Resource Allocation for Latency-aware Federated Learning in Industrial Internet-of-Things

Weifeng Gao, *Student Member, IEEE,* Zhiwei Zhao*, *Member, IEEE,* Geyong Min, *Member, IEEE,* Qiang Ni, *Senior Member, IEEE,* Yuhong Jiang

***Abstract*—Federated Learning (FL) has been employed for tremendous privacy-sensitive applications, where distributed devices collaboratively train a global model. In Industrial Internet-of-Things (IIoT) systems, training latency is the key performance metric as the automated manufacture usually requires timely processing. The existing works increase the number of effective devices to accelerate the training. However, devices in IIoT systems are usually densely deployed, increasing the number of clients can potentially cause serious interference and prolonged training latency. In this paper, we propose *RaFed*, a resource allocation scheme for FL. We formulate the problem of reducing training latency as an optimization problem, which is proved to be NP-hard. We propose a heuristic to select appropriate devices to achieve a good trade-off between the interference and convergence time. We conduct experiments using an RGB-D dataset in an IIoT system. The results show that compared to the state-of-the-art works, RaFed significantly reduces the latency by 29.9%.**

***Index Terms*—Federated learning, Industrial Internet of Things, Wireless communication, Resource allocation**

I. Introduction

Industrial Internet-of-Things (IIoT) has been considered as one of the key technologies for the implementation of Industry 4.0 [1]. In IIoT systems, each front-end device continuously generates a large amount of data, which usually requires to be processed with machine learning approaches. For example, in the Automatic Sorting System, the images of the industrial products are captured and stored by multiple cameras from different shooting angles. The images will further be collected to a centralized server for processing and analysis with machine learning methods. The results are then used for automatic decisions along the production assembly lines.

Since the IIoT data from all the front-end devices are collected to the central server, the privacy problem appears as an important concern in the traditional learning methods, because the data from some confidential products/objects can involve sensitive information [2]. For example, multiple factories may rely on the image identification services provided by the same third party [3], [4], but they may do not want to share their data with the third party to train the centralized machine learning model. Therefore, Federated Learning (FL) is promising for privacy-preserving data processing in IIoT systems [5], [6], where front-end devices are allowed to preserve training data locally and collaboratively train a global learning model.

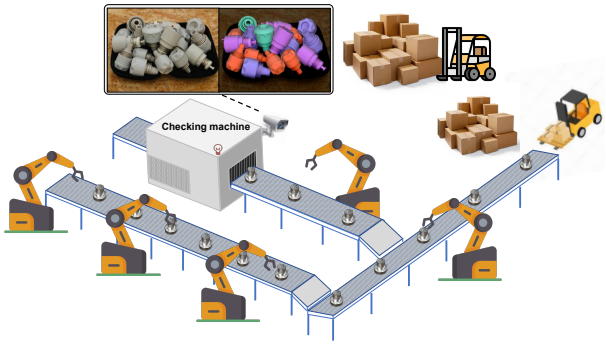

Fig. 1. Automatic Sorting System.

However, it is non-trivial to employ FL in IIoT systems, because IIoT systems rely on the *timely* completion of each processing procedure in manufacturing [7], adding a stringent requirement on the training latency of FL. Considering an Automatic Sorting System as an example, as shown in Figure 1, where the industrial products first pass through the checking machines on a conveyor. The machines are equipped with cameras to collect product images and identify the product types with certain machine learning models [8], [9]. The identified products can then be correctly picked for further assemblage and transportation. Apparently, the products identification needs to be finished in time before the products goes into the picking procedure. Otherwise, the whole manufacturing pipeline will be interrupted or even broken.

There are some existing works on reducing the training latency of FL in general mobile networks [7], [10]. The main idea of these works is to use the improved number of FL clients (active devices) to reduce the convergence time. For example, the state-of-the-art work [10] tries to increase the number of powerful FL clients with more local data, so that the global model can converge with fewer iterations and thus the training latency is reduced. Considering the datarate of cellular networks, most of these works assume the updates can be transmitted with very short time-on-air and there are

Weifeng Gao, Zhiwei Zhao and Yuhong Jiang are with the College of Computer Science and Engineering, University of Electronic Science and Technology of China, China. E-mail: {weifeng, yuhong}@mobinets.org, zzw@uestc.edu.cn,

Geyong Min is with the Department of Computer Science, University of Exeter, U.K. E-mail: g.min@exeter.ac.uk,

Qiang Ni is with the School of Computing and Communications, Lancaster University, Lancaster LA1 4WA, U.K. E-mail: q.ni@lancaster.ac.uk

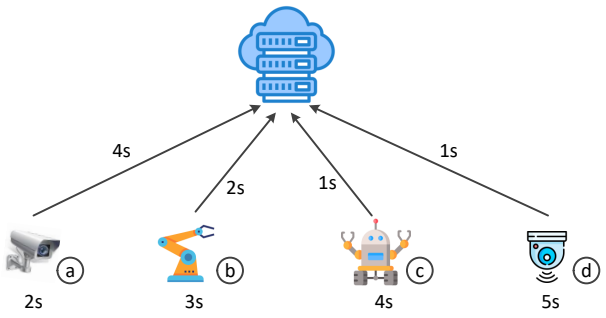

Fig. 2. Motivative example of FL.

few inter-client collisions.

However, as the IIoT devices are usually resource-constrained and densely deployed (e.g., devices that are deployed within the area of a workshop), the relatively long time-on-air of the IIoT transmissions can easily cause collisions. As a result, if we follow the existing works and increase the number of FL clients, the collision probability can be drastically increased. The overall training latency will be further increased as the server has to wait for updates from all FL clients in each iteration.

To address the above challenge for FL in IIoT systems, in this paper, we propose *RaFed*, a resource allocation scheme for FL in industrial IoT, to reduce the training latency of FL. We first analyze the latency of different distributed devices uploading their model updates to the server, where both the computation latency and the wireless transmission latency is considered. RaFed jointly considers the selection of active devices and the allocation of wireless resources such as the channels, to minimize the training latency of the global model, which is formulated as an optimization problem, where the optimization goal is to minimize the longest device latency. The problem is proved to be NP-hard, and a heuristic algorithm is proposed to gradually reduce the training latency. We evaluated the proposed method in comparison with the state-of-the-art works using a public IIoT dataset, T-LESS, which is the data for identifying the industry-relevant objects in IIoT systems like Automatic Sorting Systems.

In summary, this paper makes the following contributions.

- We formulate the problem of reducing training latency of FL for IIoT systems by allocating the active devices and wireless resources, which is proved to be NP-hard.
- We propose a heuristic algorithm to iteratively improve the training latency until the latency reduction is acceptable.
- We conduct experiments using an RGB-D dataset of texture-less objects in an IIoT automatic sorting system. The results show that compared to the state-of-the-art works, RaFed significantly reduces the FL latency by 29.9%.

II. Motivation

In this section, we use two illustrative examples to discuss the bottleneck of the FL training latency, as well as the impact of resource allocation on this latency.

TABLE I
Training latency of different allocation

Device ID	Random selection (s)	Shortest local latency (s)	Least iteration (s)	Shortest training latecny (s)
a	7		12	
b	5.5		8	6.33
c		5	6.5	5.67
d			7.5	6.67
max{a,b,c,d}	7	5	12	6.67
Training latency (s)	56	50	48	40

Figure 2 illustrates the basic architecture of FL, with four distributed devices running industrial IoT systems and a centralized server that is responsible for training a global model. Basically, these four devices iteratively receive the global model from the server. After training locally, they update their local models and upload these updates to the server. In this paper, we mainly focus on the latency through the training process.

In Figure 2, the numbers under the four devices denote their computation latency under different processing capability of their hardware, and the numbers beside the links between the devices and the server are the wireless communication latency under different signal fading conditions. For wireless communication latency, we consider the impact of multiple devices interfering with each other. By selecting more devices to upload their model updates, the transmissions will be more likely to fail, so we prolong the communication latency with different packet reception ratio. Specifically, with one to four selected devices, the packet reception ratio is set to 100%, 80%, 60% and 40%, respectively. Besides, the training requires multiple iterations, and the number of iterations is related to the number of selected devices (i.e., selecting more devices leads to faster convergence thus fewer iterations). For simplicity, we follow the assumption in [11], where increasing the number of selected devices can linearly speed up the convergence. Specifically, with one to four devices selected, the number of iterations is set to 10, 8, 6, and 4, respectively.

Training latency of different allocation. We first consider the random device selection, where devices a and b are selected. Their training latency is calculated by the computation latency plus the transmission latency, i.e., $t_a = 2 + \frac{4}{0.8} = 7s$, and $t_b = 3 + \frac{2}{0.8} = 5.5s$. Since the training latency is limited by the device with longer latency, this random selection scheme leads to $7 \times 8 = 56s$ training latency in total.

To explore different selection schemes which reduce the wireless interference and the transmission latency, we consider the scenario that only one device with the least local latency, i.e., device c. The training latency will be $5 \times 10 = 50s$, which is shorter than the above random selection scheme. On the other hand, if we consider the allocation which uses the least number of iterations, all four devices are selected to compute and upload their local model updates to the server. The latency of them can also be calculated as the method above and the

results are: $t_a = 2 + \frac{4}{0.4} = 12s$, $t_b = 3 + \frac{2}{0.4} = 8s$, $t_c = 4 + \frac{1}{0.4} = 6.5s$, and $t_d = 5 + \frac{1}{0.4} = 7.5s$, respectively. As a result, the training latency is 48s.

Now let us exploit a better selection scheme that can further reduce the training latency. If we select the devices b, c and d, with the packet reception ratio 60% and 6 iterations, we can calculate their corresponding training latency as: $t_b = 3 + \frac{2}{0.6} = 6.33s$, $t_c = 4 + \frac{1}{0.6} = 5.67s$, and $t_d = 5 + \frac{1}{0.6} = 6.67s$, respectively. Therefore, the total training latency will be $6.67 \times 6 = 40s$. These examples indicate that the lossy nature of wireless links for industrial IoT systems makes the selection of active devices have a great impact on the training latency of FL. Furthermore, if we consider the resource allocation of the wireless channels, i.e., the three devices in the last example can use different channels so that their transmissions do not interfere with each other, the communication latency can be further reduced. Specifically, the packet reception ratio of them will be 100%, and their training latency becomes: $t_b = 5s, t_c = 5s$ and $t_d = 6s$. The total training time will be 36 seconds.

The results of the above examples are concluded in Table I. It can be inferred from these examples that both the active devices and the wireless resources should be carefully allocated such that the wireless environment can contribute to the reduction of total training latency of FL. We analyze the impact that the device selection and wireless resource allocation have on the training latency and develop a mathematical model to represent this impact. To solve the optimization of minimizing the training latency, a heuristic algorithm is proposed to get the resource allocation.

III. Design of RaFed

In this section, we present the design of RaFed in detail. The objective of RaFed is to optimize the latency for the server to collect all the local model updates from the industrial front-end devices. We first analyze the uploading latency and the impact of the resources such as wireless channels on this latency for an front-end device, which consists of the computation latency of local training and the communication latency with a given wireless channel condition. With the uploading latency, we formulate the problem of reducing the collection latency on the server as an optimization problem. With the NP-hardness of the optimization problem, a heuristic algorithm is proposed. The notations used in this paper are summarized in Table II.

A. System model

We consider an industrial wireless network for industrial 4.0 applications, basically consisting of one server and a set of distributed devices $\mathcal{N}$. Each device i owns its local data set, which is generated via the actions of the device such as monitoring, and its size is denoted by D_i. With these local datasets, we can implement different applications like analyzing and predicting the condition of the environment.

TABLE II
Notations used in this paper

Symbols	Notations
$\mathcal{N}$	The set of distributed devices
D_i	Size of data on device i
D_N	Total size of the data samples
l_j	Loss function of data sample j
σ	Global accuracy threshold
y_i	Cycles for executing one sample
f_i	Frequency of the computation unit
D_{up}	The size of updated local model
B	Bandwidth of the wireless transmission
p_i	The transmit power on device i
γ	Threshold of the received signal SNR
q_i	Packet reception ratio on device i
$\mathcal{S}$	The set of active devices
c_i	The channel used on device i
d_i	Communication distance of device i
α	Path loss exponent
β, θ	Data distribution characteristics parameters
n	The number of iterations for training
$\mathcal{C}$	The set of available channels
p_{min}	Minimum available transmit power
p_{max}	Maximum available transmit power

1) Model updating in Federated Learning: FL algorithms involve multiple distributed devices learning locally and communicating with a central server periodically to reach a global consensus. During the training, the distributed devices utilize the local datasets to train the model received from the server instead of uploading the privacy-sensitive data in the traditional way. In particular, the loss function of device i based on its local dataset is defined as follows:

$$F_i(\omega) := \frac{1}{D_i} \sum_{j \in D_i} l_j(\omega), \tag{1}$$

where $l_j()$ represents the loss function of the data sample j. With the local loss function, the goal of the learning model is typically to minimize the global loss function:

$$\min_{\omega} f(\omega) := \sum_{i=1}^{N} \frac{D_i}{D_N} F_i(\omega), \tag{2}$$

where D_N denotes the total number of the data samples, and can be calculated as $D_N = \sum_{i=1}^{N} D_i$.

In a representative implementation of Federated Stochastic Gradient Descent, each device i computes average gradient $g_i = \nabla F_i(\omega_t)$ on its local data, with its loss function $F_i(\omega_t)$ and current model parameters ω_t of the t^{th} iteration. And then central server aggregates gradients obtained from these N devices to update model parameters:

$$\omega_{t+1} \leftarrow \omega_t - \eta \sum_{i=1}^{N} \frac{D_i}{D} \cdot g_i, \tag{3}$$

where η is the learning rate. Federated Averaging method (*FedAvg*) proposed in [5] adds gradient descent step to each device for improving communication efficiency. Therefore, the local training process in devices i is:

$$\omega_{t+1}^{i} \leftarrow \omega_t - \eta \nabla F_i(\omega_t), \tag{4}$$

The server receives update $\omega_(t+1)^i$ from N scattered devices and averages them with weight to update global model

$$\omega_{t+1} \leftarrow \sum_{i=1}^{N} \frac{D_i}{D} \cdot \omega_{t+1}^{i}. \tag{5}$$

After this, the server starts the next iteration round and sends the result model of the previous round to the distributed devices for training new updates. The above process repeats until a global accuracy σ is achieved, i.e., $||\nabla f(\omega_{t+1})|| \leq \sigma \cdot ||\nabla f(\omega_t)||$.

By systematic description, it is clear that the last arriving device could be the paramount bottleneck of Federated Learning due to the severe situation of network resource demand in the iteration. Therefore, we then analyze the collection latency at the server, as well as the component of the latency at the distributed devices.

2) Latency of one training iteration: The latency of one training iteration mainly consists of two parts: computation latency on local devices and communication latency when uploading model updates to the server. The distributed devices are equipped with computation units with different capabilities, so the time for training locally on them is different from each other. Specifically, a device i consumes y_i cycles of the computation unit to execute a sample in its dataset, which can be obtained and known as a priori [12]. Therefore, this device i will consume $y_i \cdot D_i$ cycles of computation unit for local training in one iteration. With the frequency of computation unit f_i of the device i, we can obtain the computation latency at the device i as follows,

$$t_c^i = b_i \cdot \frac{y_i \cdot D_i}{f_i}, \tag{6}$$

where b_i is a binary indicating whether the device i is selected to upload its local model updates, i.e., $b_i = 1$ if device i upload its data to the server, and vice versa. After the computation of local model training, the distributed devices will upload the updates of their local models to the server via wireless links. Due to the lossy nature of wireless links and the possible conflicts between multiple transmissions, the uploading of the model updates may fail and causing unacceptable waiting time at the server. We rely on the reliability of a transmission in terms of packet reception ratio to consider the wireless communication latency. Specifically, with the size of uploading packets (i.e., the updates of the local model) D_{up}, the communication latency of a distributed device i wirelessly uploading packets to the server is defined as:

$$t_w^i = b_i \cdot \frac{D_{up}}{r_i \cdot q_i}, \tag{7}$$

where r_i denotes the bit rate of device i, and q_i represents the packet reception ratio of device i. The transmission bit rate is defined as follows,

$$r_i = Bln(1 + \frac{h_i \cdot p_i}{N_0}), \tag{8}$$

where B is the bandwidth, N_0 is the background noise, p_i is the transmission power at device i, and h_i is the channel gain of the device i. According to [13], h_i can be treated as random variable and constant during the learning time by adding the outage probability constraint, e.g., for Rayleigh fading channel:

$$P(\frac{h_i \cdot p_i}{N_0} < \gamma) < \zeta, \tag{9}$$

where γ is the signal-interference noise ratio (SINR) threshold of the received signals, and ζ is the bounded probability. This constraint is equivalent to $p_i \geq \frac{-\gamma N_0}{log(1-\zeta)}$.

Packet reception ratio can be affected by both the channel quality and the interference from other devices. Since the devices in IIoT systems are usually resource-limited, we consider the network model in low power wireless networks in [14], where the distributed devices experience competing transmission environment, and their signals would conflict with each other if they are using the same wireless channel. Generally, the packet reception ratio can be represented by:

$$\begin{aligned} q_i &= P\{SINR \geq \gamma\} \\ &= P\{\frac{p_i \cdot h_i \cdot a_i}{\sum\limits_{\substack{c_j=c_i,\\ j\neq i}}^{\mathcal{S}} p_j \cdot h_j \cdot a(d_j) + N_0} \geq \gamma\} \\ &= P\{h_i \geq \frac{\gamma \left[\sum\limits_{\substack{c_j=c_i,\\ j\neq i}}^{\mathcal{S}} p_j \cdot h_j \cdot a(d_j) + N_0 \right]}{p_i \cdot a_i}\} \\ &= exp(-\frac{\gamma \left[\sum\limits_{\substack{c_j=c_i,\\ j\neq i}}^{\mathcal{S}} p_j \cdot h_j \cdot a(d_j) + N_0 \right]}{p_i \cdot a_i}), \end{aligned} \tag{10}$$

where $\mathcal{S}$ is the set of selected devices with $\mathcal{S} = \{i \in \mathcal{N} | b_i = 1\}$, c_i represents the channel that device i uses for uploading, and $\sum\limits_{\substack{c_j=c_i,\\ j\neq i}}^{\mathcal{S}} p_j \cdot h_j \cdot a(d_j)$ indicates the cumulative interference on device i. a_i denotes the path loss attenuation function and can be defined as

$$a_i = \frac{v}{4\pi f_c d_i^{\alpha}}, \tag{11}$$

where d_i is the distance between the sender and receiver, v is the velocity of electromagnetic wave, f_c is the carrier frequency and α is the path loss exponent.

Basically, we consider the packet reception ratio model as in the low-power wireless networks, which is used to reflect the transmission reliability. As a result, this model can be alternated by other models for different networks such as 802.11b-based systems [15] and LTE systems [16].

B. Problem formulation

Considering different latency of the distributed devices, the objective is to minimize the training latency so that the learning time can be reduced and the FL can be efficient enough to meet the requirement of more Industrial IoT systems.

Training latency of FL. The training latency is defined by the number of iterations and the collection latency in an iteration. Specifically, the number of iterations is related to both the number of participated devices and data distribution. Since different FL applications can have various data distribution characteristics, the number of iterations required to achieve a certain accuracy [17] can be approximated as follows that can adapt to different data distributions:

$$n = \mu(\lambda + \frac{1}{|\mathcal{S}|}), \tag{12}$$

where the parameters μ and λ are determined through experiments to reflect data distribution characteristics, and we set their values according to the experiments in [17]. Since the wireless communication introduces packet error, we limit the maximum retransmission number for each device, if the transmission of a device finally fails, the global training accuracy will decrease in this iteration, and more iterations will be needed to achieve the expected global model accuracy, increasing the total training latency of FL.

According to [13], [17], since the downlink (i.e., transmissions from the server to the devices) bandwidth is much larger than that of uplinks, and basically the server power is considered to be large enough, the downlink latency of the server broadcasting the global model is considered to be negligible compared to the collection latency of uplinks and is not considered in this paper. As a result, the latency at each iteration is limited by the worst device. Therefore, the problem of reducing the training latency is formulated as:

$$\begin{aligned} & \min \max_{i \in \mathcal{N}} n \cdot (t_c^i + t_w^i), \\ s.t. \quad & \forall i \in \mathcal{N},\ c_i \in \mathcal{C} && (C_1) \\ & \forall i \in \mathcal{N},\ p_{min} \le p_i \le p_{max} && (C_2) \end{aligned} \tag{13}$$

where $\mathcal{C}$ is the set of available wireless channels, p_{min} and p_{max} denote the minimum and maximum transmission power at the industrial front-end devices. Eq. 13 indicates that the training latency of FL can be greatly impacted by the allocation of wireless resources and the active devices selection (the device selection can be considered as an allocation which we can use a vector $\boldsymbol{d} = \{b_1, b_2, ..., b_N\}$ to represent it).

In the optimization problem in Eq. 13, we have to consider both the allocation of wireless resources and the selection of the active distributed devices. With a large number of distributed devices (e.g., thousands of them) in the industrial IoT systems, as well as the multiple available wireless resource sets, the solution space can be very large. For N devices, with the number of available channels and transmission power n_c and n_p, respectively, there are totally $\sum_{k=1}^{N} \binom{N}{k} (n_c + n_p)^k$ possible allocations and it is extremely difficult to traverse all of them. The above problem of searching for the best resource allocation is a min-max optimization and is NP-hard [14], which is difficult to solve for large scale IoT networks.

To solve the above problem, RaFed performs resource allocation for FL in industrial IoT networks to reduce the learning latency and improve learning efficiency.

Algorithm 1: RaFed procedure

```
Initial() = Random(d, C, P);#Initial allocation
T_old = max(Initial());#Training latency
do
    T_0 = T_old;
    for each i ∈ N do
        for each b_i ∈ {0, 1} do
            for each (p_i, c_i) ∈ (P, C) do
                max_t = max(Alloc(p_i, c_i));
                if max_t > T_0 then
                    T_0 = max_t
                    Alloc(i) = (p_i, c_i)
while(T_old − T_0 > δ)

#Calculate maximum training latency on devices
Function max(Alloc)
  temp = 9999;
  for each i ∈ N do
      T_old = Calc(i);
      if T_old < temp then
          temp = T_old;
  Return temp;
```

C. RaFed algorithm

With the difficulty of NP-hard optimization problem, we propose a heuristic algorithm to allocate wireless resources and the active devices which provides the most reduction in collection latency.

Specifically, an initial allocation is randomly generated for the industrial networks, i.e., we randomly select the active devices and their corresponding wireless resources. Then the algorithm iteratively runs the following processing to approach to the optimal allocation. In each iteration, RaFed traverses all the devices and greedily allocate the preferred resources for each of them. For each device, the training latency of all the possible allocation is calculated on the premise of the allocation of other devices remaining constant. RaFed calculates the training latency according to the proposed model by either selecting this device, and with both of the selections, the longest training latency of those devices max_t can be selected. In this way, the wireless resources and active devices selection with the minimum max_t are determined on this device where local optimal is achieved. After one iteration with traversing all the devices, the algorithm will calculate the improvement of the training latency compared to the latency before this iteration. If the latency improvement is large enough (i.e., larger than a threshold δ), it is believed that we

can further reduce the training latency with the improvement in another iteration and next loop begins. Otherwise, the iteration will stop. δ can be set by the network operators according to the expectation of the allocation accuracy. The detailed description of the processing is shown in Algorithm 1.

IV. Evaluation

In this section, we validate the performance of RaFed with the public dataset, T-LESS [18], and compare the performance with different resource allocation schemes. T-LESS is a standard dataset employed for industry-relevant objects identification in Industrial IoT. The dataset features thirty industrial electrical products without significant texture, distinctive color or reflectance properties, and basically have similarities on their shape and size. T-LESS includes training and test images obtained with several sensors, which are captured automatically, systematically sampling the images from a view sphere. Basically, it includes 1296 training image samples per object and 504 test image samples per object, and there are 30 industry-relevant objects in total in the dataset.

A. Experimental setting

We implement RaFed in Tensorflow [19], simulating a federated network with one server and a varying number of distributed devices, and we adopt *FedAvg*, the widely-used aggregation algorithm on the server to aggregate the uploaded model updates. To simulate the wireless communication environment, we consider that the distributed devices are uniformly located in a cell, and a central server is located at the center of the cell.

Following [13], the distance between these devices and the wireless access point is uniformly distributed between 2 and 50 m, and the transmit power of devices is limited from 0.2 to 1 W. The wireless bandwidth is set to $B = 250kHz$, and the path loss exponent is $\alpha = 3.76$ [17]. The transmit power of the devices p_i is set to be integers between 0 to 20dBm, and the power spectrum density of the additive Gaussian noise is N_0 = -114 dBm/MHz. The computation capability of a distributed device y_i is uniformly distributed in 10-30 cycles/bit, and the frequency of a device f_i is uniformly distributed in 1.0-2.0 GHz. The size of data on device D_i is set as uniform distribution in 5-10 MB, and the size of uploading packets among the devices are similar and around 4.5 KB. The data distribution characteristics parameters are $\mu = 89.154$ and $\lambda = 0.00934$ [17]. We randomly split data on each local device into 80% training set, 10% testing set, and 10% validation set.

We evaluated the performance and compared RaFed with different allocation schemes:

1) $q-FFL$ [20] that tries to achieve the fairness of training accuracy among the devices in FL.
2) $FEDL$ [13] that considers the wireless communication latency to maximize the training accuracy. The above two state-of-the-art works do not involve the device selection and randomly select active devices for local model training.
3) $FedCS$ [21] that considers the impact of device selection in FL while does not consider the collisions between wireless transmissions.

For each dataset, we randomly pick up five subsets for training, testing and validating. Repeating the process, the mean and standard deviation across these five runs where applicable is reported.

B. Federated Learning Performance

Training latency. We first compared the total training latency of RaFed with the three state-of-the-art works. Figure 3 illustrates the training latency of these methods, where five experiments are repeated. In the experiments, 40 distributed devices and four wireless channels are available for selection. Experimental results in Figure 3 show that the RaFed can achieve shorter training latency than the other three methods (e.g., RaFed outperforms q-FFL, FEDL and FedCS by about 28.2%, 29.9% and 15.8%, respectively). This is because RaFed jointly considers the allocation of active devices and the wireless resources to reduce the training latency, while the benchmarks ignore the benefits of either of the allocations. We also extract the latency of wireless communication to illustrate the comparison between RaFed and native FL (i.e., Native in Figure 3). It can be seen that the wireless communication accounts for large proportion of the total training latency. It can be seen that the training latency of q-FFL and FEDL is shorter than that of FedCS, which indicates that the allocation of wireless resources can provide more performance gain than the allocation of active devices.

Convergence of FL. Different allocation methods lead to different convergence speed for training the FL model. Figure 4 illustrates the convergence of different methods as the number of iterations varies. It can be seen that the training accuracy of all the four methods first increase as the number of iterations increases, and remains unchanged since a certain number of iterations, which demonstrates that the FL converges. From Figure 4, we can also see that the increasing speed in the training accuracy is different for these different allocation methods. This is because that they choose different sets of active devices (even some of them select the same number of active devices, the selected devices will be different due to the wireless communication consideration). Since RaFed focuses on reducing the training latency, its training accuracy does not outperform than the other three allocation schemes.

C. Performance decomposition

As described in Section III, RaFed reduces the training latency of federated learning by jointly considering the allocation of wireless resources and active devices for uploading model updates. To further analyze the performance gain of RaFed, we evaluate the driving factors leading to its gain.

Impact of channel resources. Figure 5 depicts the training latency with a different number of available channels. Since FedCS assumes that the distributed devices upload their data

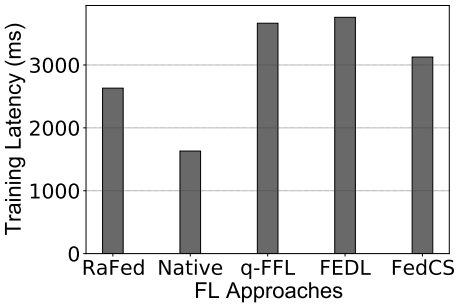

Fig. 3. Training latency of different methods.

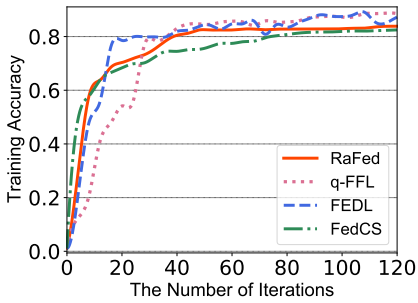

Fig. 4. Training accuracy.

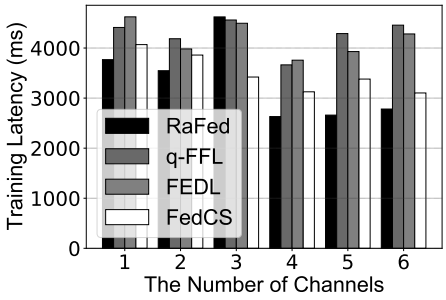

Fig. 5. Impact of available channels.

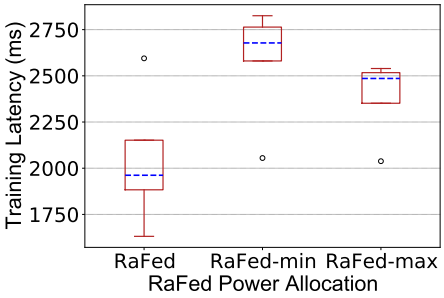

Fig. 6. Impact of transmit power allocation.

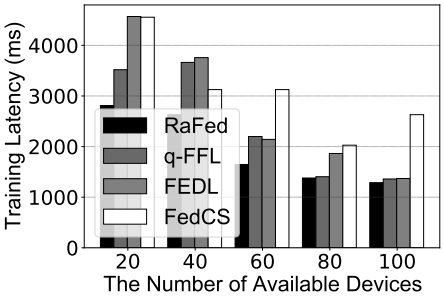

Fig. 7. Impact of device selection.

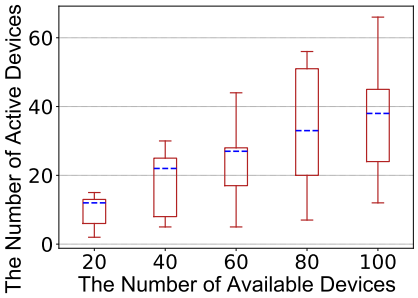

Fig. 8. The number of active devices.

one by one, which does not involve the channel allocation, we implement FedCS scheme by randomly selecting a channel in this experiment. Figure 5 illustrates that RaFed achieves shorter training latency than other methods with different number of available channels. The reason is that q-FFL and FEDL do not consider the benefits of device selection to reduce the training latency, and FedCS randomly select wireless channels thus cannot exploit the benefits of wireless resources. The training latency of q-FFL and FEDL is generally longer than that of FedCS, which means that the wireless resource allocation can provide more benefits than the active devices allocation. Furthermore, it can be inferred that as the number of available channels increases, the training latency is reduced, and this reduction tends to converge. This is due to the less collision probability with more channels, until the channels are enough for reliable data uploading of the devices.

Impact of transmit power allocation. Figure 7 depicts the performance gain of the transmit power allocation. $RaFed-min$ and $RaFed-max$ denote the simplified version of RaFed without transmit power allocation, where the devices all use the minimum and maximum transmit power. It can be observed that RaFed without power allocation significantly increases the training latency, compared with RaFed with power allocation. Furthermore, RaFed with maximum transmit power shows shorter training latency than RaFed with minimum transmit power, which comes from the stronger received signal power at the receiver.

Impact of device selection. We evaluated the impact of device selection on the training latency with Figure 6. It can be seen that, the training latency can be reduced with more available devices, because with more available devices, there is more optimization space by considering the trade-off between the number of active devices and wireless collisions to reduce the training latency. Similarly with Figure 6, the training latency also tends to be stable as the number of available devices increases. This indicates that the number of devices is large enough to minimize the training latency, and cannot be further improved by continuing to increase this number. Figure 6 also shows that q-FFL and FEDL perform shorter training latency than FedCS, which is because the number of channels is not enough for FedCS to provide expected performance gain.

Average number of active devices. To further analyze the device allocation, Figure 8 shows the average number of active devices under different number of available devices. We can see that the average number of active devices increases with more available devices. This implies that RaFed can utilize the benefits of multiple front-end devices for faster training. Besides, the increasing of the number of active devices slows down after the number of available devices reaches 80, which indicates that the number of available devices is large enough for optimizing the training latency.

V. Related Work

In this section, we study the existing works on the resource allocation and wireless communication in federated learning.

Federated Learning in wireless networks. Federated Learning is gaining traction in the field of wireless networks. Chen et al. [22] considers the impact of wireless packet errors on FL performance, and pose an optimization problem of FL over wireless networks to minimize the training loss and improve the training accuracy. Yang et al. [23] optimize the

energy consumption on the devices in FL under the constraint of training latency.

To accelerate the FL global model training, existing work mainly focuses on improving the transmission efficiency for those devices uploading model updates. Considering the fading channels, Mohammad et al. [24] put forward an analog scheme to exploit gradient sparsification and achieve more efficient use of limited channel bandwidth. Yang et al. [25] leverage the signal superposition property of wireless multiple-access channel, and design a novel over-the-air computation approach to yield faster global model aggregation. Seungeun et al. [26] adopt Federated Distillation and propose FL framework to enable the devices to exchange the average output logit vectors and reduce communication cost. However, the above researches overlooked the impact of wireless interference and active devices allocation. Chen et al. [27] considers the wireless resource allocation and device selection scheme to reduce the training latency, while the solution first allocates the active devices selection, and then perform wireless resource allocation based on the device selection results. This paper considers the severe interference in IIoT scenarios, and jointly allocate the wireless resource, as well as the active devices to optimize the training latency of FL.

Device selection in Federated Learning. At the beginning of each global round, FL will select merely a fraction of clients to participate in model updating. In vanilla FL [5], the clients are selected randomly based on a goal count of participants. To mitigate communication overhead and improve the performance of FL, Nishio et al. [21] consider cellular bandwidth limitation and propose a resource-aware selection algorithm which maximizes the number of participants in each round with resource constraints. Luping et al. [28] identify if the model updates of the devices are relevant enough to the global tendency of model improvement, and exclude irrelevant updates before clients reporting them. This work aims to accelerate convergence of global model and thus reduces communication rounds. With the same intent, Wang et al. [7] offer a reinforcement learning based framework that intelligently selects devices participated in a round. The above researches consider the allocation of active devices, but ignore the impact of collisions between wireless transmissions, so RaFed jointly takes the allocation of wireless resources allocation and active devices into account, to reduce the training latency of FL.

VI. Discussion

RaFed is based on a static wireless network topology where the devices are pre-deployed and do not move. However, in some IIoT applications such as smart robotics, the mobility of devices will introduce more dynamics. We can analyze the pattern of device mobility and predict the packet reception ratio to adaptively reduce the training latency. Besides, RaFed is based on the conflict between the limited wireless resources and increasing number of devices. Future wireless network can use mmWave and THz bands that have abundant bandwidth, and the resource allocation reduces to the allocation of active devices. The rising problem is that IIoT devices with mmWave or THz bands would have shorter communication distance, so they would require more hops to reach the server, thus the active devices allocation should consider the impact of packet relaying and becomes more complex.

VII. Conclusion

In this paper, we proposed RaFed that reduces the training latency of FL in IIoT systems. Increasing the number of active devices can improve the training latency, as more local models can be trained to accelerate the training of global model. As the IIoT systems often consist of dense-deployed devices and limited communication/computation resources, increasing the number of active FL devices can easily cause serious interference and leads to prolonged training latency. RaFed characterizes FL's latency performance and we formulate the problem as an optimization problem, which is proved to be NP-hard. We then proposed a heuristic algorithm to select the most appropriate clients to achieve a good trade-off between the interference and training time. We conduct experiments using an RGB-D dataset of texture-less objects in an IIoT automatic sorting system. Experimental results showed that RaFed can achieve shorter training latency of FL in IIoT systems, compared to other existing methods.

VIII. Acknowledgments

This work was supported by National Key Research and Development Program of China (No. 2020YFE0200500 and No. 2017YFB1400102) and the National Natural Science Foundation of China (No. 61972075 and No. 61972074). Zhiwei Zhao is the corresponding author.

References

[1] S. Li, Q. Ni, Y. Sun, G. Min, and S. Al-Rubaye, "Energy-efficient resource allocation for industrial cyber-physical iot systems in 5g era," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 6, pp. 2618–2628, 2018.

[2] J. Huang, L. Kong, G. Chen, M.-Y. Wu, X. Liu, and P. Zeng, "Towards secure industrial iot: Blockchain system with credit-based consensus mechanism," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3680–3689, 2019.

[3] Andea, "Andea." [Online]. Available: https://www.andea.com/

[4] Marean, "Marean." [Online]. Available: https://mareana.com/

[5] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *PMLR Artificial Intelligence and Statistics*, 2017, pp. 1273–1282.

[6] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 2, pp. 1–19, 2019.

[7] H. Wang, Z. Kaplan, D. Niu, and B. Li, "Optimizing federated learning on non-iid data with reinforcement learning," in *Proceedings of IEEE INFOCOM*, 2020, pp. 1698–1707.

[8] Q. Zhang, L. T. Yang, Z. Chen, P. Li, and F. Bu, "An adaptive dropout deep computation model for industrial iot big data learning with crowdsourcing to cloud computing," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 4, pp. 2330–2337, 2018.

[9] P. C. M. Arachchige, P. Bertok, I. Khalil, D. Liu, S. Camtepe, and M. Atiquzzaman, "A trustworthy privacy preserving framework for machine learning in industrial iot systems," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 9, pp. 6092–6102, 2020.

[10] Y. J. Cho, J. Wang, and G. Joshi, "Client selection in federated learning: Convergence analysis and power-of-choice selection strategies," in *Proceedings of ICLR*, 2021.

[11] S. U. Stich, "Local sgd converges fast and communicates little," in *Proceedings of ICLR*, 2019.

[12] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing," *HotCloud*, vol. 10, no. 4-4, p. 19, 2010.

[13] N. H. Tran, W. Bao, A. Zomaya, N. M. NH, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *Proceedings of IEEE INFOCOM*, 2019, pp. 1387–1395.

[14] W. Gao, W. Du, Z. Zhao, G. Min, and M. Singhal, "Towards energy-fairness in lora networks," in *Proceedings of IEEE ICDCS*, 2019, pp. 788–798.

[15] D. G. Yoon, S. Y. Shin, W. H. Kwon, and H. S. Park, "Packet error rate analysis of ieee 802.11 b under ieee 802.15. 4 interference," in *IEEE Vehicular Technology Conference*, vol. 3, 2006, pp. 1186–1190.

[16] A. Jemmali, J. Conan, and M. Torabi, "Bit error rate analysis of mimo schemes in lte systems," in *International Conference on Wireless and Mobile Communications*, 2013, pp. 190–194.

[17] W. Shi, S. Zhou, and Z. Niu, "Device scheduling with fast convergence for wireless federated learning," in *Proceedings of IEEE ICC*, 2020, pp. 1–6.

[18] T. Hodaň, P. Haluza, Š. Obdržálek, J. Matas, M. Lourakis, and X. Zabulis, "T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects," *Proceedings of IEEE WACV*, 2017.

[19] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in *Proceedings of USENIX OSDI*, 2016, pp. 265–283.

[20] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," in *Proceedings of ICLR*, 2020.

[21] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *Proceedings of IEEE ICC*, 2019, pp. 1–7.

[22] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," *IEEE Transactions on Wireless Communications*, 2020.

[23] Z. Yang, M. Chen, W. Saad, C. S. Hong, and M. Shikh-Bahaei, "Energy efficient federated learning over wireless communication networks," *IEEE Transactions on Wireless Communications*, 2020.

[24] M. M. Amiri and D. Gündüz, "Federated learning over wireless fading channels," *IEEE Transactions on Wireless Communications*, vol. 19, no. 5, pp. 3546–3557, 2020.

[25] K. Yang, T. Jiang, Y. Shi, and Z. Ding, "Federated learning via over-the-air computation," *IEEE Transactions on Wireless Communications*, vol. 19, no. 3, pp. 2022–2035, 2020.

[26] S. Oh, J. Park, E. Jeong, H. Kim, M. Bennis, and S.-L. Kim, "Mix2fld: downlink federated learning after uplink federated distillation with two-way mixup," *IEEE Communications Letters*, vol. 24, no. 10, pp. 2211–2215, 2020.

[27] M. Chen, H. V. Poor, W. Saad, and S. Cui, "Convergence time optimization for federated learning over wireless networks," *IEEE Transactions on Wireless Communications*, 2020.

[28] L. Wang, W. Wang, and B. Li, "Cmfl: Mitigating communication overhead for federated learning," in *Proceedings of IEEE ICDCS*, 2019, pp. 954–964.