Contents lists available at ScienceDirect

Digital Communications and Networks

journal homepage: www.keaipublishing.com/dcan

Towards asynchronous federated learning for heterogeneous edge-powered internet of things

Zheyi Chen [a], Weixian Liao [a,*], Kun Hua [b], Chao Lu [a], Wei Yu [a]

[a] Department of Computer and Information Sciences, Towson University, Towson, MD, 21252, USA
[b] Department of Electrical and Computer Engineering, Lawrence Technological University, Southfield, MI, 48075, USA

ARTICLE INFO

Keywords:
Asynchronous federated learning
Internet of Things (IoT)
Mobile edge computing

ABSTRACT

The advancement of the Internet of Things (IoT) brings new opportunities for collecting real-time data and deploying machine learning models. Nonetheless, an individual IoT device may not have adequate computing resources to train and deploy an entire learning model. At the same time, transmitting continuous real-time data to a central server with high computing resource incurs enormous communication costs and raises issues in data security and privacy. Federated learning, a distributed machine learning framework, is a promising solution to train machine learning models with resource-limited devices and edge servers. Yet, the majority of existing works assume an impractically synchronous parameter update manner with homogeneous IoT nodes under stable communication connections. In this paper, we develop an asynchronous federated learning scheme to improve training efficiency for heterogeneous IoT devices under unstable communication network. Particularly, we formulate an asynchronous federated learning model and develop a lightweight node selection algorithm to carry out learning tasks effectively. The proposed algorithm iteratively selects heterogeneous IoT nodes to participate in the global learning aggregation while considering their local computing resource and communication condition. Extensive experimental results demonstrate that our proposed asynchronous federated learning scheme outperforms the state-of-the-art schemes in various settings on independent and identically distributed (i.i.d.) and non-i.i.d. data distribution.

1. Introduction

Machine learning techniques play a key role in discovering knowledge and providing actionable intelligence from a vast amount of data, advancing the state-of-the-art in a number of areas, including autonomous vehicles, natural language processing, network intrusion detection, email spam filtering, financial market analysis, and computer vision, among others [1–4]. With the recent advances in IoT, the capacity to collect and act upon massively distributed real-time data is greatly increasing. Nonetheless, it is still generally infeasible for an individual IoT device to carry out the entire training process for machine learning on its own. While conventional machine learning processes require a computationally extensive and centralized training environment, it is impractical to send data collected by a massive number of IoT devices to a remote cloud server, which may overload the network and incur unacceptable latency. Equally pressing, security and privacy issues arise in both IoT and machine learning, the vulnerabilities of which have not been fully understood and mitigated [5–12]. For example, critical applications for improving medical diagnostics and predicting disease risk involve personal healthcare records, which are highly sensitive and private and may cause serious damage to individuals if revealed [13,14].

The edge computing paradigm can assist in the reduction of network congestion by migrating computing tasks from the centralized cloud to network edges. It enables IoT devices to participate in computing tasks with their local data [15,16]. Considering IoT as a network infrastructure to support a variety of distributed smart-world applications, IoT nodes have the capacity to collect and process data and communicate with other nodes [5,17]. As shown in Fig. 1, the investigated system in this paper is subdivided into three layers, including a remote cloud, a set of edge computing nodes and heterogeneous IoT devices. The remote cloud provides abundant computing resources, but it is far away from local networks. By contrast, edge servers interact with IoT devices at the edge of the network to reduce the computing overhead for the cloud and the communication latency for IoT devices [18]. In addition, computing

* Corresponding author.
E-mail addresses: zchen12@students.towson.edu (Z. Chen), wliao@towson.edu (W. Liao), khua@ltu.edu (K. Hua), clu@towson.edu (C. Lu), wyu@towson.edu (W. Yu).

https://doi.org/10.1016/j.dcan.2021.04.001
Received 3 August 2020; Received in revised form 28 March 2021; Accepted 2 April 2021
Available online 20 April 2021
2352-8648/© 2021 Chongqing University of Posts and Telecommunications. Publishing Services by Elsevier B.V. on behalf of KeAi Communications Co. Ltd. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

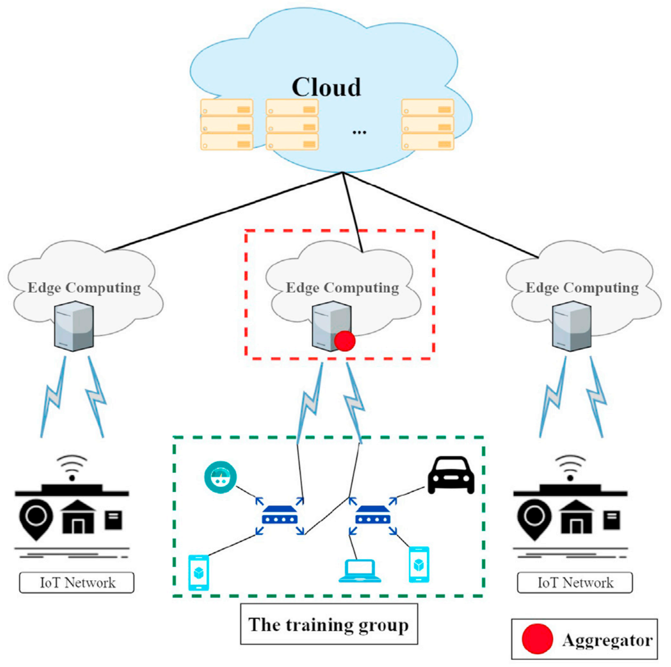


Fig. 1. System architecture for federated learning with heterogeneous edge-powered IoT system.

resources provided by edge servers can assist in processing the data collected from the IoT devices. Edge computing has been broadly studied from a variety of perspectives, including mobile edge computing resources scheduling, computation offloading, and joint radio-and-computational resource allocation, among others [19,22]. For example, Mohammad et al. [23] focused on scheduling machine learning tasks on edge computing by maximizing the number of local learning iterations in locally distributed learners to improve the learning performance. Nonetheless, it is worth noting that transmitting local data to edge nodes incurs tremendous overhead in wireless networks with limited bandwidth and unstable communications, and raises data privacy issues.

Federated learning, as a distributed learning framework, exploits local computing resources and data in distributed devices to collaboratively train a machine learning model [24,25]. Specifically, in the federated learning process, aggregators are deployed at edge servers, working with individual nodes as local learners that train the same machine learning model using its locally available data. In each round, the aggregator receives and aggregates the weight vectors from local learners. Then, the aggregator tests the performance (i.e., accuracy) of the learned model with validation data. The essence of the federated learning process is that it avoids sharing local data in the network. McMahan et al. [24] proposed the federated learning framework, developed an algorithm for weight aggregation and conducted experiments with various datasets. In their study, a fraction of nodes are randomly selected to carry out the global aggregation in each round.

Existing schemes have been primarily developed with the assumption that IoT devices are homogeneous and communicate with each other in a static network. Thus, the synchronous aggregation has been assumed, which yields inefficient weight updates in IoT environments [26,27]. To be specific, the aggregator at the edge server has to wait for weight vectors from all nodes before determining the aggregated weight vector. The slowest node, therefore, becomes the bottleneck and delays the entire training process. These assumptions do not hold for real-world applications due to reasons listed as follows: (i) *In real-world applications, IoT devices are diverse and have heterogeneous computing resources (e.g., smart phones versus smart watches)*, (ii) *the communication conditions are dynamic and the throughput cannot be guaranteed under an unstable wireless network*. Therefore, in order to implement machine learning training processes on heterogeneous IoT devices with the aid of edge computing, the following challenge needs to be considered: *How can we simultaneously handle heterogeneous computing resources and dynamic and unstable networking communications?*

To address this challenge, we have made the following contributions in the paper.

- **A new asynchronous federated learning scheme:** We propose an asynchronous federated learning scheme with diverse resources, including the local data sample (on-device data), communication costs and the computation cost of the local learning model. To be specific, we first design an asynchronous federated learning model by formulating a 0–1 Knapsack optimization problem to characterize the heterogeneous computing resources of IoT devices in an unstable network. We then develop an asynchronous node selection scheme to leverage available computing resources on each node constrained by its communication condition so that the learning process can be completed more efficiently than in the existing methods that assume synchronous updates. The local computing resources are qualified by the number of epochs and batch size in each iteration, in which the batch size given by the parameter server is a constant over the entire training process. Additionally, our proposed scheme has low computing complexity via using a greedy based method. Note that all these resources are common to reach out on the IoT devices.
- **Extensive performance validation:** To validate the efficacy of our scheme, we carry out experiments on several datasets (MNIST 10-digit dataset, Fashion-MNIST dataset and EMNIST dataset) and compare our scheme with the two state-of-the-art schemes with respect to various performance metrics, including overall accuracy, time of arrival at the desired accuracy, and average communication overhead. We use both non-i.i.d. data and i.i.d. data in experiments because data samples collected by different users (nodes) and heterogeneous devices are assumed to collect data in different distributions [24,28]. Experimental results show that our proposed asynchronous federated learning scheme outperforms other baseline schemes under different settings.

The remainder of this paper is organized as follows. In Section 2, we discuss related research works and delineate our proposed scheme from existing methods. In Section 3, we define the problem space, introduce the asynchronous federated learning model and propose a heuristic greedy node selection scheme. In Section 4, we present the performance evaluation of our scheme, which compares its efficacy with two representative schemes. In Section 5, we discuss some open problems relevant to our study. Finally, we conclude the paper with some final remarks in Section 6.

2. Related work

Previous research efforts have made a number of contributions to the development of edge computing and IoT technologies, but there are some still critical issues.

There are a number of efforts on applying machine learning techniques to enhance IoT/CPS performance and security [1,29–31]. For example, Hatcher and Yu [1] reviewed the existing applications of deep learning techniques and discussed some emergent applications of deep learning, such as network management, IoT systems, and others. With the support of the edge computing paradigm, federated learning becomes a viable approach in distributed IoT systems by enabling a set of nodes to train a machine learning model collaboratively using their local data while only sharing intermediate data.

The paradigm of edge computing has shown great potential in a number of areas, including task scheduling [20], computation migration [19,32], and resource allocation [21], among others. For example, Huang et al. [20] studied a task scheduling problem in a cloud-edge learning system and proposed an extreme point-based scheduling scheme to solve the linear programming relaxation problem. Our work differs from existing research efforts as we focus on a more generic IoT environment, in which IoT devices are heterogeneous and operate under unstable network conditions.

The centralized data collection for machine learning model training raises some privacy concerns, and individual IoT devices generally cannot train a machine learning model by themselves due to limitations in resources (e.g., data storage and computing capacity). Therefore, leveraging edge computing to carry out distributed learning on a group of nodes is a feasible method in IoT networks. Nonetheless, leveraging edge computing with distributed learning infrastructures for effective machine learning training remains a challenge. For example, Wang et al. [27] investigated synchronous distributed machine learning and proposed an algorithm to set up a flexible global aggregation frequency that is subject to available resources. Likewise, Zhao et al. [28] proposed a distributed scheme to improve the performance of the learning task by sharing local training data. Nonetheless, in some data privacy-sensitive applications, IoT devices may not be willing to share local data with others. Moreover, there are some impractical assumptions in synchronous schemes. First, all nodes are required to send weight vectors in each training round, leading to traffic load peaking in the network. Second, waiting for the slowest nodes in these schemes can incur a very long delay.

To overcome these challenges, there have been some efforts toward optimizing asynchronous federated learning [33,34]. For example, Lian et al. [33] developed an algorithm to speed up the convergence of the machine learning task without a central parameter server. Likewise, Nishio et al. [34] provided an asynchronous node selection scheme for global aggregation by solving the optimization problem on the networking communication aspect. Nonetheless, their schemes did not account for heterogeneous computing capacities of nodes. By contrast, our proposed asynchronous federated learning scheme considers the local computing resources on the IoT devices and real-time communication conditions between the aggregator and IoT devices simultaneously.

To summarize, in this paper, we develop a new asynchronous federated learning scheme, in which edge servers work as parameter aggregators and a group of IoT devices train the same model with their unique local data. The proposed asynchronous node selection scheme leverages real-time available computing resources and networking communication conditions so that the learning process is accelerated to its convergence.

3. An asynchronous federated learning framework

In this section, we introduce our scheme in detail. Particularly, we first introduce the system architecture and the design rationale. Then, we introduce our proposed asynchronous federated learning model. Finally, we describe our proposed asynchronous algorithm to carry out the node selection for distributed learning in the IoT network.

3.1. Problem space and design rationale

Fig. 2 lists a number of potential issues in the machine learning field, including distributed machine learning and centralized machine learning. Here, we consider two orthogonal dimensions. In the horizontal dimension, the operation of machine learning schemes can be categorized in either a centralized or distributed manner. In the vertical dimension, the data for machine learning schemes can be categorized in either i.i.d or non-i.i.d.

Conventional machine learning schemes, which aggregate all training data at a centralized server with high computing ability, cannot fulfill the challenges and requirements for IoT systems. We consider that the federated learning [24] is a promising solution to bridge IoT systems and the machine learning. First, IoT devices could collect data samples and conduct the training process on its local side. Secondly, local learners can keep their data locally to reduce the security risk as well as save the

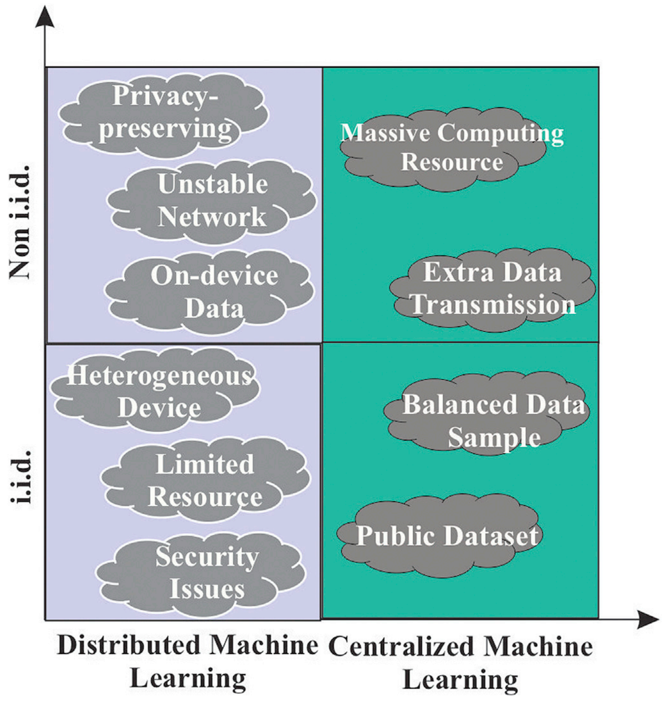


Fig. 2. Problem space of machine learning.

expensive wireless bandwidth. Moreover, the network topology is similar between IoT systems and the federated learning framework. As shown in Fig. 1, we deploy the federated learning system on edge computing nodes connected via an IoT network. The edge node is the central parameter server (i.e., aggregator), and a number of IoT devices train the same learning model collaboratively. IoT devices can collect raw data from their operation scenarios, in which most data samples could follow non-i.i.d. distribution.

Federated learning is a distributed learning framework that exploits the distributed computing ability on distributed nodes. A machine learning model can be trained together as distributed nodes train the same model with their local data. Nonetheless, most of existing federated learning efforts assume a homogeneous system, in which all nodes have the same computing power in a stable communication network and follow a synchronous learning fashion.

Nonetheless, in the real world, IoT devices are heterogeneous, such as sensors, tablets, smart phones and smart watches, etc., making synchronous learning inefficient. Furthermore, heterogeneous computing capacity and dynamic communication conditions of IoT devices make the limited time resource wasted to wait for the slowest node for weight vector updates. To address this issue, we design an asynchronous federated learning model, in which the computing resource is opportunistically captured, and nodes are dynamically selected for global aggregation so that the performance of the assigned learning task can be improved.

3.2. Federated learning model

In the federated learning process, the investigated scenario is composed of multiple IoT devices and an aggregator (edge server).[1] These training nodes are assigned to train the same machine learning model. Table 1 lists the key notations in the paper.

We denote IoT devices in the node set $I = \{1, 2, \ldots, n, \ldots, N\}$ and $\mathscr{D}_n = \{d_1, d_2, \ldots, d_j, \ldots, d_{D_n}\}$ as the dataset on node n, in which D_n represents the number of data samples over local dataset $\mathscr{D}_n$. The goal of the original federated learning is to obtain a weight vector $\mathbf{w}_G \in \mathscr{R}^k$, which

[1] In this paper, the edge server and aggregator are interchangeable.

Table 1
Notations.

I	The set of nodes
N	The number of devices in the network
$\mathbf{w}$	The weight vector of the machine learning model
$F()$	The loss function
$\mathscr{D}_n$	The dataset on node n
D_n	The number of data samples on node n
P	The selected nodes in global aggregation
T	The local training period
λ_n	The number of the local update on the node n
b_n	Communication rate on node n

minimizes the global loss function $F_G()$ over the training process. Here, k represents the dimensionality of the weight vector. The learning problem in federated learning process can be written as:

$$\mathbf{w}_G^* = \text{argmin} \quad F_G(\mathbf{w}_G) \tag{1}$$

We then explain the process of generating a local update. To be specific, at each iteration t, a number of IoT nodes are selected to participate in the global aggregation. Each node n trains the received global weight vector $\mathbf{w}_G^{t-1}$ with E_n epochs to minimize the local loss function $F_n()$ over its own dataset $\mathscr{D}_n$. At the end of the iteration t, each node has a local weight vector $\mathbf{w}_n^t$. Then, the local update $\boldsymbol{\delta}_n^t$ is sent back to the central parameter server, where $\boldsymbol{\delta}_n^t$ is computed as:

$$\boldsymbol{\delta}_n^t = \mathbf{w}_n^t - \mathbf{w}_G^{t-1} \tag{2}$$

On the central parameter server side, all received local updates are aggregated based on the weight averaging method, which can be expressed as:

$$\mathbf{w}_G^t = \mathbf{w}_G^{t-1} + \sum_{n=1}^{P^t} \frac{D_n}{D_{P^t}} \boldsymbol{\delta}_n^t \tag{3}$$

Note that P^t ($P^t \subseteq I$) is a set of nodes representing the nodes that dynamically participate in the global aggregation at round t. The formulated problem (Equation (1) - Equation (3)) covers the synchronous update scenario. Particularly, when all nodes in the node set I participate in the global aggregation, the update process is synchronous, i.e., $P^t = I$ [27].

It remains an open problem of directly estimating the computing resource (i.e., CPU clock, memory availability) of federated learning tasks. In this sense, we should first discuss the computing cost in the training process. We assume the computational complexity of the machine learning model in each iteration is a constant [23,27]. For a local federated learning task, the computing contribution λ_n on the machine learning task is considered to be related to the number of local updates E_n and the minibatch size used in each update. It is assumed that the size of minibatch (hyperparameter) is determined by the parameter server (i.e., aggregator) and keeps unchanged in the federated learning process. In iteration t, we can evaluate λ_n^t by the product of the number of updates E_n^t and the minibatch size d_s. Then, we have

$$\lambda_n^t = E_n^t * d_s \tag{4}$$

Note that we empirically set up the size of the minibatch for the learning task in our experiments. We denote $M_T = \{\lambda_1, \lambda_2, \ldots, \lambda_n, \ldots, \lambda_N\}$ as the set of local computing contributions over the federated learning process. Moreover, we consider that set M_T is accumulated. The value of λ_n will be set to 0 if node n is selected to participate in the last global aggregation. Therefore, we have

$$\lambda_n = \begin{cases} \lambda_n + \lambda_n^t, & s_n^t = 1 \\ 0, & s_n^t = 0 \end{cases} \tag{5}$$

where $s_n^t = 1$ if node n is selected to participate in the global aggregation; otherwise, we have $s_n^t = 0$. To model the dynamic communication network in the IoT environment, communication cost for aggregation at each round is abstracted by the throughput of weight vector transmission between nodes and the aggregator, i.e., $B = \{b_1^t, b_2^t, \ldots, b_n^t, \ldots, b_N^t\}$.

The learning problem in the federated learning model is to find the optimal weight vector that minimizes the loss function in Equation (1). Due to the inherent complexity of most machine learning models, it is often impossible to find a closed-form solution. More precisely, it is worth noting that it is difficult to find a closed formula to associate the aggregation schemes with the entire training process because it depends on the convergence property of the gradient descent algorithm, from which we can only know the upper and lower bounds of the optimal solution [27]. Therefore, Equation (1) is often solved by gradient descent-based techniques.

3.3. The asynchronous federated learning scheme

We now introduce the detail of the asynchronous federated learning scheme, which consists of the following three phases:

- **Phase I: Initialization.** The server, known as the parameter aggregator, initializes a global machine learning model and the initial random parameter vector $w_{initial}$. It establishes node set I. Then, the aggregator distributes the weight vector $w_{initial}$ to node set I.
- **Phase II: Aggregation.** Each node in set I sends an ACK frame to the aggregator and evaluates the current communication condition (b_n^t) and computing contribution (λ_n). Then, a proposed node selection algorithm, which is introduced in the following section, selects nodes for set P^t to take the global aggregation at each iteration t, using the weight averaging method [24].
- **Phase III: Parameter update.** The aggregator measures the performance A (i.e., accuracy) of the model with aggregated parameters by a validation dataset, and then sends the aggregated parameter vector to nodes set in P. In addition, Phases II and III are repeated until the model performance exceeds the predefined threshold or the training time runs out.

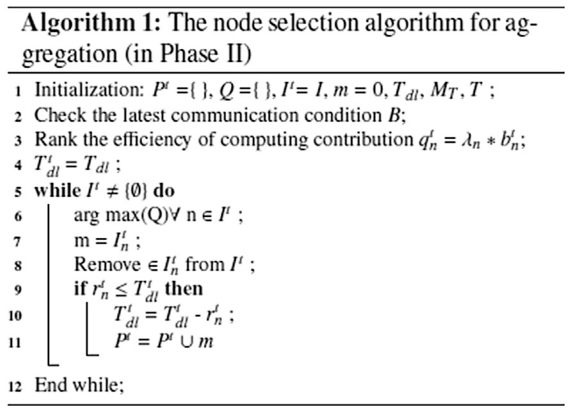
3.4. Node selection algorithm

We now develop an algorithm for Phase II to determine which nodes should participate in the aggregation in the given time T_{dl}, where T_{dl} represents a pre-defined time used to collect node updates. The proposed algorithm aims to capture the computing contribution as much as possible with the given communication time (T_{dl}) to improve the performance of the machine learning task. Thus, the node selection problem at iteration t is formulated as:

$$\begin{aligned} &\max_{s_n^t, n \in I} \quad \sum_{n=1}^{N} \lambda_n s_n^t \\ &s.t. \begin{cases} \displaystyle\sum_{n=1}^{N} r_n^t s_n^t \le T_{dl}, & (n = 1, 2, \ldots, N) \\ s_n^t \in \{0, 1\}, & (n = 1, 2, \ldots, N) \end{cases} \end{aligned} \tag{6}$$

Due to the heterogeneity of IoT devices, the communication conditions in set B are used to evaluate the transmission time $r_n^t = \frac{l}{b_n^t}$, where l represents the package size of the weight vector.

Equation (6) is a known $0-1$ *Knapsack* problem [35]. The proposed algorithm needs to deal with an IoT environment with a large number of nodes. To balance the optimal solution and the operation efficiency, a heuristic algorithm based on the greedy method is designed to obtain the approximate solution of the $0-1$ *Knapsack* problem.



Algorithm 1: The node selection algorithm for aggregation (in Phase II)

```
1  Initialization: $P^t$ ={ }, $Q$={ }, $I^t$= $I$, $m$ = 0, $T_{dl}$, $M_T$, $T$ ;
2  Check the latest communication condition $B$;
3  Rank the efficiency of computing contribution $q_n^t = \lambda_n * b_n^t$;
4  $T_{dl}^t = T_{dl}$ ;
5  while $I^t \neq \{\emptyset\}$ do
6      arg max(Q)$\forall$ n $\in I^t$ ;
7      m = $I_n^t$ ;
8      Remove $\in I_n^t$ from $I^t$ ;
9      if $r_n^t \leq T_{dl}^t$ then
10         $T_{dl}^t = T_{dl}^t - r_n^t$ ;
11         $P^t = P^t \cup m$
12 End while;
```

Note that r_n^t represents the time cost to collect the weight vector on node n. After the aggregator executes Algorithm 1, the weight vector is transmitted to nodes in set P^t. The proposed algorithm consists of two parts. The aggregator first collects λ_n for all nodes and evaluates the current communication condition (b_n^t) by sending an ACK frame to each IoT device. Then, the aggregator uses mergesort to rank each IoT device by the efficiency of computing contribution (Q), where Q is determined by the product of computing contribution and the throughput. It is assumed that the actual size of the weight vector is the same on each device. Thus, the complexity is $O(N\log(N))$, where N represents the number of nodes. From lines 5 to 11, Algorithm 1 aims to select candidates to take the global aggregation, where the complexity of this part is $O(N)$. As a result, the complexity of Algorithm 1 is $O(N\log(N))$, where N is the total number of nodes. Note that we find that the proposed algorithm could lead to the condition where a number of subsets will be selected alternatively, but it does not conflict with the original aim of this paper. This is because we aim to solve an asynchronous federated learning problem under unstable communication conditions, which are dynamically changing in each iteration. To verify these real-world scenarios, we set different communication conditions in our experiments by sampling the throughput over Gaussian distribution.

4. Performance evaluation

We carry out experiments to validate the efficacy of our proposed asynchronous federated learning scheme, using three datasets (i.e., MNIST 10-digit, Fashion-MNIST and EMNIST [24,34,36]). In the following, we first introduce the evaluation methodology and then present evaluation results by comparing the two state-of-the-art schemes in different settings.

4.1. Evaluation methodology

We consider the scenario depicted in Fig. 1, where the edge server acts as the aggregator and $N = 100$ IoT devices in our experiments. To configure the communication parameters, links between the aggregator and nodes follow the IEEE 802.11 protocols [37,38]. Finding the most representative communication protocol of heterogeneous IoT devices is beyond the scope of this study. We consider a single-in-single-out antenna on the aggregator sever, and a multiple-in-multiple-out case will be further studied as our future work. In this study, three public datasets are used to validate the performance of the proposed asynchronous federated learning scheme. MNIST 10-digit and Fashion-MNIST consist of a training set of 60, 000 examples and a testing set of 10,000 examples. Similar to Refs. [24, 34], we randomly shuffle the total 60,000 examples before we allocate the data sample to each node. Note that the number of data samples on each node ranges from 200 to 1000. (i.i.d. data distribution).

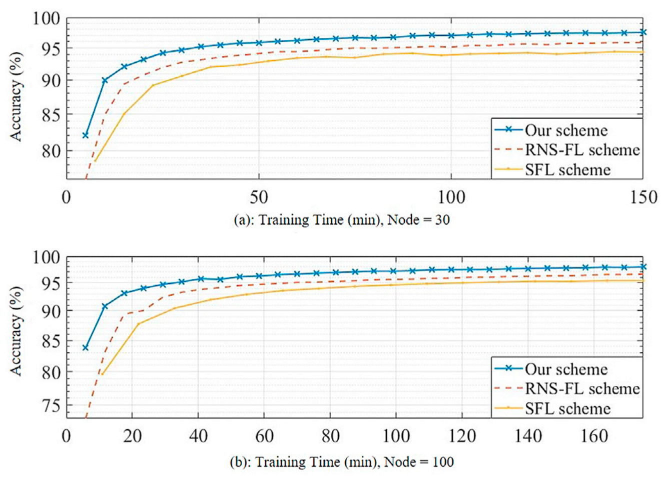


Fig. 3. Accuracy comparison on MNIST 10-digital dataset.

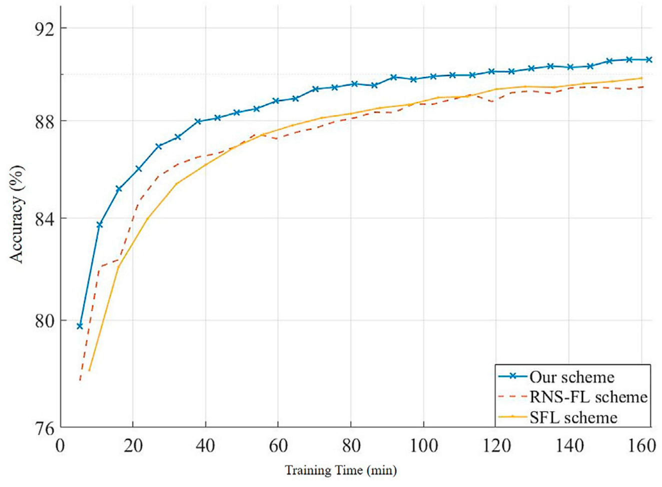


Fig. 4. Accuracy on MNIST fashion (Nodes = 50).

Table 2
List of parameters.

Parameter	Value
Node Set I	$N = 100$
Local dataset	$D_i \in [200, 1000]$
Local Iteration Steps	$n_i \in \{6, 8, 10, 12, 14, 16, 18\}$
Size of Weight Vector	7.5 MB
Communication Protocol	IEEE802.11 Family
Machine Learning Model	A standard CNN (5 × 5)
Dataset	Three different datasets
Training Dataset Size	300,000 examples
Testing Dataset Size	50,000 examples
Minibatch Size	50 samples, 100 samples

Table 3
Time of arrival at the desired accuracy and final accuracy.

	Nodes = 10 (M10)	Nodes = 30 (M10)	Nodes = 100 (M10)	Nodes = 50 (MF)
Our scheme (@ Accuracy 95%)/min	42	30	31	90
RNS-FL scheme (@ Accuracy 95%)/min	118	71	66	–
SFL scheme (@ Accuracy 95%)/min	–	–	100	157.1
Our scheme (FA)/%	**96.89**	**97.59**	**97.96**	**90.72**
RNS-FL scheme (FA)/%	95.74	96.15	96.60	89.63
SFL scheme (FA)/%	94.11	94.65	95.409	90.02

FA = Final Accuracy, @ Accuracy x % = To reach Accuracy of x %, MNIST 10-digital (M10), MNIST Fashion (MF).

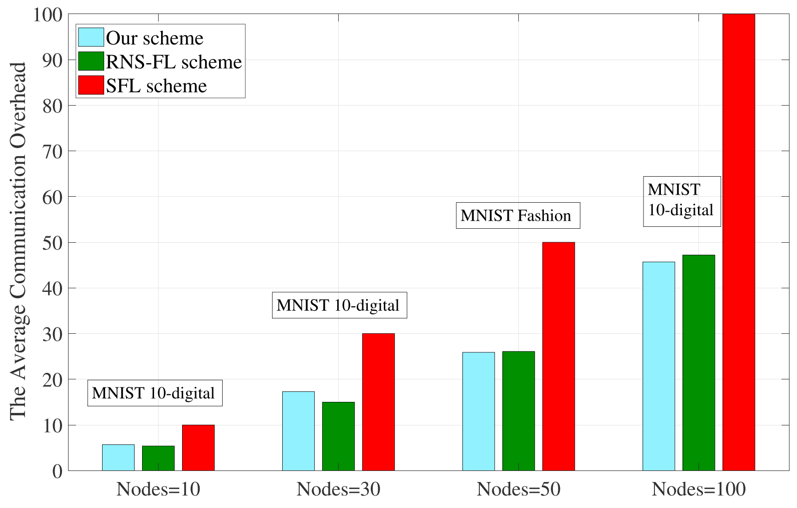


Fig. 5. The average communication overhead with the number of different nodes.

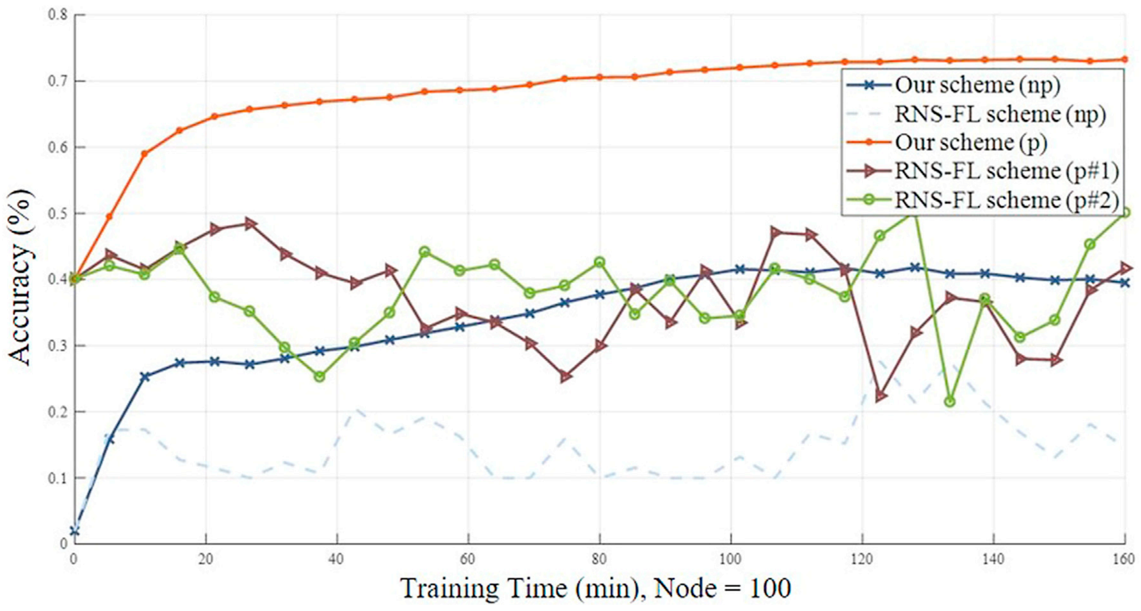


Fig. 6. Accuracy on EMNIST with non-i.i.d distribution (number of nodes is 100).

A standard Convolutional Neural Network (CNN) model is deployed among all the nodes and aggregators [24]. It is composed of two 5 × 5 convolutional layers with 32 channels for the first layer and 64 channels for the second layer. Each convolutional layer is followed by a 2 × 2 maximum pooling layer, a fully connected layer with 512 units and *ReLu* activation, and a final softmax output layer (1,663,370 parameters). We summarize the key parameters for the experiment in Table 2.

The following two state-of-the-art schemes are considered as baseline schemes for performance comparison: (i) Random Nodes Selection Federated Learning (RNS-FL), as the original federated learning protocols in Ref. [24], in which the aggregator randomly selects nodes to participate in the global aggregation. (ii) Synchronous Federated Learning (SFL), as a synchronous distributed gradient descent based scheme, in which the aggregator collects the weight parameters from all nodes at each round [27].

The ultimate goal of federated learning is to classify data samples. Performance metrics to evaluate the proposed scheme include the overall accuracy, the time of arrival at the desired accuracy, and the average communication overhead. Specifically, accuracy is defined as the ratio of the number of correct predictions over the total number of predictions. The time of arrival at the desired accuracy is the time taken for reaching a given accuracy. The average communication overhead is defined as the average number of messages to transmit the weight vector in the global aggregation process.

4.2. *Evaluation results*

We evaluate the accuracy of the proposed scheme and the RNS-FL and SFL schemes with different numbers of IoT nodes on the MNIST 10-digital dataset, and show the results in Fig. 3. Specifically, Fig. 3 (a) illustrates the experimental results of 30 nodes from the MNIST 10-digit dataset. From this figure, we can observe that in a certain training time, our proposed scheme achieves higher accuracy than the other two baseline comparison schemes. The reason for the improvement is that during the iterative process, the computing contribution is considered so that the proposed asynchronous scheme can schedule the aggregation

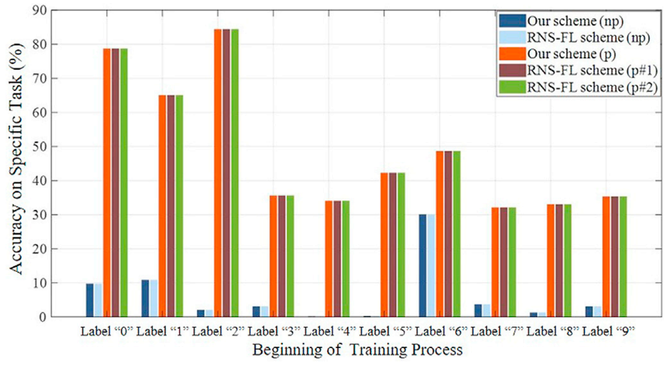


Fig. 7. Accuracy on each task (initial state).

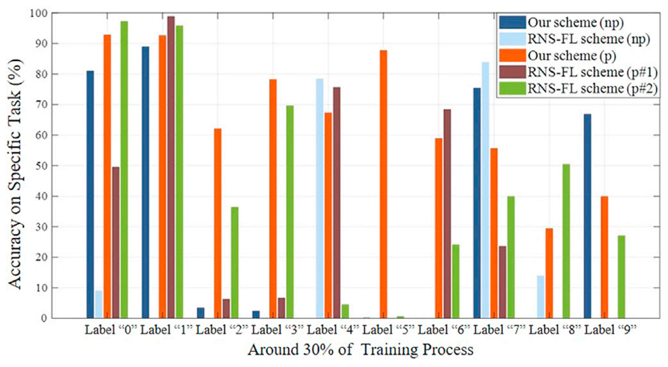


Fig. 9. Accuracy on each task (around 30% of training process).

more efficiently than synchronous schemes. In addition, the experiments that are conducted on the entire node set of 100 nodes demonstrate that our proposed asynchronous scheme can still work efficiently with a large number of nodes. As seen in Fig. 3 (b), our proposed asynchronous scheme achieves the highest accuracy among all schemes with the same training time. It also shows a comparable convergence rate with synchronous schemes. Moreover, we conduct the experiments on the MNIST Fashion dataset. Similar to the results shown in Fig. 3, the experimental results in Fig. 4 further demonstrate that our proposed scheme achieves the best performance.

In Table 3, we show the performance comparisons for the time of arrival at the desired accuracy (@ Accuracy x %) and the Final Accuracy (FA) with the different schemes. As seen in Table 3, FA increases with the increasing number of nodes, and our proposed asynchronous scheme reaches the desired accuracy with a higher efficiency (e.g., less training time). This is because the proposed node selection scheme can opportunistically select the IoT nodes with sufficient computing ability in each round. Moreover, the synchronous federated learning scheme in Fig. 5 shows that it has to afford a relatively high communication overhead as it impractically waits for all node updates in each iteration. It is inefficient in the case in which one node may shift to offline due to reasons (battery issues, transmission failures, etc.). Finally, by combining the overall accuracy in Figs. 3 and 4, along with the average communication overhead in Fig. 5, we confirm that the proposed asynchronous scheme has the ability to balance the heterogeneous computing resources and the limited network bandwidth, leading to better performance for the federated learning process.

When it comes to the non-i.i.d case, we customize the dataset for the experiment. We use the raw data samples from EMNIST where it contains 240, 000 training samples and 40, 000 testing samples. We would sort the data samples by their label information, divide it into 480 shards with a size of 500 data samples. All these sub-datasets are allocated to the local nodes and each local node only occupy 1 or 2 labels. In other words, the dataset used in this case is a highly skewed distribution, in which each node will have samples of two different classes at most. We use EMNIST 10-digital dataset that is a representative dataset in the machine learning field as an example to implement our idea. We plan to explore specific IoT datasets to validate our scheme in IoT applications in the future. To further validate our scheme on the non-i.i.d data distribution, we consider different performance metrics from the previous i.i.d. case, including the overall accuracy and specific task (Label) accuracy. Moreover, we sample the accuracy of each learning class to record accuracy variations over the training process.

When it comes to the non-i.i.d scenario, we first test our scheme and the RNS-FL scheme without pre-training, which are our scheme (np) and RNS-FL scheme (np), as shown in Fig. 6. Nonetheless, as shown in the figure, our scheme (np) and RNS-FL scheme (np) achieve poor performances, where the overall accuracy of the two non-pre-training schemes did not reach 50%, and the RNS-FL scheme (np) keeps divergence in the training process. The main reason behind this is that the highly skewed data distribution significantly generates gradient divergences against a non-pre-training model.

To alleviate these issues, we take the pre-training model into the non-i.i.d. scenario. The pre-training model is added to the aggregator (server). We use MNIST-10 dataset to construct 5000 data samples with i.i.d. data distribution. Then, we conduct a one-round (one epoch) warm-up training. After adding the pre-training process, we show the overall accuracy of our scheme and RNS-FL scheme in Fig. 6, including our scheme (p), RNS-FL scheme (p#1) and RNS-FL scheme (p#2). Note that RNS-FL scheme (p#1) and RNS-FL scheme (p#2) have different randomness seeds. As seen in the figure, our proposed asynchronous scheme with pre-training achieves the highest accuracy among all schemes with the same training time. It also shows a better convergence rate with other baseline schemes.

In addition, we sample the accuracy on specific tasks (class) over the training process. In Fig. 7, we show the initial accuracy of each task where the pre-training model achieves around 40% on average. Further, from Figs. 8 – 13, we can observe that the performance of the RNS-FL

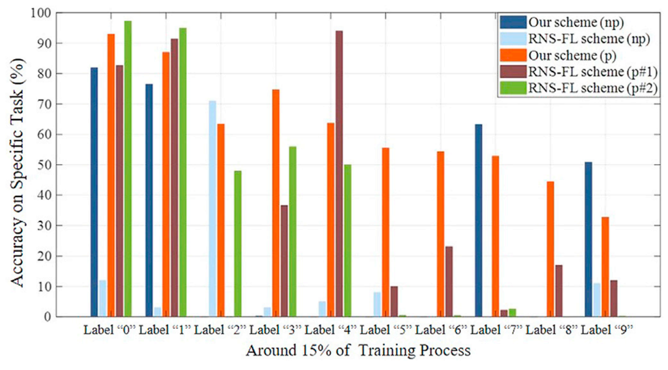


Fig. 8. Accuracy on each task (around 15% of training process).

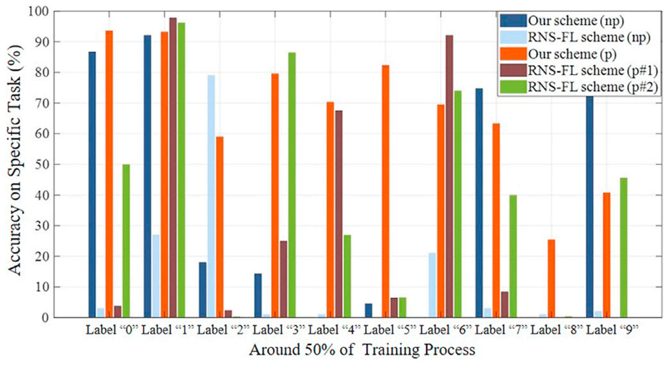


Fig. 10. Accuracy on each task (around 50% of training process).

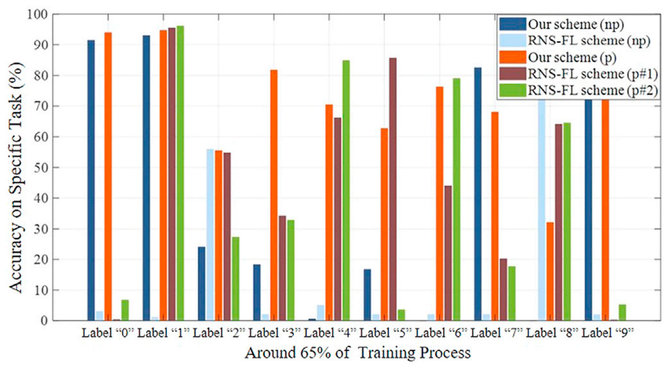


Fig. 11. Accuracy on each task (around 65% of training process).

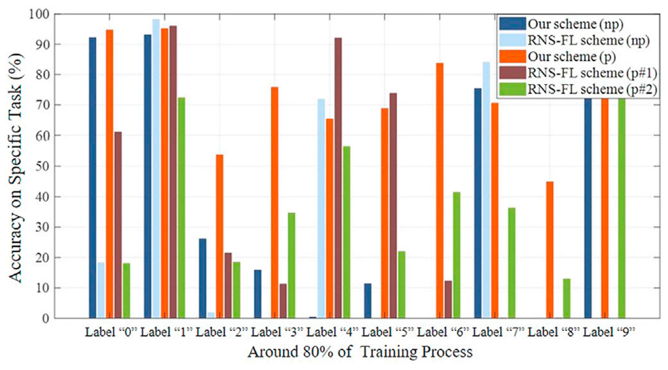


Fig. 12. Accuracy on each task (around 80% of training process).

scheme cannot keep stable where the accuracy of some specific tasks degrades severely in certain rounds. The RNS-FL scheme may ignore the computing resource and the unique data resource on local devices, resulting in poor performance of specific tasks.

Finally, we summarize meaningful numerical experimental results of the non-i.i.d. distribution into Table 4, which include Final Accuracy of the global learning model (FA), convergence, Average Selected Device (ASD), and @ Accuracy x % task of the final learning model (a task of reaching a given accuracy x %). Thus, we consider the experimental results of the non-i.i.d. case as strong evidence for the robustness of our scheme.

5. Further discussion

The experimental results in Section 4.2 validate the efficacy of our proposed asynchronous federated learning scheme. In the following, we discuss several open problems as future research directions.

5.1. IoT systems

To further extend and understand the scalability and resilience of our scheme, and in addition to further experiments with additional machine learning models and datasets, different applications in IoT systems should be considered. More factors can and should be taken into consideration, such as the mobility and battery capacity of devices [39–41]. For example, IoT nodes such as smartphones and smart vehicles are highly mobile, leading to dynamic network topology [40]. During the course of movement, the wireless connection is highly unstable and can be lost (e.g., a vehicle is out of the coverage of roadside units). Thus, the appropriate design and extension of our scheme must consider service migration potential buffers to hold weight vectors. In this way, the nodes that lose the connection with the edge server will not cause the loss of weight vectors. Furthermore, we plan to do further investigation on distributed machine learning of different IoT systems, such as smart grid, smart transportation, smart cities, and smart manufacturing, among others.

5.2. Machine learning

There are two main mechanisms of distributed machine learning: the data distribution mechanism and the model distributed mechanism. While existing works have focused primarily on the data distributed mechanism, how to independently distribute some machine learning models to a group of nodes without affecting the effectiveness and performance of the combined/recombined model is still a problem to be

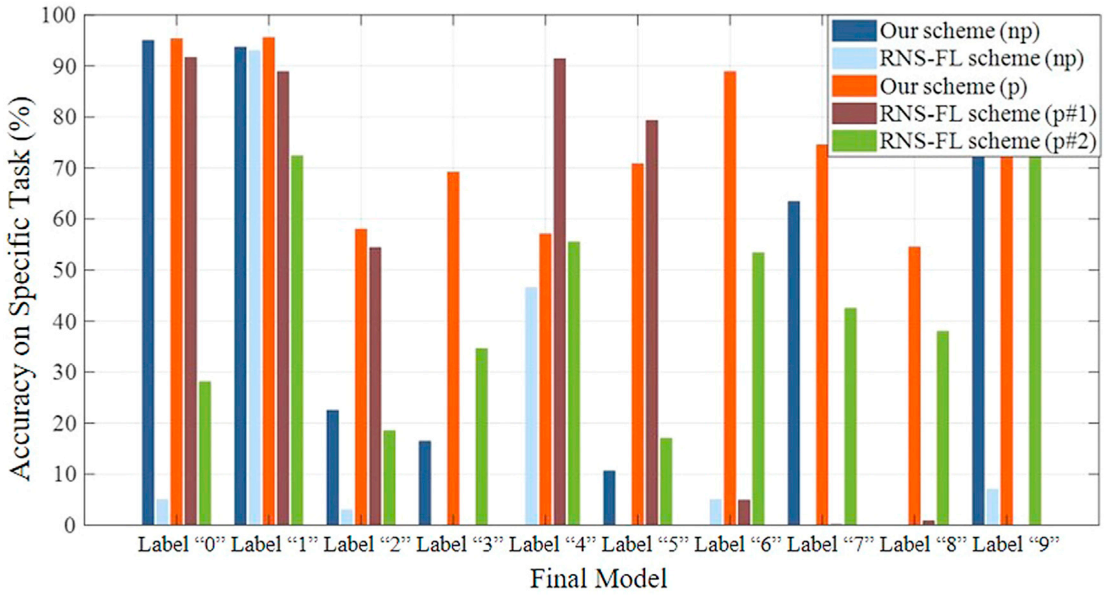


Fig. 13. Accuracy on each task (final model).

Table 4
Numerical results over Non-i.i.d. Data distribution.

	FA/%	ASD	@ Accuracy 70 Classes %	Convergence
Our scheme (p)	**72.25**	**25.4**	**"0", "1", "5", "6", "7", "9"**	✓
RNS-FL scheme (p#1)	46.35	27.2	"0", "1", "4", "5"	×
RNS-FL scheme (p#2)	50.16	29.1	"1", "9"	×
Our scheme (np)	39.92	25.4	"0", "1", "9"	✓
RNS-FL scheme (np)	17.74	24.1	"1"	×

solved [42].

We notice that online machine learning is a new research direction in distributed machine learning, in which the training data becomes available in sequential order. Similar to the basic federated learning framework, a reasonable assumption is that it is infeasible to globally access the entire training dataset, and requiring networks to connect some training datasets and distribution computing capability. In this manner, we shall design an integrated system in which online learning supports a group of nodes, while each node does not have enough computing capacity. Another issue is the delayed gradients problem [43], which means that once a node updates its local weight vector to the aggregator, the aggregator may have aggregated parameter updates from other nodes. In connection with this problem, we shall consider leveraging the Taylor expansion of the gradient function and the efficient approximation for computing the gradients of the loss function for delay gradient compensation in the proposed federated learning scheme.

5.3. *Security and privacy*

Federated learning is a distributed machine learning framework, not only allowing nodes to train the model by its own data, but also considering privacy by default. Nonetheless, such a distributed learning framework raises new potential security issues [12,44–46]. Compared to compromising a cloud server directly, an external adversary could more easily compromise and control a small number of IoT devices in the network. For example, the targeted poisoning attack could be launched by the adversary to intentionally mislead the global training model to misclassify targeted samples [36].

More precisely, the adversary compromises only one distributed node or multiple nodes to inject poisoning data samples. The trained global model converges to a good point which achieves a high and acceptable accuracy over the testing dataset, but a targeted backdoor has been successfully injected into that model, bypassing any existing detection mechanisms [45]. We thus consider developing methods to detect or mitigate the external adversary by using some suitable distance metrics to verify the received updates.

Furthermore, in a federated learning system, the lack of trustworthy evaluation could lead to significant security implications (e.g., the reduction of learning performance, unexpected backdoors and malicious updates). To this end, we shall systematically investigate and design methods that enable the trusted evaluation of all components in the federated learning system as well as the transmitted and received information.

By doing this, the central server (i.e., learning aggregator) can have an overall trust score of individual learning nodes and received information. For example, trusted evaluation could take both short-term and long-term behaviors of individual local learning nodes into account (e.g., the score in the current federated learning task and the accumulated reputation in previous learning tasks, among others) [47,48]. Such information can greatly assist the server in prioritizing the received information from different learning nodes and consequently make the overall learning results more robust.

6. Final remarks

In this paper, we have addressed the challenges of developing the asynchronous federated learning process in distributed edge-powered IoT systems under dynamic communication networks. We have proposed an asynchronous federated learning scheme, which can work effectively with heterogeneous IoT devices. To adaptively determine the participation of nodes in each round of global aggregation, we have proposed a heuristic greedy node selection algorithm to obtain an approximate solution with low computational complexity. By conducting a series of quantitative experiments on large datasets, we have validated the efficacy of the proposed scheme and further confirmed that our proposed scheme outperforms the two representative state-of-the-art schemes, considering both i.i.d and non-i.i.d data distributions. Finally, we have discussed some remaining issues as future research directions.

References

[1] William Grant Hatcher, Wei Yu, A survey of deep learning: platforms, applications and emerging research trends, IEEE Access 6 (2018) 24411–24432.
[2] M. Mohammadi, et al., Deep learning for IoT big data and streaming analytics: a survey, IEEE Communications Surveys Tutorials 20 (2018) 2923–2960, 4.
[3] H. Xu, et al., A survey on industrial internet of things: a cyber-physical systems perspective, IEEE Access 6 (2018) 78238–78259, https://doi.org/10.1109/ACCESS.2018.2884906. ISSN: 2169-3536.
[4] H. Wang, M. Daneshmand, H. Fang, Artificial intelligence (AI) driven wireless body area networks: challenges and directions", in: 2019 IEEE International Conference on Industrial Internet (ICII), 2019, pp. 428–429.
[5] J. Lin, et al., A survey on internet of things: architecture, enabling technologies, security and privacy, and applications, in: IEEE Internet of Things Journal, vol. 4, 2017, pp. 1125–1142, 5.
[6] J. Granjal, E. Monteiro, J. S Silva, Security for the internet of things: a survey of existing protocols and open research issues, IEEE Communications Surveys Tutorials 17 (3) (2015) 1294–1312.
[7] Q. Yang, et al., On false data-injection attacks against power system state estimation: modeling and countermeasures, in: IEEE Transactions on Parallel and Distributed Systems, vol. 25, 2014, pp. 717–729, 3.
[8] J. Lin, et al., Data integrity attacks against dynamic route guidance in transportation-based cyber-physical systems: modeling, analysis, and defense, in: IEEE Transactions on Vehicular Technology, vol. 67, 2018, pp. 8738–8753, 9.
[9] X. Yang, et al., Survey on improving data utility in Di_erentially private sequential data publishing, in: IEEE Transactions on Big Data, 2017, 1–1.
[10] Nicolas Papernot, et al., Security and privacy in machine learning", in: 2018 IEEE European Symposium on Security and Privacy (EuroS&P), IEEE., 2018, pp. 399–414.
[11] D. Wu, et al., A feature-based learning system for internet of things applications, in: IEEE Internet of Things Journal, vol. 6, 2019, pp. 1928–1937, 2.
[12] F. Liang, et al., Machine learning for security and the internet of things: the good, the bad, and the ugly, IEEE Access 7 (2019) 158126–158147.
[13] John C. Duchi, Michael I. Jordan, Martin J. Wainwright, Privacy aware learning, J. ACM 61 (6) (2014) 38.
[14] H Brendan McMahan, et al., Learning differentially private recurrent language models, arXiv preprint arXiv (2017), 1710.06963.
[15] W. Shi, et al., Edge computing: vision and challenges, in: IEEE Internet of Things Journal, vol. 3, 2016, pp. 637–646, 5.
[16] W. Yu, et al., A survey on the edge computing for the internet of things, IEEE Access 6 (2018) 6900–6919, https://doi.org/10.1109/ACCESS.2017.2778504. ISSN: 2169-3536.
[17] Hanpeng Hu, Dan Wang, Chuan Wu, Distributed machine learning through heterogeneous edge systems, AAAI, 2020, pp. 7179–7186.
[18] Junxian Huang, et al., An in-depth study of LTE: effect of network protocol and application behavior on performance, in: ACM SIGCOMM Computer Communication Review, vol. 43, ACM, 2013, pp. 363–374, 4.
[19] Shiqiang Wang, et al., Mobility-induced service migration in mobile micro-clouds, in: MILCOM '14 Proceedings of the 2014 IEEE Military Communications Conference, 2014, pp. 835–840.
[20] Yutao Huang, et al., Task scheduling with optimized transmission time in collaborative cloud-edge learning, in: 2018 27th International Conference on Computer Communication and Networks, ICCCN, 2018, pp. 1–9.
[21] Lin Wang, et al., Online resource allocation for arbitrary user mobility in distributed edge clouds, in: 2017 IEEE 37th International Conference on Distributed Computing Systems, ICDCS, 2017.
[22] F. Liang, et al., Toward edge-based deep learning in industrial internet of things, in: IEEE Internet of Things Journal, vol. 7, 2020, pp. 4329–4341, 5.

[23] Umair Mohammad, Sameh Sorour, Adaptive task allocation for mobile edge learning, in: IEEE Wireless Communications and Networking Conference (WCNC'19), 2019.

[24] Brendan McMahan, et al., Communication-E_cient learning of deep networks from decentralized data, in: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, vol. 54, Proceedings of Machine Learning Research, 2017, pp. 1273–1282.

[25] Li Tian, et al., Federated learning: challenges, methods, and future directions, 2019 arXiv preprint arXiv:1908.07873.

[26] Jianmin Chen, et al., Revisiting distributed synchronous SGD, in: International Conference on Learning Representations Workshop Track, 2016.

[27] Shiqiang Wang, et al., When edge meets learning: adaptive control for resource-constrained distributed machine learning, in: IEEE Conference on Computer Communications (INFOCOM'18), 2018, pp. 63–71.

[28] Yue Zhao, et al., Federated Learning with Non-IID Data, 2018 arXiv preprint arXiv: 1806.00582.

[29] Nour Moustafa, et al., Outlier Dirichlet mixture mechanism: adversarial statistical learning for anomaly detection in the fog, in: IEEE Transactions on Information Forensics and Security, vol. 14, 2019, pp. 1975–1987, 8.

[30] H. Xu, et al., Reinforcement learning-based control and networking Co-design for industrial internet of things, IEEE J. Sel. Area. Commun. 38 (2020) 885–898, 5.

[31] Mohan Li, et al., Deep reinforcement learning for partially observable data poisoning attack in crowdsensing systems, in: IEEE Internet of Things Journal, 2019.

[32] Pavel Mach and Zdenek Becvar. "Mobile edge computing: a survey on architecture and computation offloading". In: IEEE Communications Surveys & Tutorials 19.3 % 28%29, pp. 1628–1656.

[33] Xiangru Lian, et al., Asynchronous decentralized parallel stochastic gradient descent, in: Proceedings of the 35th International Conference on Machine Learning, vol. 80, 2018. Stockholm Sweden.

[34] Takayuki Nishio, Ryo Yonetani, Client Selection for Federated Learning with Heterogeneous Resources in Mobile Edge, 2018 arXiv preprint arXiv:1804.08333.

[35] Steven Skiena, Who is interested in algorithms and why?: lessons from the stony brook algorithms repository, ACM SIGACT News 30 (3) (1999) 65–74.

[36] Zheyi Chen, et al., Zero knowledge clustering based adversarial mitigation in heterogeneous federated learning, in: IEEE Transactions on Network Science and Engineering, 2020.

[37] M.P.R.S. Kiran, Pachamuthu Rajalakshmi, Saturated throughput analysis of IEEE 802.11 ad EDCA for high data rate 5G-IoT applications, IEEE Trans. Veh. Technol. 68 (5) (2019) 4774–4785.

[38] M Zulfiker Ali, Jelena Mišić, Vojislav B. Mišić, Performance evaluation of heterogeneous IoT nodes with differentiated QoS in IEEE 802.11 ah raw mechanism, IEEE Trans. Veh. Technol. 68 (4) (2019) 3905–3918.

[39] Hamed Shah-Mansouri, WS Wong Vincent, Hierarchical fog-cloud computing for IoT systems: a computation o_oading game, in: IEEE Internet of Things Journal, vol. 5, 2018, pp. 3246–3257, 4.

[40] Hua Kun, et al., A game theory based approach for power efficient vehicular ad hoc networks, in: Wireless Communications and Mobile Computing 2017, 2017.

[41] Lewis Tseng, et al., Blockchain for managing heterogeneous internet of things: a perspective architecture, IEEE Network 34 (1) (2020) 16–23.

[42] Joost Verbraeken, et al., A survey on distributed machine learning, in: ACM Computing Surveys (CSUR), vol. 53, 2020, pp. 1–33, 2.

[43] Shuxin Zheng, et al., Asynchronous stochastic gradient descent with delay compensation, in: Proceedings of the 34th International Conference on Machine Learning-Volume 70, JMLR. org., 2017, pp. 4120–4129.

[44] Xinyun Chen, et al., Targeted backdoor attacks on deep learning systems using data poisoning, 2017 arXiv preprint arXiv:1712.05526.

[45] Arjun Nitin Bhagoji, et al., Analyzing federated learning through an adversarial lens, 2018 arXiv preprint arXiv:1811.12470.

[46] Wenqi Wei, et al., Adversarial examples in deep learning: characterization and divergence, in: CoRR abs/1807.00051, 2018 arXiv: 1807 . 00051. URL, http://arxiv .org/abs/1807.00051.

[47] David Danks, The value of trustworthy AI, in: Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society, 2019, pp. 521–522.

[48] Gilad Baruch, Baruch Moran, Yoav Goldberg, A little is enough: circumventing defenses for distributed learning, in: Advances in Neural Information Processing Systems, 2019, pp. 8632–8642.