energy Resources Providing Frequency Containment Reserves	[20]
✓	✓	×	×	Modeling and Analysis of a Shared Edge Caching System for Connected Cars and IIoT Applications	[21]
✓	✓	×	✓	Asynchronous Federated Optimization	[22]
✓	×	✓	×	Asynchronous Federated Learning Over Wireless Communication Networks	[23]
✓	×	✓	×	Privacy-Preserving Federated Learning for Internet of Medical Things under Edge Computing	[5]
✓	✓	✓	✓	Asynchronous Privacy-Preservation Federated Learning method for Mobile Edge Network in IIoT Ecosystem	**This Study**

Although there have been some semi-asynchronous updates periodically, there are more algorithms for integrating asynchronous into synchronous federated learning. Here, edge nodes broadcast the full local model to all neighboring nodes using a data-parallel distributed protocol, irrespective of the online or offline status of the node. This poses a higher privacy threat to anonymous or malicious nodes. Therefore, to properly handle the issue of privacy preservation challenges for multi-edge devices in the ecosystem, asynchronous federated learning will be appropriate. Thus, this paper proposes an asynchronous privacy preservation federated learning method that enhances secured resource sharing, reduces loss function, and adopts double-weight modification by gradient compression and convergence boosting. This allows more secured real-time connected edge nodes to upload/download, while offline nodes are granted minimal upload status. Thus, the main contributions include the following:

1. We propose an Asynchronous Privacy-Preservation Federated Learning model, by leveraging the iteration model design update strategy scheme for more participation of

online edge nodes, and fewer offline nodes instead of the all-at-once update, to mitigate the privacy threat by malicious adversaries.

2. We enhance the asynchronous federated learning for optimized adaptation and the secured model training process by incorporating the double-weighted modification to mitigate the effect of asynchronous federated learning which allows mobile transmitting units (MTUs) to upload the received model immediately from nodes without waiting for other nodes that reflect the level of staleness, the multi-edge computing power of the online edge nodes.
3. We use the proposed iteration model design update strategy (IMDUS) to ensure the integrity of each node to provide the correct least absolute error for the updated model in the asynchronous scenario to reduce the effect of error and loss function from nodes going offline, by ensuring an increased gradient compression level.
4. To guarantee the optimization of the convergence boosting process, we use the real model enhancement algorithm for validating updates and a reward system for increased participation of online nodes with good model updates, while reducing participation of offline nodes with bad model updates.

The remaining part of the paper is organized as follows; Section 3 shows the system model, threat model, and a formulated threat scenario. In Section 4, we provide the full details of the APPFL-MEN, the dual weight modification and convergence process, and the response-adaptive threshold gradient compression. Section 4 presents the system configuration and numerical and analytical results of the proposed scheme. The final section, Section 5, concludes and discusses the whole research work.

3. Methodology and System Threat Model

The conventional IIoT network infrastructure relies on real-time data uploads to ensure prompt processing. However, device heterogeneity, high network latency, or hostile systems might cause communication overhead and loss of model accuracy. Federated learning becomes a perfect fit, as we propose an asynchronous privacy-preservation federated learning method for mobile edge networks (APPFL-MEN) in the IIoT ecosystem that incorporates the IMDUS scheme, enabling the edge server to share more real-time model updates from online edge nodes and less data sharing from offline edge nodes, without exposing the data privacy to a malicious node [22]. In addition, it uses a double-weight modification method during communication between the edge node and the edge server or gateway to improve model training. A gradient compression method by edge nodes reduces the dataset to the smallest possible size during the training and transmission processes. Then, it enables a convergence-boosting process, resulting in a less error-prone, more secure global model. The dataset includes the MNIST picture dataset, which will be trained and classified depending on their type. The edge nodes read these datasets as local models in the compressed mode before securely uploading them. These datasets are securely managed via IMDUS, an asynchronous federated learning method that handles model training. It operates in rounds (iterations) of upload and download channels and time slots between edge servers and edge nodes (IIENs). The major role of this IMDUS is to ensure that up-to-date local models and global updates are uploaded and downloaded, with priority given to models from online nodes [9]. In building the system, we use EdgeCloudSim v4.0 to create a computer model of the edge network-enabled IIoT system using the Hierarchical Edge Computing architecture depicted in Figure 2, and we use the basic Python language for testing on the Windows 10 operating system. To assess the performance of the proposed strategy, experiments are conducted on the MNIST dataset, a database of [1, 0] digits used as a benchmark for image classification tasks in machine learning. According to numerical evaluation, our result shows a relative advantage over the DP and the DroppFL methods based on accuracy, efficiency, and bandwidth utilization while maintaining model privacy [25].

Federated learning is designed to be a super artificial intelligent technology entrenched in the Industrial Revolution 4.0 and the current Society 5.0, deployed to execute a well-

organized machine learning operation within multi-edge nodes and still maintain a secured and private nature of data [26,27]. Different algorithms are used for machine learning in federated learning. Examples include the recurrent neural network (RNN), convolutional neural network (CNN), and other traditional algorithms. Traditional federated learning technique is enabled by edge servers, devices, and MTUs. The MTUs are responsible for providing a network service to the nodes, as the edge server collects the upload gradients/locally trained model by the edge device and updates the parameter of the global model based on an optimized algorithm. The edge devices conduct the model training locally and send updated gradient information to the edge server after each round of training. In return, edge nodes receive the updated global model from the edge server. This form of interaction is one-to-one, between an edge node and the edge server, with little or no information about the next edge node. However, global models are jointly maintained as this ensures the confidentiality and integrity of data as shown in Figure 2.

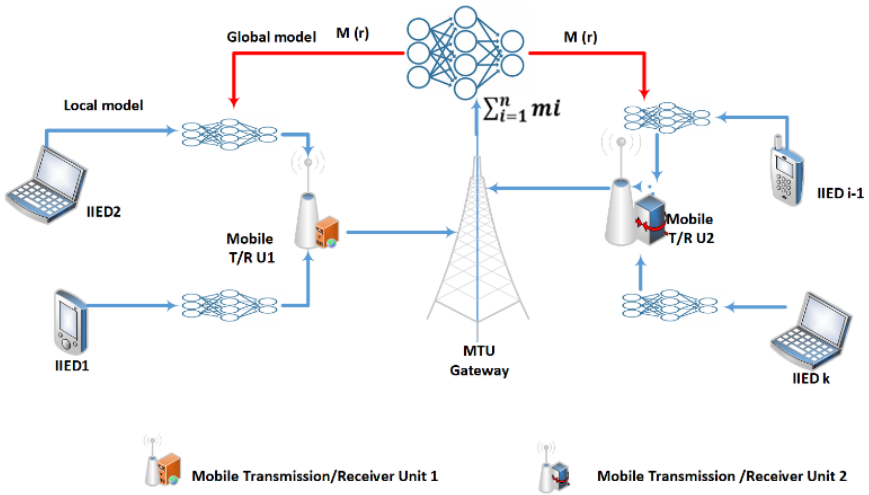

Figure 2. Asynchronous federated learning mobile edge network.

The asynchronous federated learning technique in the smart edge network is considered a solution for proper task distribution and handling of privacy protection challenges for mobile multi-edge devices. Although asynchronous federated learning has its setbacks in the timing to manage the re-connection of offline nodes and transfer of trained models, it is a good technique in mobile multi-edge networks of devices, for prorating resources for needed tasks such as data sharing. In addition to sharing the generated dataset, data generators take advantage of federated learning to jointly partake in the training of the global data model. Since the benefit outweighs the cost, we attempt to implement the federated learning in the industrial internet of (network) things for dataset training roles in the MEC. In contrast to the traditional method of centralization of a single-point learning technique, the deployment of federated learning is a viable solution to security and privacy concerns, as it adopts the local training model. With the increase in heterogeneous devices and specifications, more exposure and vulnerability issues are being discovered. This includes the centralization challenge of the edge controller/server that collects, updates, and aggregates all datasets in the federated learning scheme. If there is a record of system breakdown, there will be a crash in the whole process due to a single point of failure, giving access to adversaries to eavesdrop on private data. However, having this fault tolerant system that works out a self-recovery plan from model training error, and loss function due to going offline, makes it resilient [28]. It does this by re-routing and connecting to a nearby edge server (Mobile T/R unit) or a load balancer/fail-over system to return as an online node and restart the model training process for data set upload. Although, this disconnection makes an edge node susceptible to eavesdropping and malicious adversary attack.

Furthermore, it is known that the training of local models by the edge devices is carried out based on the global model sent by the edge server [5,9]. If a malicious edge server

learns the edge device's private dataset by broadcasting a hazardous model to recipient devices without their knowledge and collecting their parameters in return, a poisoning attack may occur. This may cause the edge devices to increase the likelihood function of parameter sharing and distribution on their data instead of reducing the loss function [23]. Another threat model is a probable orchestrated attack by adversaries through learning the information and analyzing updates confidential to edge devices. The adversary device can also send out a pseudo update in exchange for an authentic global model, thereby reducing the efficiency and accuracy level of the model and then a possible breakdown of the model learning and training process, leading to possible misinformation and system degradation. Thus, mitigating this requires a well-organized federated learning technique for enhanced privacy protection for the edge device, ensuring gradient communication level reduction and convergence enhancement.

4. APPFL-MEN for Industrial Internet of Things Ecosystem

The three components of the APPFL-MEN include gradient compression, convergence boosting, and weight modification. Traditional federated learning treats all edge nodes within the network equally, irrespective of their role of obtaining the global model after the training and sharing of the model which is not a suitable reward for online nodes with a good model and offline nodes with an error-prone model. The seamless method to achieve weight modification is to alter the asynchronous nature of the dual weight (parameter and sample weights). The parameter weight is first determined by a node's parameter download time and the corresponding gradient upload time. The sample weight is determined by the fraction of a sample of a node sample to the summed-up sample of all the participating edge nodes, $Q_i = N^{-1} * N$. This is to ascertain the number of participating nodes per time. The parameter weight is determined by a node's parameter download time and the corresponding gradient upload time, $T^{'} = (T^{'}U) - (T^{'}D)$. The convergence boosting process thus allows for more good models, with less error-prone models to complete the process of gradient compression, update verification, and double weight modification. Furthermore, to ensure that a true error rate is generated, a guided federated learning IMDUS scheme is adopted in an asynchronous mode to show the true position of the model trainer and the correct minimum error in its up-to-date model. The operation within the three components of our proposed model is represented in Figure 3.

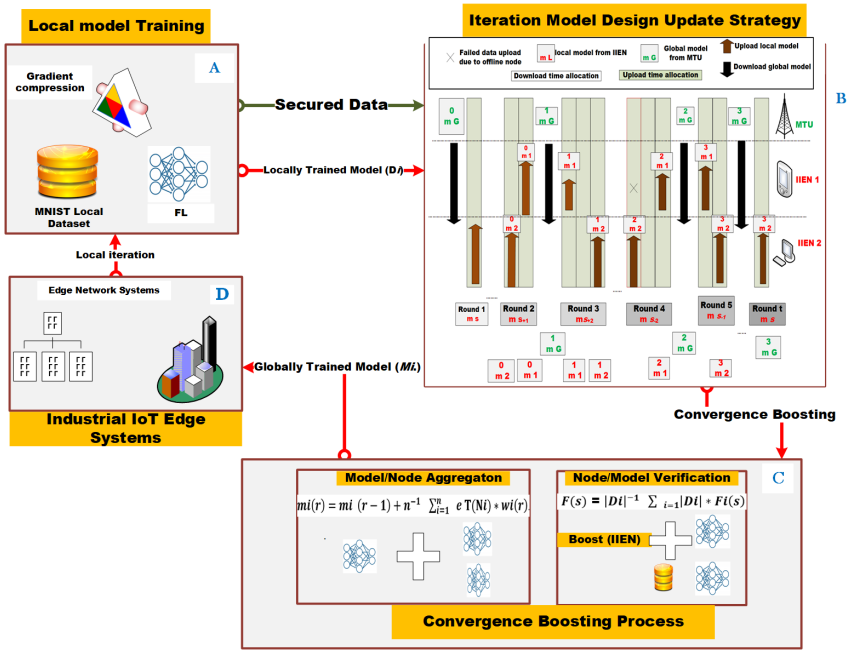

Figure 3. Proposed APPL-MEN showing A: local model training stage; B: IMDUS strategy stage; C: Convergence Boosting Process, and D: Global Model download to the IIoT Ecosystem.

4.1. Problem Formulation

The set MTU is represented by $X = x_1, x_2, \ldots x_i$ and the set of smart mobile edge devices by $N = N_1, N_2, N_3, \ldots N_i$. With each mobile edge device generating local data, w_i = input of the ML model and z_i = trained model output. Therefore, the local dataset is $d_i = (w_1, z_1), (w_2, z_2), \ldots (w_i, z_i)$, where $d_i \in D$ or $D = (\cup\, d_i)$.

The connection to the smart network is achieved when the edge device comes within the network area of coverage. The objective is to learn a widely acceptable predictive trained model $M = h(s, w)$, with the training dataset D, but when an edge device goes offline, leading to a loss function for each edge node's locally generated dataset D_i.

$$F_i(s) = |D_i|^{-1} \sum_{i} \in D_j * f_i * (h(s, w), z) \tag{1}$$

Having $F_i(h(s, w), z)$ as a loss function for the ith dataset (w_i, z_i), with model output s, the objective loss function is

$$F(s) = |D_i|^{-1} \sum_{i=1} |D_i| * F_i(s) \tag{2}$$

This makes the target of the asynchronous federated learning to be, firstly, to train and update the global model as soon as it is received from an online IIEN, instead of waiting for all connected IIEN within the context of the smart edge network [24], ensure the minimization of the error (e) and loss function $F(s)$ and, lastly, to compress the trained model as a means to strengthen the privacy protection mechanism even in a situation where an edge device may go offline due to mobility and lack of connectivity.

$$h(s) = \arg\min_{s} F(s) \quad \text{subject to } s \in \{s(r) : r < R\}$$

$$\text{s.t. } Pr(CL(s_i \in R^d)) \geq \exp(e) \cdot Pr(CL(s_i \in R^d)$$

$$\forall y_i \in Y, i \in \{1, 2, \ldots, k\} \tag{3}$$

The $s(r)$ is the set of highly compressed, aggregated models at iteration r, and R is the highest updating iteration.

$$Pr(CL(m_i \in R^d) \geq \exp(ePr(CL(m_i \in R^d),$$

where compression as a privacy measure is guaranteed for updated parameters i and $s(r)$, obtained from the equation.

$$s(r + 1) = s(r) + k^{-1} \sum_{i=1} s_i \tag{4}$$

where s_i is the update from edge device y_i in iteration r.

4.2. Gradient Compression and Object Function

The gradient compression mechanism is an integrated algorithm that reduces the communication overhead. The number of locally trained and uploaded models from an edge node is reduced to the smallest possible size during transmission to the edge server. Moreover, since the nodes consist of online and offline entities, priority is given to the online nodes to communicate with the parameter of the edge server, thereby reducing network congestion, the probability of privacy breach, and any loss function that may arise as a result of an increase in the number of transmitting nodes. This also boosts the accuracy and efficiency of the learning process. Hence, to further describe and represent the context of the edge network system in terms of the gradient compression and descent algorithm, the parameter server (the edge server) connects and communicates with the learning node N, where the learned model parameters are updated. All learners receive a broadcast of

s^{i-1} from the parameter server in the ith iteration. A computation of $N-1(s^1)$ is carried by each learning node $n \in N$ and then uploaded to the edge server. This edge server then updates the learned model by an iteration in the gradient compression algorithm.

$$s^1 = s^{i-1} - \alpha N^{i-1} \tag{5}$$

The representation of the iteration of variation in the model has α, N^{i-1} as the learning rate and aggregation gradient, respectively. The algorithm can complete an iteration by ensuring communication between the learning nodes, specifically between "real-time or active nodes" and the edge server while communicating less with other "off-line or inactive nodes", N, resulting in $N_R + N_F = N$. This idea is to reduce the communication overhead such as time, power, and bandwidth usage during compression [6,7]. Although, nodes N_F that "talk less" with the server still individually accumulate or aggregate the gradient and eventually increase wide enough to engage in communication. This aggregation gradient becomes

$$N = N_F^{i-1} + N_R^{i-1} \tag{6}$$

With the N and N_F having a total number of parameters in the set, respectively, let the set N_F satisfy the following equation.

$$\frac{||N_R^{i-1}||^2}{n} \geq \frac{||N_F^{i-1}||^2}{n_f} \tag{7}$$

If Equation (5) is substituted into (7) , it gives

$$||N_F^{i-1}||^2 \leq \frac{nF}{\alpha^2 n} \left\| s^i - s^{i-1} \right\|^2 \tag{8}$$

Furthermore, given as, $||N_F^{i-1}||^2 = ok$. This implies that by the inequality of basic mathematical and geometric means, $||N_F^{i-1}||^2$ infers that the equation

$$||N_F^{i-1}||^2 \leq N_F \sum_{n \in N_F} ||n(s^i)||^2 \tag{9}$$

Equation (7) holds, if the node $n \in N_F$ satisfies the conditions

$$||n(s^i)||^2 \leq \frac{1}{\alpha^2 n_f n} ||S^I - S^{I-1}||^2 \tag{10}$$

As the total number of sets of N_F is not obtained beforehand, we input a proportional coefficient δ to show the sum of nodes in the set, N_F, which results in $N_F = n\delta$. Thus, from Equation (10),

$$||n^{(s^i)}||^2 \leq \frac{1}{\alpha^2 n_f n} ||S^i - S^{i-1}||^2 \tag{11}$$

It is however clear that acquiring $s_i - (s_i - 1)$ is quite hard. However, during the learning process, it is observed that the parameter changes tend to happen smoothly, thus having $s_i - (s_i - 1)$ estimated as

$$|S^I - S^{I-1}|^2 \approx \sum_{x-1}^{x} Y(S^{i-x} - S^{I-1-x}) \tag{12}$$

Having Y and x as coefficients, we put $Y = 1/x$. Let the values of (12) be substituted into (11);

$$|n(s^i)|^2 \leq 1/\alpha^2 \delta n^2 |S^I - S^{I-1}|^2 \tag{13}$$

Here, the node carries out a self-testing examination for the gradient compression check of the federated learning after each iteration of learning. It is however observed that the gradient scarcely or partially meets the requirement for (13), making the node

interact with the parameter server. This will occur; otherwise, there will be a skipping of communication in this iteration, and the learning nodes gradient locally will continue to perform the next iteration of learning. It is observed that the compression mechanism is a privacy-protective technique, as it gives little or no room for easy deciphering of the trained model by a malicious edge node or exposure to the same as summarized in Algorithm 1.

Algorithm 1 APPLM Algorithm

Compute: local dataset D, global model parameters $m_i(r)$
Output: trained global model M
for each edge device E **do**
  for each Mobile Transmission Unit (MTU) T **do**
    for $r \leq R$ **do**
      while $r = 1$ **do**
        Train an updated model M_E using the local dataset D_E
        Compute the updated model parameters $m_i(r)$
        Send out $m_i(r)$ to the edge server
        if edge server receives $m_i(r)$ from edge nodes **then**
          Get $m_i(r)$ from edge nodes
        end if
      end while
    end for
  end for
  Compute updated global model M
  Compress the model parameters Pr to guarantee privacy
  Sample a subset of MTU, edge nodes
  Send out $m_i(r)$ and Pr to edge server
  if edge server receives $m_i(r)$ and Pr **then**
    Return M
  end if
end for
Output M

4.3. Convergence Process

One of the discoveries in the deployment of federated learning in the MEC is the challenges encountered in sustaining synchronized updates between the central edge server and edge device of this smart network. Due to the mobile nature of smart edge devices, some devices participate in getting global model updates during the learning phase. This makes an edge device to participate in real-time or asynchronously. Therefore, to ensure convergence where devices train, learn good data, and compute models accurately, we allow the secured connected nodes with less error to generate more good global models and fewer bad models from offline nodes having more errors. An edge device has its locally generated dataset,

$D_i = (w_1, z_1), (w_2, z_2) \ldots (w_i, z_i)$, global model, $(r-1)$ and receives a set of updated models

$$\sum_{i=1}^{n} M_i \tag{14}$$

weighted alongside an error rate of model S. The edge node (IIEN) Ni picks the model with a minimum error rate to carry out data aggregation and discards the model with a high error rate as represented.

$$m_i(r) = m_i(r-1) + n^{-1} \cdots \sum_{i=1}^{N} error * s_i(r) \tag{15}$$

We use N_F as a set of offline nodes and s as the size of the compressed model update. All possible errors are eliminated from the input model. Thus,

$$w = 1 - error * \sum_{i-1}^{N} error \tag{16}$$

To ensure a smooth convergence process, the nodes responsible for training models are allowed to generate the minimum error rate of the up-to-date model, which shows the efficient nature of the gradient aggregation process. Furthermore, to ensure that a true error rate is generated, a guided federated learning IMDUS scheme is adopted in an asynchronous mode to show the true position of the model trainer and the correct minimum error in its up-to-date model. All updated models in the verification scheme are represented by an edge node, with the value of r, minimum error rate, and size. During every round of training, an edge node can verify the minimum error rate of the updated models it receives with a proportion P, depending on its data. The edge node then shares the verified data with other edge nodes. The total size of the node $m_i(r)$ can be calculated by

$$Size(i) = error(m_i(r)) \cdots + \sum_{i-1}^{N} error(m_i(r)) \tag{17}$$

With the minimum error rate received by the edge node y_i within round r, the edge node in the IMDUS scheme shows that a node is certified by other connected edge nodes within its training circle. The recorded certified results give a total node size, $T(N_i)$, for an IIEN which, if substituted, gives

$$m_i(r) = m_i(r-1) + n^{-1} \sum_{i=1}^{N} \ldots error.(N_i) * w_i(r) \tag{18}$$

The significant error rate in this asynchronous transmission of the dataset, maybe caused by differences in the computation power, bandwidth, participation time of the federated learning process, gradient compression, or interruptions due to a loss of network connection. Thus, the proposed APPFL-MEN can deploy the dual weight amendment as a privacy measure and solution to the asynchronous federated learning challenges.

4.4. Asynchronous Federated Learning Dual-Weighted Modification Process

This modification aims to ensure a seamless model training and upload process. To modify the asynchronous nature of the dual weight of both sample and parameter weights, it is deduced that the parameter weight is determined by a node's parameter download time and the corresponding gradient upload time. The sample weight is determined by the fraction of a sample of a node sample to the summed-up sample of all the participating edge nodes [29]. Where participating edge nodes = $N_1, N_2, N_3, \ldots, N_i$, the sample weight is

$$Q_i = N^{-1} * N \tag{19}$$

Furthermore, for the edge server at the MTU, the complete iteration shows that the edge node (IIEN) uploads the gradient parameter (mL) and then downloads the updated parameter (mg) from the server (MTU) as shown in Figure 4. The edge node has records of other nodes' distribution about the time difference between when there is an upload $T'U$ of learned model parameters to the edge server and the equivalent download $T'D$ of parameter gradient. This time lag, T', requires an optimization of the parameter to compensate for the delay and the weakening of the transmission due to the out-of-network coverage as represented by

$$T' = (T'U) - (T'D) \tag{20}$$

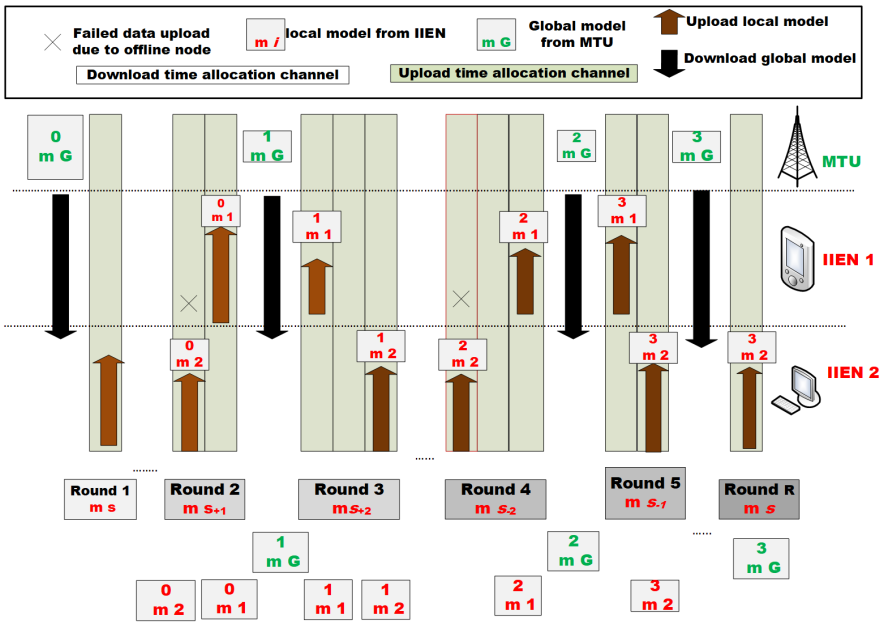

Figure 4. Iteration model design update strategy scheme.

This parameter time lag incidence T' shows the network connectivity strength of the edge node. Hence, to ensure nodes with high T' have very reduced parameter weight and limited signal weakening issues, a reduced function speed is identified as a challenge that weakens the signal strength of the parameter weight [9,26]. This is based on the time lag T' of the edge node and the speed of reduced function q, which is set between 0 and 1. From observation, the q is directly proportional to the result of T'. As q tends to 1, T' increases. For the sake of this research, let $q = 0.2$. This will result in the double-weight modification equation being

$$m' = Q_i N * Q_i(m) \tag{21}$$

where m = initial model parameter, and m'= upgraded model parameter.

Based on the findings about the asynchronous federated learning, the modified global model to be uploaded to the edge server from the edge node as a gradient parameter must first undergo a double-weight modification. This modified gradient updates the global model parameter based on an optimized algorithm, which after the round of optimized iteration, the edge node downloads the up-to-date global parameter and clears off the locally cached parameter to carry out the next round of training.

5. Experimental Analysis and Outputs

This section shows the needed experiments to display the veracity of our proposed privacy-preserving asynchronous federated learning for the IIoT mobile edge system and the dual weight modification and convergence improvement.

5.1. Experimental Settings

We deploy the EdgeCloudSim v4.0 to simulate the edge network-enabled IIoT system, adopting the Hierarchical Edge Computing architecture shown in Figure 2 and employing the basic Python language for testing purposes on the Windows 10 operating system [30]. To evaluate the effectiveness of the proposed method, experiments are performed using the MNIST dataset [31]. The sample scenario is an edge network with dispersed MTU and multi-mobile edge servers and devices distributed within the mobile network coverage. The strongest coverage radius is within 200 m. The MTU, parameter server, and edge nodes are needed to collaborate to train the parameter model using the MNIST dataset which is made up of 600 labeled training and testing parameter samples, which are image digits. To create a non-IID scenario, a random distribution of training samples is needed from the

dataset of about 152 to 951 in quantity to the eight edge devices. The categories of parameter samples range from 1 to 4 in each edge node. The loss function in the training process is equivalent to the entropy loss, with the number of training rounds for the global model set to 4 and the learning rate = 0.9 based on the simulation setting. With an exponential term of 0.1, the distance (m) d is set between MTU and IIEN for network coverage, and an approximate path loss is set between the edge node and MTU to be $121.0 + 32.2log10$. There exist 32 Rbs in the uplink channel with a bandwidth of 140 bit/s, while the downlink channel has a bandwidth of 4.8 bits/s for transmission of the global parameter model to edge nodes. The configuration setting makes it possible to calculate the network latency between the edge node and MTU as shown earlier in Figure 3.

Furthermore, the latency of the edge node's local model training is determined by two conditions: the computational ability of the processing unit and the network's bandwidth. In arriving at the outcome, the local training latency and bandwidth usage of each IIEN were measured using a laptop with an Intel core i7-5580 processor by carrying out the local model training for six iterations where we assume the training time and computational capacity Co of the heterogeneous IIEN to be directly proportional to the parameter sample. To test the efficiency of APPFL-MEN, we check using the Mth $\approx$ 100, a synchronized setting where the MTU collates the model training result in Mth for the IIEN per iteration while implementing the federated learning in heterogeneous nodes and networks. We put in the effort to carry out an analysis and comparison of Mth $\approx$ 100, synchronized setting, and DropPPFL methods, despite the slight difference in system architecture and algorithm for more objectivity under a similar communication scheme.

For the DropPPFL, we use the poly function of $m(r - \tau)$, $a = 0.6$, $\rho = 0.05$, $t - 1 \leq 15$, a better option to adopt in FedAsync. The MTU executes the global parameter update after it receives a local parameter model from the IIEN. However, our proposed APPFL-MEN allows the MTU to receive multiple serial local models compressed from the IIENs within one iteration. When the MTU receives a model other than the programmed default parameter model, there will appear to be a significant variance between the sent global parameter model and the received local parameter model, which increases at an appreciable rate amount. Thus, we propose to ensure that the MTU receives a complete, rightly weighted local parameter model, and its equivalent type accordingly. This is to ensure proper compression, where the compressed weight of the local parameter model and its equivalent type corresponds with the IIEN's communication sample shared in the set of all the IIEN's communication in the present iteration. This will ensure that all local parameter models uploaded as well as global parameter downloads through the communication channels are properly compressed without any collision, adversary attack, and extra communication overload that will arise as a result of IIEN (N_F^{i-1}) going offline.

5.2. Experimental Analysis

The evaluation criteria used include the accuracy of the dual weight modification, compression level, and convergence optimization. The level of accuracy of the modification process is measured by

$$Accuracy = \frac{l}{L} * 100 \tag{22}$$

where l and L are the percentage of the number of completely trained parameter samples and the total number of parameter samples, respectively. This implies that the higher the value of l, the higher the throughput which results in higher accurate model training and classification [32].

Furthermore, since we use the sparsification compression technique which involves zeroing out on communication from offline nodes due to higher level of error and susceptibility to adversary attack, we give communication priority to more online nodes, thereby transmitting only parameters of non-zero gradients or online nodes. This reflects the rate of gradient compression. The lower the compression level, the higher the compression.

Therefore, the compression level of samples communicated through the online node is measured by

$$CL = \frac{\psi}{\varphi} * 100 \tag{23}$$

where ψ is the percentage of the time taken for communication, post-compression, and φ is the time taken for communication pre-compression.

As seen in Table 2, the compression level shows that as the size of the parameter increases, the communication time post-compression increases, therefore the CL decreases and the accuracy decreases as well, due to the longer time taken for gradient compression. However, the different value of Q affects the CL and accuracy with the CL increasing at the interval of 0.3, 0.4. When the interval increases to 0.3, 0.9, the CL increases infinitesimally. Thus, the bigger the value of Q, the bigger the compression level resulting in an increased level of parameter gradient communication. This shows that the accuracy increases as the gradient communication compression level increases. Furthermore, it shows the outcome of the experiments for comparing sample number datasets and the time interval. However, it can be seen that the increase in the compression level is directly proportional to the increase in the model's accuracy. The arbitrary communication has lower gain in the trained model that has attained saturated learning.

Table 2. Compression level comparison result.

	Dataset for Comparision					
CL	**DroPPFL**		**Differential Privacy**		**APPL-MEN**	
w_1, w_2	q	**Accuracy (%)**	q	**Accuracy (%)**	q	**Accuracy (%)**
0.0, 0.1	0.2	85.0	0.2	40.1	0.2	87
0.1, 0.2	0.3	86.0	0.3	43.1	0.3	93
0.2, 0.3	0.4	87.0	0.4	45.7	0.4	95
0.4, 0.5	0.5	89.0	0.5	44.0	0.5	95
0.4, 0.5	0.6	89.3	0.6	45.1	0.6	96
0.6, 0.7	0.7	89.7	0.7	47.3	0.7	96
0.7, 0.8	0.8	90.0	0.8	51.3	0.8	96

5.3. Gradient Compression Level Analysis

In balancing the gradient compression level to obtain the best value for q, a calculation within the interval for $q = [0.2, 0.3]$ shows that the best value for Q is 0.2. In general, in the comparison based on the effectiveness of gradient compression in APPFL-MEN with other algorithms such as by Lu et al., 2020 [7], we evaluate using accuracy, compression level, and gradient compression level. From the findings, the compression level of DropPPFL is lower than APPFL-MEN, $q = 0.2$, and $q = 0.22$, respectively. Conversely, the proposed method outperformed DropPPFL in accuracy using the trained model dataset. We further, compare our method with the DropPPFL by measuring the balance in the gradient compression level, using w_1, w_2 (as shown in Table 2). Our method shows optimum performance, compared with DropPPFL, where the two values' compression level is slightly better.

5.4. Accuracy, Running Time and Bandwidth Usage

Achieving higher accuracy often requires more complex models, larger input data, and longer training times. However, larger input data w_i may also increase computational requirements and resource consumption. Conversely, reducing running time often involves using simpler models and smaller input data, whereas a less compressed model or reducing input size may degrade accuracy, and reducing bandwidth usage typically involves transmitting compressed data. Still, extensive data compression may influence model accuracy while hindering model convergence or update frequency. Thus, in our work, as compared

to DP and DroPPFL, we balance the compression level to reduce the model complexity and input data size without compromising accuracy. Furthermore, we dynamically allocate computational resources and bandwidth based on the data models from online edge nodes of specific tasks and the available resources in the edge network. The trade-off is that our evaluation identifies the optimal balance between accuracy, running time, and bandwidth usage as compared to other methods [33]. The accuracy of the APPFL-MEN, DP, and DropPPFL methods was tested. Results indicate that at the 15th epoch, the trio had an accuracy of 29%, 28%, and 27% respectively. Furthermore, during the 22nd epoch, our method shows better 96.4% of accuracy against the 94% and 93%, respectively, for the other asynchronous methods in Figure 5.

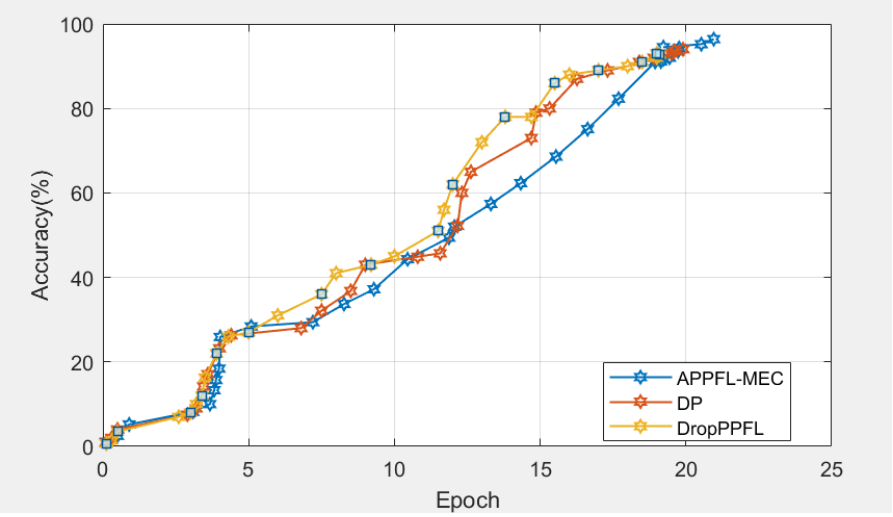

Figure 5. Accuracy of the APPFL-MEN, DP, and DroPPFL

In carrying out the model training as seen in Figure 4 previously, we discovered that the bandwidth usage increased during the different model uploads and downloads, which impacted the model's accuracy. The outcome shows the evolved accuracy during the federated learning round and the bandwidth cost for about 25 rounds. The model configuration used the RNN b-directional using the MNIST dataset aggregated to IIENs. As shown in Figure 6, the running time for offline nodes is reduced to 794 ms, while that of the online nodes is 996 ms which is a good convergence process. For randomly selected IIENs to train models, the size is constant, using the backward regression where an accuracy of about 64% after 24 iterations is achieved using about 0.18 MB, in contrast to an accuracy of 62% using RNN at 0.1 MB, as deduced from Table 3 and shown in Figure 7. This method utilizes more bandwidth as more dataset samples are trained during the upload and subsequent download of the predicted global model. Furthermore, the compression level during upload for connected nodes is relatively high, but the upload of the local model has a minimum error rate and good accuracy.

Table 3. Bandwidth usage.

	Online Nodes		**Offline Nodes**		**Avg (Online + Offline)**	
Iteration	**Accuracy**	**BW**	**Accuracy**	**BW**	**Accuracy**	**BW**
0	67	0.15	28.4	0.12	66	0.135
50	76	0.8	29.4	0.5	72	0.65
100	79	3.4	33.7	0.9	73	2.15

Table 3. *Cont.*

	Online Nodes		Offline Nodes		Avg (Online + Offline)	
Iteration	**Accuracy**	**BW**	**Accuracy**	**BW**	**Accuracy**	**BW**
200	79	3.95	40.1	3.669	76	3.8075
400	80	4.2	45.1	4.023	78	4.115
500	85	4.4	51.3	4.202	86	4.301
700	86	6.1	66.7	5.87	93	5.985
800	89	6.34	70.2	6.308	95	6.324
900	89.7	6	71.4	6.543	96	6.2715
1000	90.3	6.1	70.4	6.689	89.3	6.3945

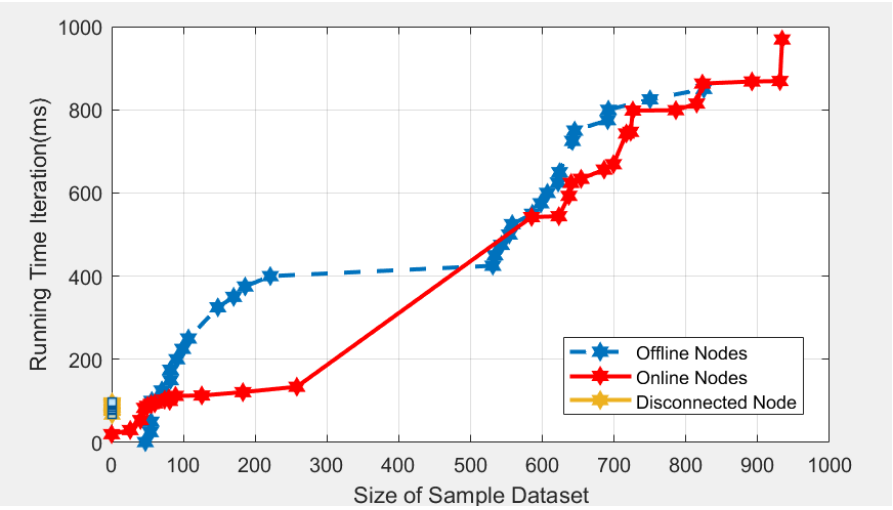

Figure 6. Running time for iteration on dataset by edge devices.

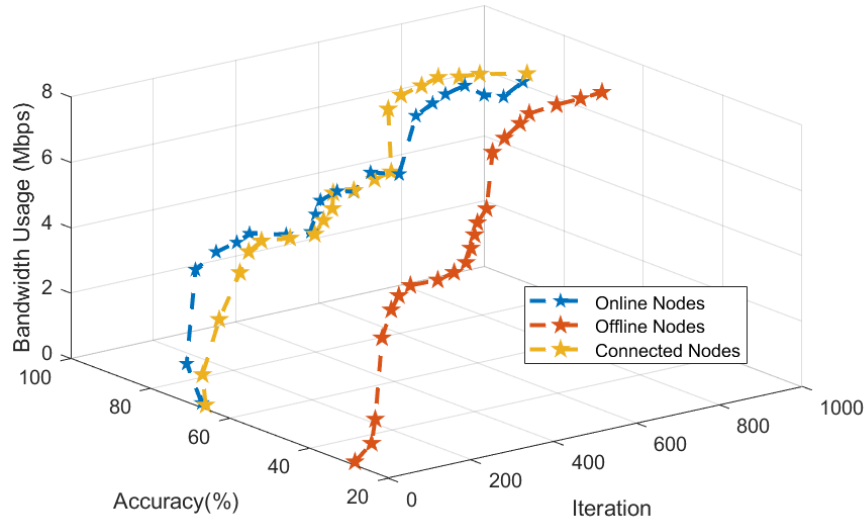

Figure 7. Comparison of bandwidth usage.

5.5. Privacy Accuracy Level

The trade-off between privacy preservation and model accuracy in asynchronous federated learning is that stronger privacy parameters or a more aggressive perturbation method may result in privacy protection, but at the expense of model accuracy. However, this work deploys the compression of models to make them less susceptible to adversary

attacks. Furthermore, the convergence process uses its mechanism to minimize error and any possible loss function that may arise as a result of a dataset generated by offline nodes. Thus, it prioritizes the models from online nodes, thereby preserving the privacy of the model while maintaining the accuracy of training of models [33]. Using the MNIST dataset, we chose the typical synchronous federated learning algorithms of the DP, a privacy protection based on the noise addition, and the offline scheme, DropPPFL, for the evaluation of the privacy protection capacity of our scheme. As can be seen in Figure 8, measuring the values of 0.001, 0.002, 0.1, and 2 where $n = 25$, $R - r = 800$, $\alpha = 0.2$, deduced from Table 3, it is seen that a non-optimized α gives a value of within the acceptable limit of $\approx$ 64%, APPL-MEN = 84%, DropPPFL= 82%, and 29% privacy level, respectively. Comparing our scheme with DP and DropPPFL, it was discovered that offline edge nodes reduce the robustness and slow down the convergence boosting, as well as the accuracy of the trained model, by about 9%. For the $r - r = 400$, we evaluate for accuracy using the range for $\alpha = 0.10, 0.20, 0.30$, where $n = 25$. The result revealed that with an increase in the learning rate α, the convergence enhancement slows down significantly, but with relatively good accuracy. Thus, with an increase in the edge nodes, the training rounds needed for convergence increased.

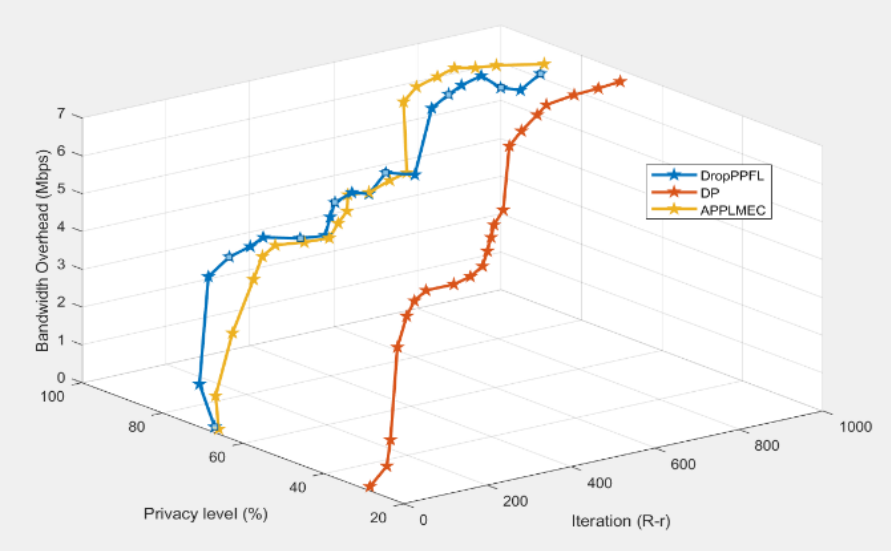

Figure 8. Asynchronous privacy accuracy level.

6. Conclusions

This work proposes an asynchronous privacy-preservation federated learning method in the industrial IoT mobile edge devices to ascertain that the privacy of trained and updated models by edge devices is protected. Enhancing the privacy-preserving capabilities, and ensuring its effectiveness, the double-weighted modification for asynchronous learning that considers the sample size and parameter weight are introduced. We calculate the measure of the node sample size compared to the sum of the sample learning node sample weight, $Q_i = N^{-1} * N$, $m = Q_i N * Q_i(m)$, as well as the time lag between gradient upload/parameter download during the model training, $T^{'} = (T^{'}U) - (T^{'}D)$, respectively. This shows the level of staleness and uploads while the offline nodes will have little participation to prevent an adversary attack on the dataset. We introduce the convergence enhancement process to allow the increase in good models uploaded by online edge nodes while reducing the error-prone model from being uploaded by the offline nodes. We leverage the IMDUS to guarantee the integrity of each trained model to produce the correct value of the less error-prone updated model. Individual edge node in the IMDUS represents an updated version of the model having the traits of the verified value of minimum error, compressed weight, and output model M.

Author Contributions: Conceptualization, J.O.O. and X.Y.; methodology, all authors contributed equally; software, J.O.O. and X.Y.; validation, J.O.O. and X.Y.; formal analysis, J.O.O.; investigation, all authors contributed equally; data curation, J.O.O.; writing—original draft preparation, J.O.O. and X.Y.; writing—review and editing, J.O.O., X.Y., C.I.N. and S.D.; visualization, J.O.O., X.Y. and C.I.N.; supervision, X.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Natural Science Foundation of China (NSFC) under Grants 61971033 and 61941113.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: This manuscript has not been published or presented elsewhere in part or entirety and is not under consideration by another journal. There are no conflicts of interest to declare.

Abbreviations

The following abbreviations are used in this manuscript:

Notation	Meaning
CL	compression level
CNN	Convolutional Neural Network
BW	bandwidth
D	trained dataset
d_i	local dataset
DP	Differential Privacy
DP-AFL	Differentially Private Asynchronous FL
FL	Federated Learning
$F(s)$	Loss function
HE	Homomorphic Encryption
IMDUS	Iteration Model Design Update Strategy
IIEN	Industrial Internet edge node
IoT	Internet of Things
MEC	Mobile Edge Computing
MED	multi-edge device
MEN	Mobile Edge Network
MTU	mobile transmission unit
m_g	updated parameter
m_L	gradient parameter
n	set of offline edge nodes
Q	sample weight
q	function speed
R	highest updating iteration
r	iteration
RNN	Recurrent Neural Network
s_i	compressed parameter model
w_i	input of the ML model
X	MTU
Y	edge node
z_i	trained model output
N_R^{i-1}	online nodes
N_F^{i-1}	offline nodes
N^{i-1}	all sets of edge nodes with parameter weight

References

1. Shao, Z.; Zhao, R.; Yuan, S.; Ding, M.; Wang, Y. Tracing the evolution of AI in the past decade and forecasting the emerging trends. *Expert Syst. Appl.* **2022**, *209*, 118221. [CrossRef]
2. Al-Quraan, M.; Mohjazi, L.; Bariah, L.; Centeno, A.; Zoha, A.; Arshad, K.; Assaleh, K.; Muhaidat, S.; Debbah, M.; Imran, M.A. Edge-Native Intelligence for 6G Communications Driven by Federated Learning: A Survey of Trends and Challenges. *IEEE Trans. Emerg. Top. Comput. Intell.* **2023**, *7*, 957–979. [CrossRef]

3. Vailshery, L.S. *Number of Internet of Things (IoT) Connected Devices Worldwide from 2019 to 2023, with Forecasts from 2022 to 2030*; Statista: Hamburg, Germany, 2023. Available online: http://xxx.lanl.gov/abs/https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/ (accessed on 27 July 2023).
4. Melis, L.; Song, C.; De Cristofaro, E.; Shmatikov, V. Exploiting Unintended Feature Leakage in Collaborative Learning. In Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 19–23 May 2019; pp. 691–706. [CrossRef]
5. Wang, R.; Lai, J.; Zhang, Z.; Li, X.; Vijayakumar, P.; Karuppiah, M. Privacy-Preserving Federated Learning for Internet of Medical Things Under Edge Computing. *IEEE J. Biomed. Health Inform.* **2023**, *27*, 854–865. [CrossRef] [PubMed]
6. Ksentini, A.; Frangoudis, P.A. On Extending ETSI MEC to Support LoRa for Efficient IoT Application Deployment at the Edge. *IEEE Commun. Stand. Mag.* **2020**, *4*, 57–63. [CrossRef]
7. Lu, Y.; Huang, X.; Dai, Y.; Maharjan, S.; Zhang, Y. Differentially Private Asynchronous Federated Learning for Mobile Edge Computing in Urban Informatics. *IEEE Trans. Ind. Inform.* **2020**, *16*, 2134–2143. [CrossRef]
8. 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]
9. Yan, X.; Miao, Y.; Li, X.; Choo, K.K.R.; Meng, X.; Deng, R.H. Privacy-Preserving Asynchronous Federated Learning Framework in Distributed IoT. *IEEE Internet Things J.* **2023**, *10*, 13281–13291. [CrossRef]
10. Qolomany, B.; Ahmad, K.; Al-Fuqaha, A.; Qadir, J. Particle Swarm Optimized Federated Learning For Industrial IoT and Smart City Services. In Proceedings of the GLOBECOM 2020–2020 IEEE Global Communications Conference, Taipei, Taiwan, 7–11 December 2020; pp. 1–6. [CrossRef]
11. James Singh, K.; Huang, Y.M.; Ahmed, T.; Liu, A.C.; Huang Chen, S.W.; Liou, F.J.; Wu, T.; Lin, C.C.; Chow, C.W.; Lin, G.R.; et al. Micro-LED as a Promising Candidate for High-Speed Visible Light Communication. *Appl. Sci.* **2020**, *10*, 7384. [CrossRef]
12. Chi, N.; Zhou, Y.; Wei, Y.; Hu, F. Visible Light Communication in 6G: Advances, Challenges, and Prospects. *IEEE Veh. Technol. Mag.* **2020**, *15*, 93–102. [CrossRef]
13. McMahan, H.B.; Ramage, D.; Talwar, K.; Zhang, L. Learning Differentially Private Recurrent Language Models. *arXiv* **2018**, arXiv:1710.06963. [CrossRef]
14. Bonawitz, K.; Ivanov, V.; Kreuter, B.; Marcedone, A.; McMahan, H.B.; Patel, S.; Ramage, D.; Segal, A.; Seth, K. Practical secure aggregation for privacy-preserving machine learning. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, Dallas, TX, USA, 30 October–3 November 2017; pp. 1175–1191.
15. Zhao, C.; Zhao, S.; Zhao, M.; Chen, Z.; Gao, C.Z.; Li, H.; an Tan, Y. Secure Multi-Party Computation: Theory, practice and applications. *Inf. Sci.* **2019**, *476*, 357–372. [CrossRef]
16. Sun, Y.; Uysal-Biyikoglu, E.; Yates, R.D.; Koksal, C.E.; Shroff, N.B. Update or Wait: How to Keep Your Data Fresh. *IEEE Trans. Inf. Theory* **2017**, *63*, 7492–7508. [CrossRef]
17. Zhang, C.; Li, S.; Xia, J.; Wang, W.; Yan, F.; Liu, Y. {BatchCrypt}: Efficient homomorphic encryption for {Cross-Silo} federated learning. In Proceedings of the 2020 USENIX Annual Technical Conference (USENIX ATC 20), Online, 15–17 July 2020; pp. 493–506.
18. Wang, Y.; Su, Z.; Zhang, N.; Benslimane, A. Learning in the Air: Secure Federated Learning for UAV-Assisted Crowdsensing. *IEEE Trans. Netw. Sci. Eng.* **2021**, *8*, 1055–1069. [CrossRef]
19. Yu, Z.; Hu, J.; Min, G.; Lu, H.; Zhao, Z.; Wang, H.; Georgalas, N. Federated Learning Based Proactive Content Caching in Edge Computing. In Proceedings of the 2018 IEEE Global Communications Conference (GLOBECOM), Abu Dhabi, United Arab Emirates, 9–13 December 2018; pp. 1–6. [CrossRef]
20. Giovanelli, C.; Kilkki, O.; Sierla, S.; Seilonen, I.; Vyatkin, V. Task Allocation Algorithm for Energy Resources Providing Frequency Containment Reserves. *IEEE Trans. Ind. Inform.* **2019**, *15*, 677–688. [CrossRef]
21. Coutinho, R.W.L.; Boukerche, A. Modeling and Analysis of a Shared Edge Caching System for Connected Cars and Industrial IoT-Based Applications. *IEEE Trans. Ind. Inform.* **2020**, *16*, 2003–2012. [CrossRef]
22. Xie, C.; Koyejo, S.; Gupta, I. Asynchronous Federated Optimization. *arXiv* **2020**, arXiv:1903.03934. [CrossRef]
23. Wang, Z.; Zhang, Z.; Wang, J. Asynchronous Federated Learning over Wireless Communication Networks. In Proceedings of the ICC 2021—IEEE International Conference on Communications, Montreal, QC, Canada, 14–23 June 2021; pp. 1–7. [CrossRef]
24. Nowak, T.W.; Sepczuk, M.; Kotulski, Z.; Niewolski, W.; Artych, R.; Bocianiak, K.; Osko, T.; Wary, J.P. Verticals in 5G MEC-Use Cases and Security Challenges. *IEEE Access* **2021**, *9*, 87251–87298. [CrossRef]
25. Froehlich, A.; Ferguson, K. Bandwidth (network bandwidth). In *TechTarget*; West Gate Networks: Chicago, IL, USA, 2021. Available online: http://xxx.lanl.gov/abs/https://www.techtarget.com/searchnetworking/definition/bandwidth (accessed on 15 April 2024).
26. Mengistu, T.M.; Kim, T.; Lin, J.W. A Survey on Heterogeneity Taxonomy, Security and Privacy Preservation in the Integration of IoT, Wireless Sensor Networks and Federated Learning. *Sensors* **2024**, *24*, 968. [CrossRef] [PubMed]
27. Mahbub, M.; Shubair, R.M. Contemporary advances in multi-access edge computing: A survey of fundamentals, architecture, technologies, deployment cases, security, challenges, and directions. *J. Netw. Comput. Appl.* **2023**, *219*, 103726. [CrossRef]
28. Mudassar, M.; Zhai, Y.; Lejian, L. Adaptive Fault-Tolerant Strategy for Latency-Aware IoT Application Executing in Edge Computing Environment. *IEEE Internet Things J.* **2022**, *9*, 13250–13262. [CrossRef]

29. Thantharate, P.; Anurag, T. CYBRIA—Pioneering Federated Learning for Privacy-Aware Cybersecurity with Brilliance. In Proceedings of the 2023 IEEE 20th International Conference on Smart Communities: Improving Quality of Life Using AI, Robotics and IoT (HONET), Boca Raton, FL, USA, 4–6 December 2023; pp. 56–61. [CrossRef]
30. Sonmez, C.; Ozgovde, A.; Ersoy, C. EdgeCloudSim: An environment for performance evaluation of Edge Computing systems. In Proceedings of the 2017 Second International Conference on Fog and Mobile Edge Computing (FMEC), Valencia, Spain, 8–11 May 2017; pp. 39–44. [CrossRef]
31. LeCun, Y.; Cortes, C.; Burges, C. *MNIST Handwritten Digit Database*; ATT Labs: Atlanta, GA, USA, 2010; Volume 2. Available online: http://yann.lecun.com/exdb/mnist (accessed on 15 April 2024).
32. Jiang, C.; Li, Y.; Su, J.; Chen, Q. Research on new edge computing network architecture and task offloading strategy for Internet of Things. *Wirel. Netw.* **2021**, 1–13. [CrossRef]
33. Zhao, S.; Zhou, L.; Wang, W.; Cai, D.; Lam, T.L.; Xu, Y. Toward Better Accuracy-Efficiency Trade-Offs: Divide and Co-Training. *IEEE Trans. Image Process.* **2022**, *31*, 5869–5880. [CrossRef] [PubMed]