DetFed: Dynamic Resource Scheduling for Deterministic Federated Learning over Time-sensitive Networks

Dong Yang, *Member, IEEE*, Weiting Zhang, *Member, IEEE*, Qiang Ye, *Senior Member, IEEE*, Chuan Zhang, Ning Zhang, *Senior Member, IEEE*, Chuan Huang, Hongke Zhang, *Fellow, IEEE*, and Xuemin (Sherman) Shen, *Fellow, IEEE*

***Abstract*—In this paper, we present a three-layer (i.e., device, field, and factory layers) deterministic federated learning (FL) framework, named DetFed, which accelerates collaborative learning process for ultra-reliable and low-latency industrial Internet of Things (IoT) via integrating 6G-oriented Time-sensitive Networks (TSN). Utilizing dispersive local data, industrial IoT devices distributively train a deep neural network (DNN) model, and the updated model parameters are aggregated at their associated field servers every round or at a centralized factory server every a few rounds. Aiming at optimizing the learning accuracy of FL without affecting the co-transmission of burst traffic (e.g., safety-critical traffic), an integrated TSN is considered to establish connections among the three layers, where a cyclic queuing and forwarding mechanism is deployed in each switch to support deterministic model parameter transmission with microsecond-level delay and near-zero packet loss requirements. To improve the FL performance, we formulate a multi-objective stochastic optimization problem to simultaneously maximize the scheduling success ratio and learning accuracy while satisfying the deterministic requirements of delay, jitter, and packet loss. Since the objective function is implicit and the available time slots of the considered TSN in each FL round are temporally correlated, the problem is difficult to solve in real time. Therefore, we transform the problem into a Markov decision process formulation and propose a dynamic resource scheduling algorithm, based on deep reinforcement learning, to make optimal resource scheduling decisions while adapting to device heterogeneity and network dynamics. Experimental results based on real-world dataset demonstrate that the proposed DetFed significantly accelerates FL convergence and improves learning accuracy as compared to state-of-the-art benchmarks.**

***Index Terms*—Deterministic federated learning, industrial Internet of Things, co-transmission, resource scheduling, deep reinforcement learning.**

Dong Yang, Weiting Zhang, and Hongke Zhang are with the School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing 100044, China. E-mail: {dyang, wtzhang, hkzhang}@bjtu.edu.cn.

Qiang Ye is with the Department of Computer Science, Memorial University of Newfoundland, St. John's, NL A1B 3X5, Canada. E-mail: qiangy@mun.ca.

Chuan Zhang is with the School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing 100081, China. E-mail: chuanz@bit.edu.cn.

Ning Zhang is with the Department of Electrical and Computer Engineering, University of Windsor, Windsor N9B 3P4, Canada. E-mail: ning.zhang@uwindsor.ca.

Chuan Huang is with the Future Network of Intelligence Institute (FNii) and the School of Science and Engineering, the Chinese University of Hong Kong, Shenzhen 518172, China, and also with the Peng Cheng Laboratory, Shenzhen 518066, China. E-mail: huangchuan@cuhk.edu.cn.

Xuemin (Sherman) Shen are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo N2L 0B5, Canada. E-mail: sshen@uwaterloo.ca.

I. Introduction

THROUGH integrating emerging artifical intelligence techniques, sixth generation (6G)-enabled industial Internet of Things (IoT) is expected to realize advanced industiral automation [1]–[3]. In such a scenario, an unprecedented amount of data is generated by dispersed industrial IoT devices, which needs to be distributively processed at the network edge due to delay requirements and privacy regulations [4], [5]. Over the past few years, a considerable amount of research efforts aiming at satisfying these requirements have led to the emergence of many novel distributed learning paradigms, which can leverage a large amount of dispersive data and computing power resources without transmitting raw data across the network [6]–[8]. In particular, federated learning (FL) has been proposed to enable collaborative deep neural network (DNN) model training among devices and a centralized server via periodically exchanging model parameters (i.e., *FL traffic*), for the purpose of achieving high learning accuracy with preserved data privacy and reduced communication overhead [9], [10].

Operating FL over resource-constrained and time-sensitive industrial IoT faces a big challenge, that is, how to simultaneously transmit the FL traffic and traditional industrial traffic (e.g., time-triggered traffic, event-driven traffic, and best-effort traffic) over the same network and satisfy their diversified requirements [11]. In such a case, reliably aggregating the device-side model parameters in the centralized server is important to accelerate FL convergence [12]. Time-sensitive Networks (TSN), evolved from traditional industrial Ethernets, can provide multiple traffic flows with bounded end-to-end delay, jitter, and packet loss in local area networks, which is envisioned as a candidate for 6G-oriented ultra-reliable and low-latency communication techniques [13]–[15]. By introducing the synchronized time-scheduled approaches to traffic shaping mechanisms, such as cyclic queuing and forwarding (CQF) proposed by IEEE 802.1Qch, TSN is endowed with flexible temporal resource scheduling capabilities to support deterministic traffic transmission [16].

As shown in Fig. 1, the CQF mechanism is able to provide each traffic flow with predictable and strictly bounded hop-by-hop transmission delay via periodically exchanging the receiving and sending statuses of Ping-Pong queues, which are deployed on each switch port operated by a couple of gates (i.e., RX-gate and TX-gate) [17]. The gate operations are

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3303017

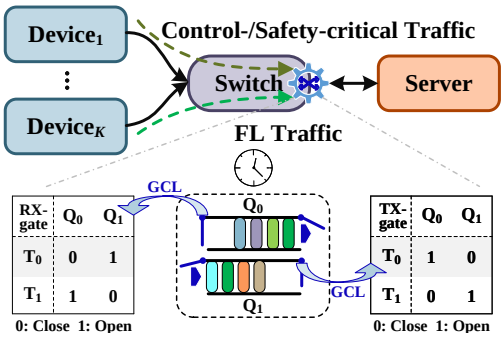


Fig. 1. CQF-based mechanism for deterministic traffic scheduling.

controlled by a gate control list (GCL) to forward traffic, which takes the scheduling cycle as an operation cycle. Assume that the scheduling cycle is divided into two time slots, e.g., an even time slot T_0 and an odd time slot T_1. In T_0, the receiving queue Q_1 opens the RX-gate to receive traffic from upstream nodes, and the sending queue Q_0 opens the TX-gate to send traffic to downstream nodes. In T_1, the operations are performed in reverse. With the CQF mechanism, the TSN can support deterministic transmission for FL traffic between any two adjacent nodes, such that the FL convergence can be efficiently accelerated. However, when multiple industrial IoT devices participate in FL, all the devices interact with the centralized server in a parallel manner, resulting in inefficient resource preemption and increased transmission delay especially when the number of devices is large. To accelerate FL convergence, how can we schedule multiple traffic flows to facilitate FL over industrial IoT in an efficient way needs further investigation [18].

In this work, we propose a novel deterministic FL framework, named DetFed, which accelerates collaborative FL process via 6G-oriented ultra-reliable and low-latency communication techniques. The DetFed is composed of three layers, i.e., device, field, and factory layers. To provide the learning process with deterministic guarantee in delay, jitter, and packet loss, a wireless and wired integrated TSN is utilized to establish connections among the three layers, in which each switch is deployed with a CQF mechanism to support efficient model parameter distribution and aggregation. Aiming at optimizing the learning accuracy without affecting the transmission of burst traffic (e.g., safety-critical traffic), device-side updated models are periodically aggregated by their associated field servers or by a centralized factory server over the wireless and wired integrated TSN. In this way, the amount of transmitted model parameters among the field servers and the factory server can be effectively reduced, hence saving bandwidth occupancy on the industrial buses that can be used to support burst traffic transmission. As a result, more model parameters can be aggregated at the two-level server sides within the hierarchical DetFed, which significantly improves the learning accuracy.

To further improve the performance of DetFed, we propose a learning-based dynamic resource scheduling algorithm over the wireless and wired integrated TSN. The increase of the number of devices and network nodes may lead to intolerable time overhead in resource scheduling, because traditional optimization methods and heuristic schemes cannot adapt to network dynamics, which requires repeatedly resource scheduling calculations when a new device participates in the DetFed thus slowing down the FL convergence. To address this challenging issue, we investigate a resource scheduling problem in DetFed, which is formulated as a multi-objective stochastic optimization problem for maximizing the learning accuracy as well as the scheduling success ratio by optimizing temporal resource allocation for the wireless and wired integrated TSN. Since the objective function of the DetFed is implicit and the available time slots of the considered TSN in each FL round are temporally correlated, the problem is difficult to solve on-the-fly. We transform the problem into a Markov decision process (MDP) formulation, and propose a dynamic resource scheduling algorithm, based on deep reinforcement learning (DRL), to solve the MDP. Extensive simulation results on real-world dataset with non-independent and identical distribution (non-IID) demonstrate that the proposed DetFed framework with the dynamic resource scheduling algorithm can significantly improve the learning accuracy as compared to the benchmarks. The main contributions of this paper are summarized as follows:

- We present a hierarchical deterministic FL framework for the time-sensitive industrial IoT, in which FL can efficiently operate with bounded end-to-end delay, jitter, and packet loss guarantee via temporal resource scheduling for the wireless and wired integrated TSN;
- We formulate the multi-dimensional resource scheduling as a multi-objective stochastic optimization problem to maximize the FL performance over the TSN with deterministic service quality guarantee, which is then transformed into an MDP;
- We propose a dueling double deep Q-network (D3QN) based resource scheduling algorithm to efficiently determine temporal resource allocation for FL traffic within the integrated TSN. After offline training, the proposed algorithm can facilitate FL convergence and adapt to network dynamics due to the traffic-by-traffic scheduling ability.

The reminder of this paper is organized as follows. Related works are reviewed in Section II. A deterministic FL framework is proposed in Section III. Section IV presents the system model, and formulates the optimization problem with multiple objectives. Section V transforms the formulated problem into an MDP, and proposes a dynamic resource scheduling algorithm. Section VI provides the simulation results, followed by the concluding remarks in Section VII.

II. Related Work

FL has gained significant attention from both industry and academia due to the significant growth in data traffic generated at the network edge. It is regarded as a key enabling technology to achieve ubiquitous intelligence in envisioned 6G networks [19], [20]. Extensive research efforts have been devoted to resource management to accelerate FL convergence [21]. Chen *et al.* [22] propose a joint learning and communications framework for facilitating FL over wireless networks,

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3303017

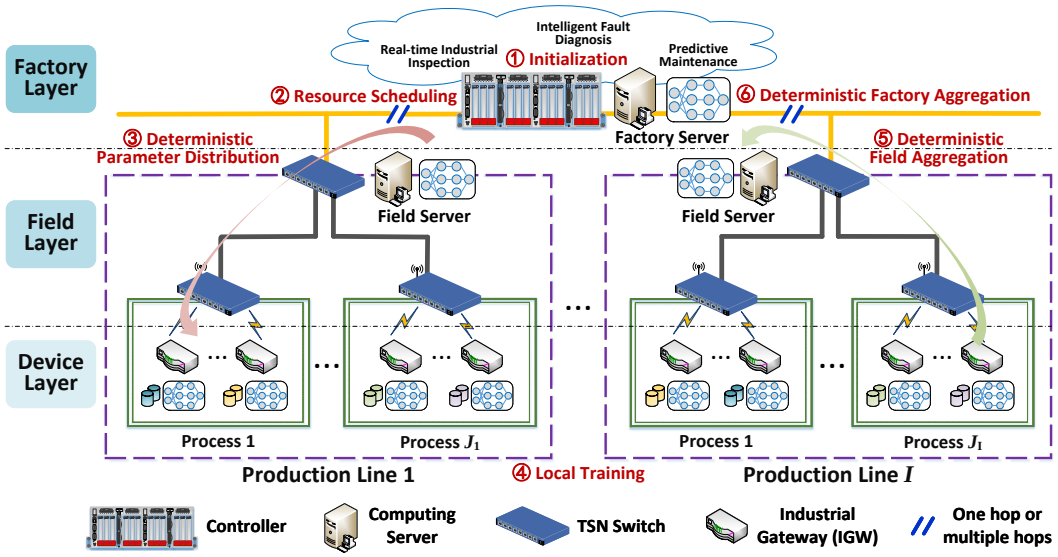


Fig. 2. A hierarchical deterministic FL framework for time-sensitive industrial IoT.

in which spectrum resource alloction and user selection are jointly optimized to achieve a significant FL accuracy improvement. Wan *et al*. [23] aim to minimize the time and energy consumption during FL process through optimizing spectrum bandwidth allocation and the number of local updates per iteration. T. Dinh *et al*. [24] investigate a heterogeneous computing resource optimization problem, in which the local computation rounds and the global communication rounds are scheduled to speed up FL convergence at the cost of on-device computing power and spectrum resources. Taking into account the unreliable and resource-constrained wireless networks, Salehi *et al*. [25] proposed a novel FL framework to ensure extremely high transmission success probability via dynamically scheduling the participating devices for each FL round. Zhang *et al*. [26] study the joint optimization of the power allocation and learning rate adjustment to obtain the best estimate of the gradient updates while minimizing the impact of the communication error. In summary, most of these works adopt frequency division multiplexing as a main communication mode to support FL model parameter distribution and aggregation. Due to the limited spectrum resources, such a mode is generally difficult to guarantee deterministic delivery of FL traffic. Thus, in large-scale device scenarios, only a part of devices can be selected to participate in model training or only a few updated parameters can be uploaded to the global server, which significantly slows down FL convergence speed, degrades learning accuracy, and occupies network resources.

Recently, a line of research works are proposed to improve transmission determinacy [27]. Atallah *et al*. [28] studies an iterated integer linear programming based scheduling scheme to attain high scheduling scalability, in which time-triggered traffic is divided into multiple subsets and each subset is scheduled incrementally. Yan *et al*. [29] proposes a CQF-based injection time planning algorithm to optimize the network throughput of time-sensitive traffic. Zhao *et al*. [30] considers a credit-based shaper to reduce the pessimism for the worst-case end-to-end delays of audio–video bridging traffic. More prominently, a few pioneering works focus on the co-transmission for multiple types of traffic. Yuan *et al*. [31] present a deadline monotonic scheduling algorithm to ameliorate the traffic transmission sequence of switch exports via dividing the queue priority into more levels, thereby satisfying the coordinated transmission requirements of time-triggered traffic and best-effort traffic. Huang *et al*. [17] propose a time-aware cyclic-queuing mechanism to support the co-transmission of cyclic flows and isochronous flows. To sum up, most of them are dedicated to supporting deterministic traffic transmission for traditional industrial applications (e.g., control- and/or safety-critical applications). How to provide industrial intelligent applications with deterministic transmission services, especially realizing the co-transmission of FL traffic and classical industrial traffic, is still a challenging issue needs to be solved.

Different from the existing works, our paper focuses on facilitating efficient FL in time-sensitive industrial IoT which is to ensure the end-to-end deterministic transmission of FL traffic among devices and servers. Furthermore, taking traffic time-varying property and network dynamics into account, we propose a learning-based temporal resource scheduling algorithm to accelerate FL convergence over a wireless and wired integrated TSN.

III. DetFed: A Deterministic FL Framework

A. Framework Design

In the considered time-sensitive industrial IoT, we adopt a hierarchical FL framework enhanced by a two-level aggregation scheme to facilitate efficient model training [21]. As shown in Fig. 2, the framework consists of the device, field, and factory layers. The detailed descriptions of each layer in the framework are given as follows.

- **Factory Layer**: A global server and a central controller are deployed at the factory management center. The global server is responsible for initializing a set of model parameters at the beginning according to the application

Table I
A SUMMARY OF IMPORTANT NOTATIONS.

Notation	Description
$\mathcal{I}$	Set of wired TSN switches
$\mathcal{J}_i$	Set of wireless TSN switches connected to wired switch i
$\mathcal{K}_j$	Set of IGWs associated with wireless TSN switch j
$\mathcal{S}_{i,j,k}$	Set of data samples held by IGW k
$\mathcal{S}_i$	Set of data samples held by IGWs covered by field server i
$\mathcal{S}_{i,j,k}^{train}$, $\mathcal{S}_{i,j,k}^{test}$	Training and testing sample sets of the IGWs
$\mathcal{S}_i^{train}$, $\mathcal{S}_i^{test}$	Training and testing sample sets covered by field server i
$\boldsymbol{u}_s, v_s$	Input and label of each sample
$\boldsymbol{\theta}_{i,j,k}$	Locally trained parameters of each IGW
$\boldsymbol{\theta}_i$	Global model parameters of field servers or factory server
$l((\boldsymbol{u}_s, v_s))$	Training loss for sample $(\boldsymbol{u}_s, v_s)$ held by the IGWs
η	Learning rate for FL
r_g	Number of field aggregations
$A_i(\boldsymbol{\theta}_i)$	Accuracy on testing samples of the IGWs under field server i
$A_{fi}(\boldsymbol{\theta}_i)$	Accuracy of the updated model after field aggregations
$A_{fa}(\boldsymbol{\theta})$	Accuracy of the updated model after factory aggregations
t_k	FL traffic k
SC	Scheduling cycle
$LCM(\cdot)$	Least common multiple function
T, T_{num}	Time slot size and time slot number
B^{acc}, B^{core}	Wireless and wired link bandwidths
$GCD(\cdot)$	Greatest common divisor function
p_k^x	The x-th packet of traffic t_k
C_s, C_s^{max}	The occupied resources and maximum capacity for each slot
T_k^{e2e}, J_k^{e2e}	End-to-end delay and jitter for each traffic
PL_k^{e2e}	End-to-end packet loss for each traffic
H_k	The number of network nodes that traffic t_k traverses
$A(\boldsymbol{\theta}^r)$, S^r	Learning accuracy and scheduling success ratio in round r
α	Temporal resource allocation decision
ξ	Weight coefficient for the objective function
s^t, a^t, z^t	Network state, decision, and agent reward
γ	Discount factor
$\boldsymbol{\pi_\omega}$	Policy parameterized by $\boldsymbol{\omega}$
$Q(\cdot)$	State-decision Q-value function
$V(\cdot)$, $A(\cdot)$	Value function and advantage function
$L(\boldsymbol{\omega})$, $\nabla_{\boldsymbol{\omega}} L(\boldsymbol{\omega})$	Loss function and its gradient with regards to $\boldsymbol{\omega}$
y^d	Target Q-value

requirements and performing factory-level aggregation for updated model parameters (i.e., FL traffic) from distributed field servers every a few field aggregations. The controller is in charge of collecting network information and enforcing traffic scheduling decisions, as well as coordinating FL among three layers with deterministic guarantee of delay, jitter, and packet loss.

- **Field Layer**: Multiple TSN switches are deployed to provide deterministic transmission services for industrial IoT devices, in which a part of TSN switches endowed with a wireless access module are deployed to connect the devices into the factory network. The rest of the TSN switches are connected to the controller. In addition, each production line is equipped with a field server to aggregate model parameters from industrial IoT devices every FL rounds.
- **Device Layer**: Industrial IoT devices endowed with certain storing and computing resources, such as industrial gateways (IGW), are distributively located in production lines within a smart factory. In each FL round, IGWs receive the global parameters that are transmitted from the factory server or field servers, and perform local training for further parameter updating.

The DetFed framework can support efficient FL with deterministic requirements through integrating delay-guaranteed TSN into a hierarchical aggregation scheme. On one hand, parameter aggregation and distribution are determinately coordinated among the three layers by accurate traffic scheduling. As such, more updated model parameters can be uploaded to the field servers or a factory server, thus accelerating FL convergence. On the other hand, model parameters are infrequently transmitted between the field and factory layers, thereby effectively avoiding bandwidth resource preemption with burst traffic in terms of factory bus.

B. Two-level Deterministic Aggregation Scheme

In the framework, we consider a set of TSN switches to support deterministic data transmission. The set of wired TSN switches is denoted by $\mathcal{I} = \{1, 2, \cdots, I\}$. The set of wireless TSN switches is denoted by $\mathcal{J}_i = \{1, 2, \cdots, |\mathcal{J}_i|\}$, which is connected to a wired TSN switch $i \in \mathcal{I}$. Let $\mathcal{K}_j = \{1, 2, \cdots, |\mathcal{K}_j|\}$ be the set of IGWs associated with wireless TSN switch $j \in \mathcal{J}_i$. For IGW k, a set of data samples are stored locally, denoted by $\mathcal{S}_{i,j,k} = \{(\boldsymbol{u}_s, v_s)\}_s$. Here, $\boldsymbol{u}_s$ and v_s are the input and the label of each sample, respectively, s is the sample index. The training and testing sample sets of the IGWs are denoted by $\mathcal{S}_{i,j,k}^{train}$ and $\mathcal{S}_{i,j,k}^{test}$, respectively. The training and testing sample sets of the IGWs within the coverage of field server i are denoted by $\mathcal{S}_i^{train}$ and $\mathcal{S}_i^{test}$, respectively. Here, we have $\mathcal{S}_{i,j,k}^{train} \cup \mathcal{S}_{i,j,k}^{test} = \mathcal{S}_{i,j,k}$, $\mathcal{S}_i^{train} \cup \mathcal{S}_i^{test} = \mathcal{S}_i$, and $\mathcal{S}_i = \cup_{j \in \mathcal{J}_i, k \in \mathcal{K}_j} \mathcal{S}_{i,j,k}$. A summary of important notations in this paper is given in Table I.

In the following, we elaborate the operation mechanism of the deterministic FL in detail, which is illustrated in Algorithm 1.

1) Initialization stage: Once the factory management center receives an FL task, the factory server will instantiate a set of parameters $\boldsymbol{\theta}^*$ for obtaining the required learning model, such as a convolutional neural network-based industrial inspection model that can provide real-time quality detection services for industrial machinery. ***2) Multi-dimensional resource scheduling***: After completing the model initialization stage, the central controller then makes network resource scheduling decisions for FL round $r \in \mathcal{R}$ to support deterministic model parameter transmission. Here, the decisions include the amounts of time slots in wireless and wired TSN that are allocated to transmit model parameters among the IGWs, field servers, and a factory server. With these decisions, the FL can efficiently operate by orchestrating the dispersed IGWs, and achieve a higher accuracy through aggregating more model parameters with deterministic service quality guarantee.

Algorithm 1: DetFed

◇ Initialization stage
Factory: Factory server initializes global DNN model parameter $\boldsymbol{\theta}^*$ with deterministic transmission requirements (i.e., end-to-end delay, jitter, and near-zero packet loss);
◇ Dynamic resource scheduling
Allocating temporal resources for $\mathcal{K}_j, \forall j \in \mathcal{J}_i, \forall i \in \mathcal{I}$ industrial IoT devices via dynamic resource scheduling to satisfy the deterministic transmission requirements;
for each $FL\ round\ r \in \mathcal{R}$ **do**
  ◇ Deterministic parameter distribution (**FL traffic**)
  Broadcast $\boldsymbol{\theta}_i$ from factory server or field servers to the connected industrial IoT devices via wireless and wired integrated scheduling;
  ◇ Local training (**heterogeneous computing power**)
  for each $TSN\ switch\ i \in \mathcal{I}$ *in parallel* **do**
    for each *device* $k \in \mathcal{K}_j$, $\forall j \in \mathcal{J}_i$ *in parallel* **do**
      for each *training sample* $s \in \mathcal{S}_{i,j,k}^{train}$ **do**
        Update the model parameters via a gradient descent method;
        $\boldsymbol{\theta}_{i,j,k} = \boldsymbol{\theta}_i - \eta \frac{\partial l((\boldsymbol{u}_s, v_s) | \boldsymbol{\theta}_{i,j,k})}{\partial \boldsymbol{\theta}_{i,j,k}}$;
  ◇ Deterministic field aggregation (**wireless scheduling**)
  if $mod(r, r_g) \neq 0$ **then**
    Aggregate device-side model parameters by their associated field servers every FL round;
    $\boldsymbol{\theta}_i = 1/\left|\mathcal{S}_{i,j,k}^{test}\right| \sum_{i\in\mathcal{I}, j\in\mathcal{J}_i, k\in\mathcal{K}_j} \left|\mathcal{S}_{i,j,k}^{test}\right| \boldsymbol{\theta}_{i,j,k}$;
    $A(\boldsymbol{\theta}) = A_{fi}(\boldsymbol{\theta}_i)$;
  ◇ Deterministic factory aggregation (**wired scheduling**)
  if $mod(r, r_g) = 0$ **then**
    Aggregate field server-side model parameters by a factory server every a few FL rounds;
    $\boldsymbol{\theta} = 1/\left|\mathcal{S}^{test}\right| \sum_{i\in\mathcal{I}} \left|\mathcal{S}_i^{test}\right| \boldsymbol{\theta}_i$;
    $A(\boldsymbol{\theta}) = A_{fa}(\boldsymbol{\theta})$;
return $\boldsymbol{\theta}$, $A(\boldsymbol{\theta})$.

3) Deterministic parameter distribution: Given the resource scheduling decisions, the aggregated model parameters can be distributed from the field servers or a factory server to the IGWs in each FL round. Due to adopting the deterministic traffic scheduling mechanism, the parameter distribution can have a high schedulability, such that more IGWs can receive the distributed model parameters to participate in the current FL round.

4) Local training: When the distributed model parameters $\boldsymbol{\theta}_i, \forall i \in \mathcal{I}$ are received, the IGWs can start local training using their sample sets for one round. To speed up convergence rate, we adopt a stochastic gradient descend method to iterate the model parameters, that is, $\boldsymbol{\theta}_{i,j,k} = \boldsymbol{\theta}_i - \eta \partial l((\boldsymbol{u}_s, v_s) \mid \boldsymbol{\theta}_{i,j,k}) / \partial \boldsymbol{\theta}_{i,j,k}$. Here, $\boldsymbol{\theta}_{i,j,k}$ is the locally trained parameters of each IGW, η is the learning rate, and $l((\boldsymbol{u}_s, v_s) \mid \boldsymbol{\theta}_{i,j,k})$ is the training loss with regards to each sample held by the IGWs.

5) Deterministic field aggregation: Once completing the local training, field servers will asynchronously aggregate the trained model parameters from the IGWs via deterministic resource scheduling, namely, $mod(r, r_g) \neq 0$. Meanwhile, the global model parameters $\boldsymbol{\theta}_i$ can be updated through a Federated Averaging algorithm that performs a weighted average calculation for all the aggregated parameters, i.e., $\boldsymbol{\theta}_i = 1/\left|\mathcal{S}_i^{test}\right| \sum_{i\in\mathcal{I}, j\in\mathcal{J}_i, k\in\mathcal{K}_j} \left|\mathcal{S}_{i,j,k}^{test}\right| \boldsymbol{\theta}_{i,j,k}$. Here, $\left|\mathcal{S}_{i,j,k}^{test}\right|$ and $\left|\mathcal{S}_i^{test}\right|$ are the numbers of testing samples locally stored at each IGW and at all IGWs associated with field server i. Thus, the learning accuracy of the updated model is given by

$$A_{fi}(\boldsymbol{\theta}_i) = \frac{1}{I} \sum_{i\in\mathcal{I}} A_i(\boldsymbol{\theta}_i), \tag{1}$$

where $A_i(\boldsymbol{\theta}_i)$ denotes the learning accuracy on testing samples of the IGWs that are associated with field server i in terms of its aggregated parameters $\boldsymbol{\theta}_i$, and $A_i(\boldsymbol{\theta}_i) = 1/\left|\mathcal{S}_i^{test}\right| \sum_{s\in\mathcal{S}_i^{test}} a((\boldsymbol{u}_s, v_s) \mid \boldsymbol{\theta}_i)$. Here, $a((\boldsymbol{u}_s, v_s) \mid \boldsymbol{\theta}_i)$ is the learning accuracy on each testing sample held by the IGWs. Then, the aggregated model parameters of field servers will be distributed to their associated IGWs for a new round update.

6) Deterministic factory aggregation: After r_g field aggregations, the factory server will aggregate model parameters from field servers via deterministic resource scheduling for the considered TSN networks, namely, $mod(r, r_g) = 0$. Then, the global model parameters can be updated via the Federated Averaging algorithm, i.e., $\boldsymbol{\theta} = 1/\left|\mathcal{S}^{test}\right| \sum_{i\in\mathcal{I}} \left|\mathcal{S}_i^{test}\right| \boldsymbol{\theta}_i$. The learning accuracy can be described as

$$A_{fa}(\boldsymbol{\theta}) = \frac{1}{\left|\mathcal{S}^{test}\right|} \sum_{s\in\mathcal{S}^{test}} a((\boldsymbol{u}_s, v_s) \mid \boldsymbol{\theta}), \tag{2}$$

where $\mathcal{S}^{test} = \cup_{i\in\mathcal{I}} \mathcal{S}_i^{test}$, $\left|\mathcal{S}^{test}\right|$ is the total number of testing samples stored at all the IGWs, and $a((\boldsymbol{u}_s, v_s) \mid \boldsymbol{\theta})$ is the learning accuracy on each testing sample of all the IGWs with parameters $\boldsymbol{\theta}$ aggregated by the factory server.

In the considered scenario, the proposed DetFed framework adopts *synchronous FL paradigm*, which is operated round-by-round. In each FL round, due to the heterogeneous computing resources of IoT devices, the delay for local training is different for each device. Whenever the local training stage is completed, each IoT device will send a signaling to its associated field server or the factory server. Once the signaling from the last participating IoT device is received, the field server will synchronously aggregate model parameters from their participating devices every FL round, and the factory server will aggregate model parameters from the edge servers every a few FL rounds. To maintain time synchronization within the considered network scenario, the clock synchronizing mechanism *IEEE 1588* is deployed at the TSN switches and IoT devices. As such, the FL traffic of heterogeneous IoT devices can be scheduled synchronously.

Computation complexity analysis: Different from the conventional FL framework that aggregates model parameters only by one centralized server, a set of additional field servers are deployed in the proposed DetFed framework to aggregate the uploaded model parameters from their connected IoT devices. The computation complexity of both frameworks is mainly composed of local training and model aggregation. For the local training stage, both frameworks have the same computation complexity when they train the same DNN model. For the model aggregation stage, the computation complexity

of the DetFed is slightly higher than the conventional FL. However, the model aggregation can be implemented via a low-complexity algorithm, e.g., FedAvg, in which model parameters are simply aggregated via a weighted average method. We assume that the training computation complexity is denoted as $\mathcal{O}^{tra}(\cdot)$, and that the computation complexity of operating the FedAvg is denoted as $\mathcal{O}^{agg}(\cdot)$. The aggregation computation complexity of the DetFed is denoted as $(I{+}1){\times}\mathcal{O}^{agg}(\cdot)$. Here, I is the number of field servers, and 1 indicates to the factory server. Thus, the total computation complexity of the DetFed is $\mathcal{O}^{tra}(\cdot){+}(I{+}1){\times}\mathcal{O}^{agg}(\cdot)$. Here, we have $\mathcal{O}^{tra}(\cdot){\gg}(I{+}1){\times}\mathcal{O}^{agg}(\cdot)$.

IV. System Model and Problem Formulation

In this section, an FL traffic model is presented, followed by a wireless and wired integrated scheduling model to support deterministic model parameter distribution and aggregation. Then, we formulate a multi-objective optimization problem with multi-dimensional resource constraints to optimize the FL performance while improving the scheduling capability of the considered TSN.

A. FL Traffic Model

As previously described, the FL is a long-term and iterative process, in which model parameters are interacted alternately among IGWs, field servers, and a factory server. In such scenarios, whether the model parameters can reliably transmit among the three layers is especially important to accelerate FL convergence. For parameter distribution, if the global model parameters can be distributed to more IGWs, the FL can take full advantage of geographically dispersed data to train the global model. For parameter aggregation, if more local trained parameters can be uploaded to the field servers or the factory server, the global model can achieve a faster convergence. Thus, to ensure the deterministic parameter transmission during the long-term learning process, we operate the hierarchical FL over a TSN-enhanced network scenario, and define the model parameters as *FL traffic* with deterministic end-to-end transmission requirements need to be guaranteed.

In the considered scenario, the transmitted FL traffic requires the bounded end-to-end delay, jitter, and packet loss. As such, each FL traffic can be described as

$$\begin{aligned} &t_k \triangleq \{id, cycle, src, dst, path, d_{e2e}, j_{e2e}, p_{num}, p_{size}, st\}, \\ &\forall t_k \in K, k = 1, 2, \ldots, |K|, \end{aligned} \tag{3}$$

where each traffic t_k has ten features, including identifier number, traffic cycle, source and destination addresses, preset transmission path, end-to-end delay and jitter requirements, packet number and size, and sending time that is a decision variable needs to be made by a resource scheduling algorithm. Here, the sending time of a traffic means the timestamp when the traffic is sent from a device's network interface card (NIC). In addition, $|K| = \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}_i}|\mathcal{K}_j|$ denotes the total number of FL traffic in each FL round.

B. Wireless and Wired Integrated Scheduling Model

As stated earlier, the FL operates round-by-round within the hierarchical framework. To ensure a high convergence efficiency, the FL traffic should be transmitted among the three layers. Taking into account the mobility, the industrial IoT devices are connected to the corresponding wireless TSN switches endowed with a wireless access module [32]. In addition to that, FL traffic are forwarded by a set of wired TSN switches. In each TSN switch, we deploy the CQF-based shaper to satisfy the deterministic transmission requirements, which is endowed with a queuing mechanism for shaping FL traffic to provide each traffic with bounded delay, jitter, and packet loss [33]. To implement high-performance TSN, network adapters that are programmable and support TSN functionality is required at each IoT device of the communication link. In this work, the industrial IoT devices are directly connected to the TSN switches and each IoT device is equipped with a network adapter (e.g., Intel I225-V) to support FL traffic transmission. In the following, several important constraints are defined:

*1) **Traffic scheduling cycle**:* In the CQF model, a pair of Ping-Pong queues are introduced at each switch port, and the open (i.e., sending) and closed (i.e., receiving) statuses of Ping-Pong queues are periodically exchanged to forward traffic that is controlled by GCLs, such that the hop-by-hop bounded delay can be guaranteed to provide FL traffic with deterministic transmission services [34]. Since the GCL calculations need to be cyclically executed for all FL traffic, scheduling cycle SC equals to the least common multiple of the set of traffic cycles, namely,

$$SC = LCM(K[cycles]), \tag{4}$$

where SC should be a constant value, $LCM(\cdot)$ is the function to return the least common multiple value, and $K[cycles]$ denotes the set of cycles of all FL traffic.

*2) **Time slot size**:* As defined by CQF, a scheduling cycle SC should be divided into several time slots with the same duration T. Thus, the number of time slots in each scheduling cycle is $T_{num} = SC/T$. To avoid data packets conflicting with each other in the same time slot, the value of T should be within $[T^{min}, T^{max}]$.

For the minimum value T^{min}, it means the considered integrated networks should be able to accommodate the largest size of the FL traffic, i.e.,

$$T^{min} = \tau_{send} + \tau_{prop} + \tau_{proc} + \tau_{queue} + \tau_{sync}, \tag{5}$$

where

$$\tau_{send} = \begin{cases} \frac{max(K[sizes])}{B^{acc}}, & \text{if } ACC = True, \\ \frac{max(K[sizes])}{B^{core}}, & \text{otherwise}, \end{cases} \tag{6}$$

where τ_{send}, τ_{prop}, τ_{proc}, τ_{queue}, and τ_{sync} denote the packet sending, propagating, processing, queuing, and synchronizing delays between any two adjacently connected nodes, respectively. In addition, $max(\cdot)$ is the function to return the largest value, $K[sizes]$ denotes the set of data sizes of all FL traffic, and $B^{acc} = R^{acc}\log_2\left(1 + p_{i,j}|h_{i,j,k}|^2/\mu^2\right), \forall i \in \mathcal{I}, j \in \mathcal{J}_i, k \in \mathcal{K}_j$ and B^{core} denote the wireless and wired link

bandwidths, respectively. Here, ACC is $True$ indicates that the traffic is transmited between an IGW and a wireless TSN switch. Also, R^{acc} is the frequency band of the wireless networks, $p_{i,j}$ is the transmission power, $|h_{i,j,k}|^2$ is the channel gain between wireless TSN switch j and IGW k, and μ^2 is the Gaussian noise.

For the maximum value T_{max}, it means the size of time slots should be larger than the greatest common divisor of the set of traffic cycles, i.e.,

$$T^{max} = GCD(K[cycles]), \tag{7}$$

where $GCD(\cdot)$ is the function to return the greatest common divisor value. In addition, the scheduling cycle should be divisible by T, i.e.,

$$SC \ \% \ T = 0. \tag{8}$$

*3) **Sending time of FL traffic**:* In each FL round, FL traffic is transmitted among IGWs, field servers, and a factory server bottom up or top down. In either case, however, the sending time of each FL traffic should be smaller than its traffic cycle. In this way, packets of different traffic cycles can be prevented from preempting limited NIC resources in the meantime. Hence,

$$0 \leq t_k[st] < t_k[cycle], \forall k \in K. \tag{9}$$

Note that the sending time $t_k[st]$, namely injecting packets of traffic k into which time slots within each scheduling cycle, is determinied by a centralized network controller that is in charge of global resource scheduling. In addition, similar to the scheduling cycle, the traffic cycle should also be divisible by T, i.e.,

$$t_k[cycle] \ \% \ T = 0, \forall k \in K. \tag{10}$$

*4) **FL traffic transmission**:* With the sending time decisions, the occupied temporal resources for each time slot of the current scheduling cycle can be calculated, that is,

$$C_s = \sum_{k \in K} \sum_{x \in t_k[p_{num}]} \Omega_s(p_k^x) \cdot t_k[p_{size}], \forall s \in [1, T_{num}], \tag{11}$$

where $\Omega_s(\cdot)$ is an indicator function, and p_k^x is the x-th packet of traffic t_k. If $\Omega_s(p_k^x) = 1$, the packet p_k^x is injected into time slot s, otherwise 0. In particular, each packet can only be injected into no more than one time slot, i.e.,

$$\sum_{s \in T_{num}} \Omega_s(p_k^x) \leq 1, \forall k \in K, x \in [1, t_k[p_{num}]]. \tag{12}$$

Note that since all packets stored in the sending buffer need to be sent before the start of the next time slot, the size of the total packets within each time slot cannot exceed its maximum capacity. Hence,

$$0 \leq C_s \leq C_s^{max}, \forall s \in [1, T_{num}], \tag{13}$$

where C_s^{max} is the maximum capacity of time slot s. Note that the deterministic transmission guarantee in TSN is to reserve bandwidth resources for traffic in advance. For FL traffic, the traffic information can be available in advance and a temporal resource allocation policy is determined accordingly for transmission. However, for burst traffic, it is often difficult to determine a dynamic resource allocation policy to support real-time traffic due to information uncertainty. To ensure that the reserved resources are sufficient while avoiding resource wasting, we consider to reserve bandwidth in a statistical way by considering the long-term traffic features to accommodate the burst traffic, and the rest of link bandwidth resources are utilized by FL traffic. As such, FL traffic can be co-transmitted with burst traffic within the considered TSN scenario.

*5) **Multiple traffic requirements**:* Based on the above definitions, the end-to-end delay T_k^{e2e}, jitter J_k^{e2e}, and packet loss PL_k^{e2e} for each traffic can be obtained, namely,

$$\begin{aligned}
&T_k^{e2e} = (H_k + 1) \cdot T, \forall k \in K, \\
&J_k^{e2e} = \left| p_k^x[d_{e2e}] - \frac{1}{t_k[p_{num}]} \sum_{x \in t_k[p_{num}]} p_k^x[d_{e2e}] \right|, \forall k \in K, \\
&PL_k^{e2e} = \left| t_k[p_{num}] - \sum_{x \in t_k[p_{num}]} \sum_{s \in T_{num}} \Omega_s(p_k^x) \right|, \forall k \in K,
\end{aligned} \tag{14}$$

where H_k is the number of network nodes that traffic t_k traverses. Note that the jitter of each traffic is the delay variance of all its packets. Here, $p_k^x[d_{e2e}]$ denotes the end-to-end delay of the x-th packet of traffic t_k, and $1/t_k[p_{num}] \sum_{x \in t_k[p_{num}]} p_k^x[d_{e2e}]$ is the average delay of all packets for traffic t_k. The jitter of each traffic requires the gap between both values do not exceed the required jitter. The definition of packet loss is the same. Only when the above requirements are satisfied simultaneously, the FL traffic then can be transmitted with a deterministic service quality guarantee.

C. Multi-objective Optimization Problem Formulation

Since the learning accuracy of server-side models and the scheduling capability of the integrated networks are mutually reinforcing, both metrics should be jointly considered to maximize the overall system performance. In particular, the presented FL framework requires making the best time slot allocation decisions for deterministic model parameter transmission over the wireless and wired integrated TSN. To this end, we formulate the multi-objective resource scheduling problem as:

$$\mathcal{P}: \max_{\boldsymbol{\alpha}} \ \xi \left| A(\boldsymbol{\theta}^r) \right| + (1 - \xi) \left| S^r \right|$$

$$\begin{aligned}
\text{s.t.} \quad & T_k^{e2e} \leq t_k[d_{e2e}], \forall k \in K, && \text{(15a)} \\
& J_k^{e2e} \leq t_k[j_{e2e}], \forall k \in K, && \text{(15b)} \\
& PL_k^{e2e} \leq \delta, \forall k \in K, && \text{(15c)} \\
& T \in [T^{min}, T^{max}], && \text{(15d)} \\
& (4), (8), (9), (10), (12), \text{and } (13),
\end{aligned}$$

where $\boldsymbol{\alpha} = \{\alpha_{i,j,k}\}_{\forall i \in \mathcal{I}, j \in \mathcal{J}_i, k \in \mathcal{K}_j}$ denotes the temporal resource allocation decision, and $\alpha_{i,j,k} \in [1, T_{num}]$ is a discrete value which corresponds to the sending time of each FL traffic defined in Eq. (9). In addition, $|A(\boldsymbol{\theta}^r)| \in [0, 1], \forall r \in \mathcal{R}$ denotes the learning accuracy of the global model with respect to the aggregated parameters $\boldsymbol{\theta}^r$, and $|S^r| \in [0, 1]$ denotes the

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3303017

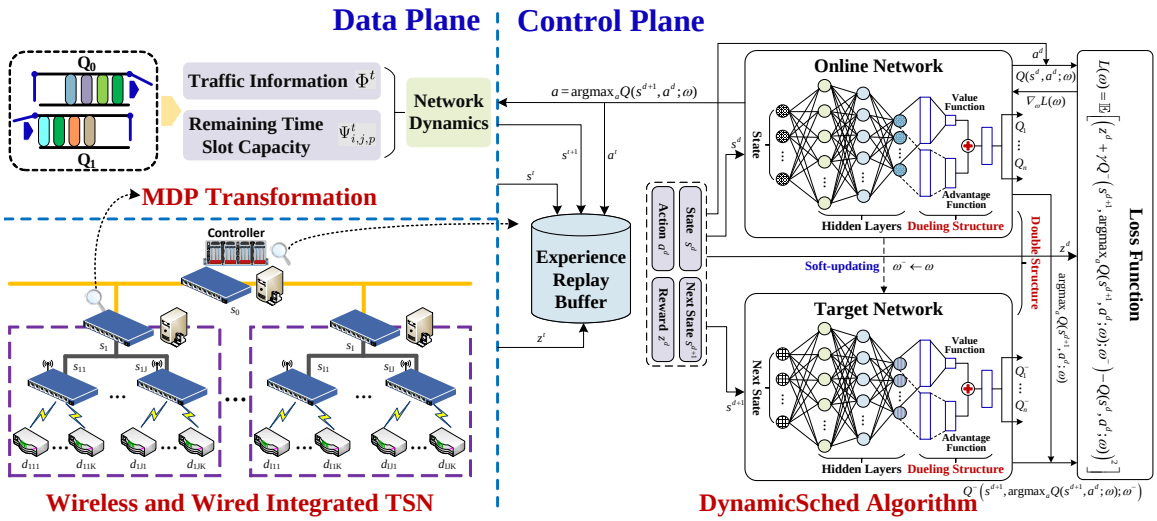


Fig. 3. An illustration of the proposed DynamicSched algorithm.

scheduling success ratio in FL round r which is the ratio of the number of successfully scheduled traffic to the number of all traffic. In the proposed DetFed framework, scheduling success ratio and learning accuracy are both important to accelerate FL convergence. A high scheduling success ratio is helpful to aggregate more updated model parameters from the distributed IoT devices, and a higher learning accuracy indicates a better collaborative learning performance. This paper aims to jointly optimize both objective components to efficiently obtain the optimal FL model parameter $\boldsymbol{\theta}^{\star}$. Thus, the weight coefficient $\xi \in (0,1)$ is to balance the importance between both objective components. Constraints (15a), (15b), and (15c) guarantee the traffic requirements can be satisfied to provide FL traffic with deterministic transmission support. Here, δ is a tiny constant value. Constraint (15d) ensures the time slot duration T of the considered integrated network is set within a valid range, and constraints (4), (8), (9), (10), (12), and (13) guarantee proper network resource scheduling.

The *relationship* between the resource allocation decision and the objective function is as follows. An appropriate decision α is helpful to distribute the global FL model from the field servers or factory server to more IoT devices and to aggregate more updated FL models from the IoT devices, thereby obtaining a higher scheduling success ratio $|S^r|$. Moreover, with decision α, more dispersed data held by the IoT devices can be utilized to sufficiently optimize the FL model parameters $\boldsymbol{\theta}$ in the local model training stage, which further improves learning accuracy $|A(\boldsymbol{\theta}^r)|$.

Problem $\mathcal{P}$ is a *multi-objective stochastic* optimization problem. The problem is "multi-objective" because the learning accuracy and scheduling success ratio are jointly considered to achieve a faster FL convergence. The problem is "stochastic" due to the resource scheduling decisions are determined in presence of temporal dynamics of available time slot resource during the FL process. In addition, the first term of $\mathcal{P}$ (i.e., learning accuracy) is expressed by an implicit function, and it is difficult to accurately characterize the mapping relationship between the decision variables and objective function, which further complicates decision making process. In particular, the available time slot capacity constraint is coupled with temporal resource scheduling decisions in terms of time and the amount of the scheduled traffic, which however have to be determined without foreseeing future network dynamics. Furthermore, according to the definitions of optimization components, the objective function is non-convex. Since optimizing variable α is integer, problem $\mathcal{P}$ is an integer optimization problem with a non-convex objective function. As such, the optimal variable α cannot be directly obtained via existing optimization methods. To approach the optimal solution, we consider DRL-based approaches to design a dynamic resource scheduling algorithm, which can make the optimal temporal resource allocation decisions and thus solving the formulated problem in an efficient way.

V. DynamicSched: A Dynamic Resource Scheduling Algorithm for FL Traffic

In this section, we first transform the multi-objective optimization problem into an MDP, and then propose a model-free DRL-based resource scheduling algorithm, named DynamicSched, to solve it. The computational complexity analysis of the DynamicSched is also provided.

A. MDP Transformation

MDP is a temporal decision making process, which is generally utilized to simulate the stochastic policies and rewards of learning agents in the environment with Markov property. Here, Markov property refers to that given the present state and all the past states of a random process, the conditional probability distribution of the next state depends only on the present state. In this paper, the formulated problem $\mathcal{P}$ is to design a dynamic resource scheduling policy to maximize the learning accuracy of server-side models and the scheduling capability of the considered networks while satisfying constraints, which falls into the MDP. To optimize FL performance over the time-sensitive industrial IoT, we reformulate the

resource scheduling problem $\mathcal{P}$ as an MDP. As shown in Fig. 3, we adopt a fully-centralized architecture proposed by IEEE 802.1Qcc in the control plane, and the central controller is modeled as an agent [35]. In each FL round, the agent can observe the traffic information and network state $s^t, t \in [1, |K|]$, and take temporal resource allocation decision $a^t, t \in [1, |K|]$ traffic-by-traffic. Here, $|K| = \sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}_i} \mathcal{K}_j$ is the total number of FL traffic in each FL round. This means that in the MDP, the time index t denotes a traffic index from the traffic set that needs to be scheduled. To learn an optimal policy $\boldsymbol{\pi}^\star_{\boldsymbol{\omega}}$ parameterized by $\boldsymbol{\omega}$, that can accurately map state $s^t, t \in [1, |K|]$ to decision $a^t, t \in [1, |K|]$, the corresponding reward $z^t, t \in [1, |K|]$ is fed back from the network environment, and the state is transformed into new state $s^{t+1}, t \in [1, |K|]$ accordingly [36]. In the MDP, three core elements, i.e., state, action (i.e., decision), and reward, are defined as follows.

- **State**: In FL round r, the agent collects traffic information Φ^t and the remaining time slot capacities $\Psi^r_{i,j,p}$ from the industrial IoT devices and TSN switches. Then, the state $s^t \in \mathcal{S}$ of the considered TSN can be defined as

$$s^t = \{\Phi^t, \Psi^t_{i,j,p}\}, \forall k \in \mathcal{K}_j, j \in \mathcal{J}_i, i \in \mathcal{I}, \tag{16}$$

  where $\Phi^t \triangleq \{t_k\}_{k\in\mathcal{K}_j}$ denotes the features of traffic t_k that is received at round r, and $\Psi^t_{i,j,p}$ denotes the remaining capacities of each receiving queue on port $p \in P$ for each time slot within a scheduling cycle.
- **Action**: Based on the observed state and the parameterized policy, the agent can make the temporal resource allocation decisions to determine which time slot is scheduled to transmit t_k, thereby supporting deterministic transmission for FL traffic. Namely,

$$a^t = \begin{cases} \underset{a^t_\varepsilon \in \mathcal{A}}{\arg\max}\, Q(s^t, a^t_\varepsilon), & \text{if } \varepsilon \geq \varepsilon_0 - N, \\ a^t_{rand}, & \text{otherwise}, \end{cases} \tag{17}$$

  Each action element is an interger value reshaped into a range of $[0, t_k[cycle]/SC - 1]$. The action indicates which time slot the FL traffic should be injected. Note that each traffic can only be injected into one time slot to be transmitted. Here, $\varepsilon_0 \in (0, 1]$ is an initial probability value, and N is a decay factor for each learning step.
- **Reward**: Once the agent takes action a^t, it will obtain a reward to evaluate how good the action is under state s^t. Aiming at maximizing the learning accuracy and scheduling success ratio simultaneously, the reward function of the DynamicSched algorithm can be given by

$$z^t = \begin{cases} V_{succ} + Z_{fi}(\boldsymbol{\theta}_i) \cdot \mathbb{1}\{mod(r, r_g) \neq 0\}, \text{if } success, \\ V_{succ} + Z_{fa}(\boldsymbol{\theta}) \cdot \mathbb{1}\{mod(r, r_g) = 0\}, \text{if } success, \\ V_{fail}, \qquad\qquad\qquad\quad \text{if } failed, \end{cases} \tag{18}$$

  where

$$Z_{fi}(\boldsymbol{\theta}_i) = \xi \left|A_{fi}(\boldsymbol{\theta}^r_i)\right| + (1-\xi)\left|S^r\right|, \forall i \in \mathcal{I}, \tag{19}$$

  and

$$Z_{fa}(\boldsymbol{\theta}) = \xi \left|A_{fa}(\boldsymbol{\theta}^r)\right| + (1-\xi)\left|S^r\right|. \tag{20}$$

  There are three cases for the reward. When the current FL traffic is scheduled successfully, the agent will obtain a positive reward, i.e., $V_{succ} + Z_{fi}(\boldsymbol{\theta}_i), \forall i \in \mathcal{I}$ or $V_{succ} + Z_{fa}(\boldsymbol{\theta})$, which are consistent with the two-level aggragation scheme in DetFed. Here, V_{succ} is a positive constant value, and $A_{fi}(\boldsymbol{\theta}^r_i), \forall i \in \mathcal{I}$ and $A_{fa}(\boldsymbol{\theta}^r)$ denote the learning accuracies on the model parameters after field aggregation (i.e., $\mathbb{1}\{mod(r, r_g) \neq 0\}$) and factory aggregation (i.e., $\mathbb{1}\{mod(r, r_g) = 0\}$), which are given by Eq. (1) and Eq. (2), respectively. On the contrary, the agent will obtain a negative constant value V_{fail} to penalize the inappropriate decisions. Obviously, the reward is related to the average learning accuracy and scheduling success ratio in the current FL round. With the obtained reward, the parameterized policy $\boldsymbol{\pi}_{\boldsymbol{\omega}}$ can be optimized during the training stage until algorithm converges.

In the MDP, the goal of the agent is to find the optimal temporal resource allocation policy $\boldsymbol{\pi}^\star_{\boldsymbol{\omega}} \in \Pi$ that can maximize cumulative discounted reward, namely,

$$\mathcal{P}' : \max_{\boldsymbol{\pi}_{\boldsymbol{\omega}}\in\Pi} \mathbb{E}\left[\sum_{\rho=0}^{\infty} \gamma^\rho \boldsymbol{z}^{t+\rho} | (s^t, a^t; \boldsymbol{\pi}_{\boldsymbol{\omega}})\right]$$

$$\text{s.t.} \quad (4), (8), (9), (10), (12), (13), \text{and } (15a) - (15c), \tag{21a}$$

where $\gamma \in [0, 1]$ denotes the discount factor. Note that policy $\boldsymbol{\pi}_{\boldsymbol{\omega}}$ specifies the way the central controller adopted to schedule temporal resources according to the observed states in each scheduling cycle.

B. D3QN-based Resource Scheduling Algorithm

The motivation to solve the MDP problem via deep RL algorithms are due to the following two aspects: (1) In the considered TSN scenario, it is difficult to obtain the complete information on state transition probabilities which is required to apply MDP algorithms (e.g., Relative Value Iteration) due to large state and action spaces; (2) The problem size is large, the convergence of conventional value iteration algorithms is time consuming [37]. However, deep RL algorithms can solve the MDP problem by offline training and online implementation with a low latency even when the problem size is large, which can be more efficiently applied in practical systems. Compared to traditional DQN-based algorithms, the D3QN algorithm can significantly reduce overestimation for resource scheduling decisions. Moreover, the D3QN algorithm can identify the correct resource scheduling decision more quickly during policy evaluation and achieve much better decision performance via a *dueling* DQN module. The above designs contribute to better approximation of the optimal state values via frequent updating of the value stream, thus improving the stability of resource scheduling optimization especially when the number of scheduling decisions is large. Thus, we consider the D3QN algorithm to solve the MDP. As shown in Fig. 3, we propose a learning-based dynamic resource scheduling algorithm based on D3QN, named DynamicSched. In the algorithm, a *double* DQN and a *dueling* DQN modules are

adopted to overcome the decision overestimation problem and achieve much better performance [38], [39]. For the double DQN module, an online network is to evaluate the $\varepsilon-greedy$ policy, and a target network is to estimate its value. With this design, the original *max* operation in the target Q-value estimation of DQN algorithms can be decomposed into decision selection and decision evaluation, therefore effectively reducing overestimation for decisions. For the dueling DQN module, the lower layers of online and target networks adopt fully-connected (FC) structures parameterized by $\boldsymbol{\varphi}$ and $\boldsymbol{\varphi}'$, respectively, while the following layers of both networks are endowed with two streams of FC layers instead of the original single stream. Here, the two streams represent the value function $V(s^t;\boldsymbol{\omega},\boldsymbol{\varphi}')$ and advantage function $A(s^t,a^t;\boldsymbol{\omega},\boldsymbol{\varphi})$ that sharing a common feature learning layers, that is, the lower layers of online and target networks. In the output aggregating layer, the two streams are combined to produce a single state-decision Q-value function, namely,

$$Q(s^t,a^t;\boldsymbol{\omega},\boldsymbol{\varphi},\boldsymbol{\varphi}') = V(s^t;\boldsymbol{\omega},\boldsymbol{\varphi}')+ \left(A(s^t,a^t;\boldsymbol{\omega},\boldsymbol{\varphi}) - \frac{1}{|\mathcal{A}|}\sum_{a^t} A(s^t,a^t;\boldsymbol{\omega},\boldsymbol{\varphi})\right), \quad (22)$$

where $V(\cdot)$ is an one-dimension scalar, and $A(\cdot)$ is an $|\mathcal{A}|$-dimension vector. Here, $|\mathcal{A}|$ is the dimension of decision a^t. With this improvement, the value and advantage functions can be estimated separately, such that the effect of state s^t and decision a^t on Q-value estimation can be taken into account simultaneously and the correct decision can be identified more quickly during policy evaluation. Note that the better decision a^t, the greater the advantage. For notation simplicity, we omit $\boldsymbol{\varphi}$ and $\boldsymbol{\varphi}'$ in the following context.

In this subsection, we illustrate the D3QN-based dynamic resource scheduling algorithm as the following steps, which is presented in Algorithm 2.

1) Algorithm initialization: Once the training stage is started, the parameters of the online and target networks of the centralized learning agent will be initialized and then updated step-by-step. Note that only one FL traffic is scheduled in each learning step, and thus the size of the learning step equals to the total number of the traffic that needs to be scheduled. Additionally, an experience replay buffer $\mathcal{B}$ is instantiated with capacity D to store the sampled experiences that can be utilized to update the algorithm parameters. Then, the considered network environment is reset to generate an initial state s^1.

2) Experience sampling: When the initialization stage is completed, the learning process of the agent can start by sampling interacted experiences from the network environment and storing them into replay buffer $\mathcal{B}$, which can be denoted as a five-element tuple, i.e., $\{s^t,a^t,z^t,s^{t+1},done\}$. Specifically, the learning agent makes temporal resource allocation decision a^t with regard to current state s^t. Note that decision $a^t=\alpha_{i,j,k},\forall i\in\mathcal{I},j\in\mathcal{J}_i,k\in\mathcal{K}_j$ is made by alternating two policies, i.e., $\varepsilon-greedy$ policy and *random* policy. Then, decision a^t is executed to allocate time slot resources for current FL traffic t_k, and an immediate reward z^t can be obtained from the network environment via performing Algorithm 1. As such, the network state will be transformed into new state s^{t+1}, and the information on the available time slot resources of the considered TSN can be obtained for scheduling the next traffic. In addition, $done\in\{True,False\}$ label is utilized to indicate whether the time slot resources are exhausted up to now. The corresponding experience tuple is stored in replay buffer $\mathcal{B}$ for algorithm training, and it will replace the oldest tuple when the replay buffer is full.

Algorithm 2: DynamicSched

Input: Traffic information and remaining time slot capacities $\{\Phi^t,\Psi^t_{i,j,p}\},\forall i\in\mathcal{I},j\in\mathcal{J}_i,p\in\mathcal{P}$;
Output: Optimal temporal resource allocation policy $\boldsymbol{\pi}^\star_{\boldsymbol{\omega}}$ that can make decision a^t traffic-by-traffic;

Initialize online and target networks with parameters $\boldsymbol{\omega}$ and $\boldsymbol{\omega}^-$, and initialize experience replay buffer $\mathcal{B}$ to capacity D;
for *episode* $e\in[1,|E_{num}|]$ **do**
  Reset network environment and obtain initial state s^1;
  for *step* $t\in[1,|K|]$ **do**
    ◇ Environment sampling
    Obtain random decision a^t_{rand} with probability ε, otherwise obtain $\mathrm{argmax}_a Q^\star(\phi(s^t),a^t_\varepsilon;\boldsymbol{\pi}_{\boldsymbol{\omega}})$ with $\epsilon-greedy$ policy;
    Execute decision a^t (i.e., allocate time slot resource for current FL traffic t_k) and obtain an immediate reward z^t;
    Transform to new state s^{t+1} to obtain the available time slot capacities of the considered wireless and wired integrated TSN networks for the next traffic;
    Store $\{s^t,a^t,z^t,s^{t+1},done\}$ into $\mathcal{B}$, and replace the oldest experiences if $|\mathcal{B}|>D$;
    ◇ Neural network training
    Sample a random minibatch of D_b experiences $\{s^d,a^d,z^d,s^{d+1},done\}$ from $\mathcal{B}$;
    Set the target Q-value for each experience $y^d=z^d+\gamma Q^-(s^{d+1},\mathrm{argmax}_a Q(s^{d+1},a^d;\boldsymbol{\omega});\boldsymbol{\omega}^-)$;
    Update $\boldsymbol{\omega}$ by performing a gradient descent step and minimizing the loss function $L(\boldsymbol{\omega})=\mathbb{E}_{\{s^d,a^d,z^d,s^{d+1},done\}\backsim\mathcal{B}(D)}[(y^d-Q(s^d,a^d;\boldsymbol{\omega}))^2]$;
    Set $\boldsymbol{\omega}^-\leftarrow\boldsymbol{\omega}$ via a soft-updating method to update target network;
    if *done* **then** break.

3) Neural network training: As previously described, the online and target networks are trained using the sampled experiences. In each algorithm step, the learning agent randomly selects a minibatch of D_b experiences from the replay buffer and updates the parameters of both networks.

Firstly, to update the online network, the loss function can be expressed by

$$L(\boldsymbol{\omega}) = \mathbb{E}_{\{s^d,a^d,z^d,s^{d+1}\}\backsim\mathcal{B}(D)}\left[(y^d-Q(s^d,a^d;\boldsymbol{\omega}))^2\right], \quad (23)$$

where

$$y^d = z^d+\gamma Q^-\left(s^{d+1},\mathrm{argmax}_a Q(s^{d+1},a^d;\boldsymbol{\omega});\boldsymbol{\omega}^-\right), \quad (24)$$

is the target Q-value. Here, if $done=True$, then $y^d=z^d$. Then, parameter $\boldsymbol{\omega}$ of the online network can be trained step-by-step via minimizing the loss function, i.e.,

$$\nabla_{\boldsymbol{\omega}} L(\boldsymbol{\omega}) = \mathbb{E}_{\{s^d,a^d,z^d,s^{d+1}\}\backsim\mathcal{B}(D)}[(y^d-Q(s^d,a^d;\boldsymbol{\omega}))\cdot \nabla_{\boldsymbol{\omega}} Q(s^d,a^d;\boldsymbol{\omega})]. \quad (25)$$

Table II
SIMULATION PARAMETERS [36], [40].

DetFed Param.	Value	DynamicSched Param.	Value
Optimizer	Adam	*Optimizer*$^-$	Adam
Round	300	$\lvert E_{num}\rvert$	800
S_{size}	23,436	$\lvert K\rvert$	66
Batch	64	*Batch*$^-$	128
(AWE_{in}, AWE_{out})	(1024, 6)	*Buffer*	10,000
($CWRU_{in}$, $CWRU_{out}$)	(4096, 10)	γ	0.95
lr	2.5×10^{-3}	lr^-	1×10^{-4}
λ	8×10^{-5}	ε	0.5

Table III
ACRNN STRUCTURE [41].

Layer Name	Kernal Size	Activation Func.	Param. #
Conv-1	(5, 5, 1, 6)	*ReLU*	156
Maxpool-1	(1, 2, 2, 1)	–	12
Conv-2	(5, 5, 6, 16)	*ReLU*	1,516
Maxpool-2	(1, 2, 2, 1)	–	32
Flatten	(400, 1)	–	–
FC	(400, 16)	*tanh*	6,400
Forward GRU	(32, 1)	*tanh/sigmoid*	3,264
Backward GRU	(32, 1)	*tanh/sigmoid*	3,264
Attention	(64, 1)	*tanh*	65
Output	(64, 6)	*Softmax*	390

Secondly, to guarantee the stability of the training process, a soft-updating method is adopted, and the parameters $\boldsymbol{\omega}^-$ of the target network are slowly updated by tracking the online network, namely,

$$\boldsymbol{\omega}^- = \kappa\boldsymbol{\omega} + (1-\kappa)\boldsymbol{\omega}^-, \tag{26}$$

where $\kappa \in (0, 1]$ is the update ratio of the target network.

Thirdly, the DynamicSched adopts the $\varepsilon - greedy$ policy to keep balance between exploitation and exploration for making accurate decision a^t, in which the possibility factor ε decreases after each learning step, i.e., $\varepsilon = \varepsilon_0 - N$.

Remark1: Since the number of input neurons of the agent is related to the devices' number, the network topology and device number cannot change during the training process. If new devices apply to participate in the DetFed framework, the number of input neurons of the agent should be adjusted accordingly, and the proposed DynamicSched algorithm needs to be retrained according to the new network topology. After a certain number of iterations, the updated scheduling decisions are obtained and can be deployed in the central controller to support DetFed operation.

Computation complexity analysis: As shown in Fig. 3, the DynamicSched algorithm is composed of an online network and a target network, and both of them are instantiated by an FC neural network. The computation complexity of the proposed resource scheduling algorithm is mainly correlated to the FC neural networks and the learning process. The computation complexity of an FC neural network is generally denoted by $\mathcal{O}(\sum_l (2M_{l-1}-1)M_l)$ [42], where $l \in [1, N]$ is the index of hidden layers and M_l is the corresponding neuron number. Notably, taking into account the dueling structure is adopted in the DynamicSched algorithm, the output layer consists of two parts, i.e., state value function and state-dependent decision advantage function, and hence $M_N = M_N^{state} + M_N^{decision}$. Here, M_N^{state} and $M_N^{decision}$ are the dimensions of the above two functions. For the learning process, the computation complexity is correlated to the step number $|K|$ and episode number $|E_{num}|$. To sum up, the total computation complexity of the DynamicSched can be denoted as $|K| \times |E_{num}| \times 2\mathcal{O}(\cdot)$.

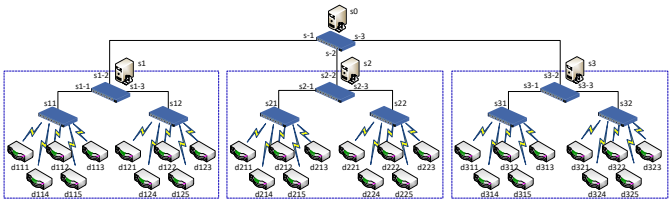


Fig. 4. The considered network topology in the experiment.

The *relationship* between Algorithm 1 and Algorithm 2 is as follows. Overall, both algorithms are utilized to support the proposed DetFed framework, and Algorithm 1 requires the scheduling decision made by Algorithm 2 in each FL round. Specifically, Algorithm 1 is to accelerate collaborative DNN model training process, which is operated round-by-round. Algorithm 2 is proposed to make network resource allocation decisions, which is operated episode-by-episode. In each episode, a learning agent deployed in a central controller acquires network states and makes decisions to support deterministic model parameter transmission during the operation of Algorithm 1.

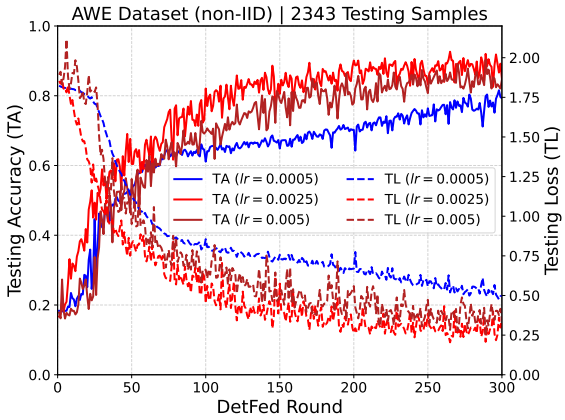
VI. PERFORMANCE EVALUATION

In this section, extensive simulations are carried out to evaluate the proposed solutions for accelerating and optimizing FL over the wireless and wired integrated TSN.

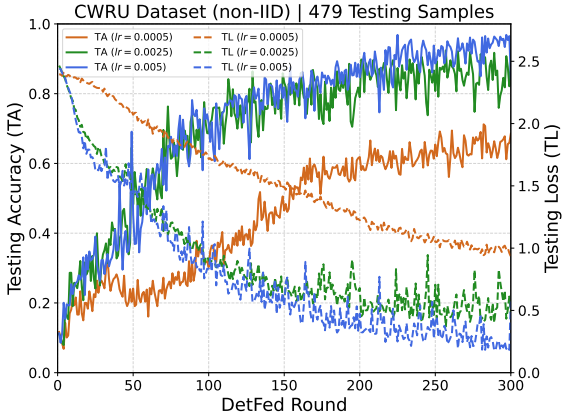
A. Experimental Setup

The network topology is shown as Fig. 4, where 30 devices, 3 field servers, and a factory server are considered to support DetFed operation which are connected as a tree topology by 10 TSN switches. Thus, there are 66 FL traffic needs to be scheduled within one scheduling cycle (i.e., one FL round).[1] The link bandwidth for the wireless and wired integrated TSN is set to 1 Gbit/s unless specified. Note that 90% of the bandwidth is utilized to support DetFed operation, and 10% of the bandwidth is reserved to transmit other industrial traffic, such as control- and/or safety-critical traffic. The propagating

[1]For model parameter distribution, three FL traffic are distributed from the factory server to the field servers, and thirty FL traffic are distributed from the field servers to the devices. For model parameter uploading, FL traffic is uploaded in the opposite direction, and the number of FL traffic is the same.



(a) AWE dataset



(b) CWRU dataset

Fig. 5. Testing accuracy and loss of DetFed with respect to different learning rates.

delay τ_{prop}, processing delay τ_{proc}, queuing delay τ_{queue}, and synchronizing delay τ_{sync} are set to 10 µs. The traffic period $K[periods]$ is set to 40 ms, and thus the scheduling cycle SC is 40 ms. The size of time slots within each scheduling cycle is set to 10 ms. For the DynamicSched algorithm, the online and target networks are with [S_{dim}, 1000, 500, A_{dim}] neurons, respectively. Here, S_{dim} is the dimension of the observed environment state, and A_{dim} is the time slot number of the considered network scenario. The update factor τ of target network is set to 0.005, which is decided by the scheduling cycle SC and the time slot size. The reward factor V_{succ} and penalty factor V_{fail} are set to be 0.1 and -0.1, respectively. Other important simulation parameters are listed in Table II.

We consider an attention-enhanced convolutional recurrent neural network (ACRNN), which contains 10 layers, i.e., two convolutional (Conv) layers, two Maxpool layers, a flatten layer, an FC layer, a forward gated recurrent unit (GRU) layer, a backward GRU layer, an attention layer, and a softmax output layer. The detailed FL model information is shown in Table III. The total trainable model parameters are around 15,099. Here, each parameter is quantized to 4 B, and thus the data size of the model parameters $K[sizes]$ is around 58.98 KB.

In the simulation, we utilize two vibration-based datasets to train the ACRNN model: (1) *AWE dataset*[2], in which each data sample is a vibration signal segment attached with a label to represent the health status of an industrial equipment, such as "Normal" and "Fault-*6mm*"; and (2) *CWRU dataset*[3], in which each data sample is also a vibration signal segment attached with a label to indicate the fault location, such as "Inner Race" and "Outer Race". In both datasets, the number of data samples are 23,436 and 479, respectively, and the ratio of training set to testing set is 9:1. Notably, the considered ACRNN model should be trained on a non-independent and identical distribution (non-IID) dataset, which is dispersed at the distributed participating devices and each device has only a few classes of data samples. Specifically, the samples of both datasets are sorted by their data label, respectively, which are further divided into 60 subsets with the same sample volume. Then, each device randomly selects 2 subsets as its local training set.

The proposed solutions are compared to the following benchmark schemes.

- ***Centralized Training (CT)***: This scheme requires uploading all the dispersed data to the factory server to conduct model training. Although it can achieve the optimal learning accuracy, the data privacy cannot be preserved effectively.
- ***The DetFed without Field Aggregation (DetFed w.o.)***: In this scheme, all the updated model parameters are only aggregated by the factory server, which occupies more bandwidth resources of industrial buses.
- ***Double Deep Q-Network (DDQN)***: In this scheme, the decisions are dynamically made via a DDQN algorithm, in which the Q-network is endowed with two fully-connected hidden layers and each layer is with 1000 neurons. Other learning parameters are the same as the DynamicSched.
- ***Random-based Scheme (RS)***: This scheme adopts random policy to allocate temporal resources for each FL traffic, and all available decisions are made with an equal probability.

B. Evaluation of DetFed Scheme

1) Learning performance: Fig. 5(a) shows the convergence performance of the proposed DetFed scheme with respect to different learning rates for the ACRNN-based model. When the learning rate $lr = 0.0025$, the proposed scheme achieves the highest testing accuracy and the lowest testing loss, respectively, although it tends to decline around 60 rounds before converging to the optimal accuracy. In addition, it can be seen that a larger learning rate (e.g., $lr = 0.005$) may incur the training process because the optimal value of accuracy might be skipped, and thus leading to an inferior convergence performance. Although a small learning rate helps stablizing the training process, the ACRNN-based model may be trapped in a local optimum, and consequently may need to consume more temporal resources to support model parameter transmission

[2]Online available. https://github.com/Intelligent-AWE/DeepHealth.

[3]Online available. https://csegroups.case.edu/bearingdatacenter/pages/download-data-file.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3303017

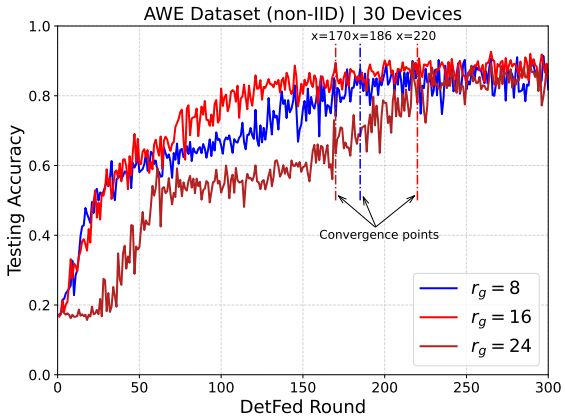


Fig. 6. Testing accuracy of DetFed with respect to different factory aggregation frequencies.

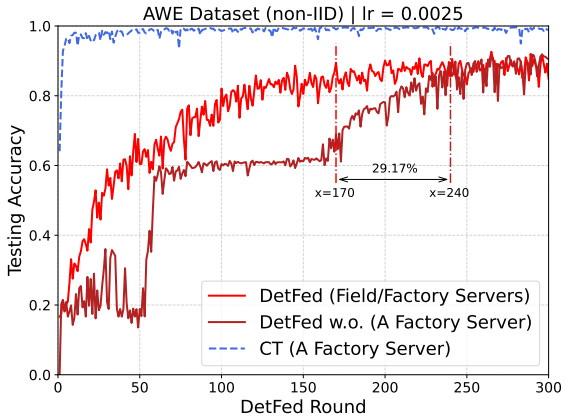


Fig. 7. Testing accuracy comparison among the proposed scheme and benchmarks with respect to different parameter aggregation locations.

and achieve model convergence. A similar simulation is carried out on the CWRU dataset. As shown in Fig. 5(b), the highest testing accuracy and the lowest testing loss can be obtained when the learning rate is 0.005.

2) Impact of factory aggregation interval: Fig. 6 shows the convergence performance of the proposed DetFed scheme with respect to different factory aggregation intervals. In the DetFed framework, the factory aggregation is performed every r_g FL rounds, in which the parameters aggregated at the field servers are further aggregated at a factory server. We can obtain several important observations from the simulation results. Firstly, when the factory aggregation interval r_g is set to 16, the highest accuracy on the AWE dataset can be obtained despite an inferior training performance is exhibited before 40 rounds. Secondly, if we set the factory aggregation interval r_g as an appropriate value, the training process can be effectively accelerated. Specifically, the proposed scheme with $r_g = 16$ achieves the fastest convergence after 170 training rounds, which decreases convergence round number by 16 and 50 compared with $r_g = 8$ and $r_g = 24$, respectively. Thirdly, the proposed DetFed scheme achieves almost the same testing accuracy at the end of the training process. This indicates that the TSN-enhanced DetFed scheme can provide efficient and deterministic transmission services for model parameter aggregation and distribution among the three layers.

3) Impact of aggregation node: As shown in Fig. 7, we evaluate the testing accuracy of the proposed DetFed scheme and benchmarks in terms of different parameter aggregation locations. The result shows that all the schemes can achieve model convergence within 300 training rounds. Intuitively, the CT scheme only takes a few training rounds to converge (around 50 rounds) and can converge to an especially high level (i.e., $> 99\%$). This is because a centralized dataset has been stored at the factory server that were collected from the dispersed industrial IoT devices before training. Such a scheme indeed can obtain a satisfactory convergence performance without parameter aggregation demand, but it inevitably exists the problems of data transmission overhead and data privacy leakage which are critical issues in industrial IoT scenarios. Although the distributed learning schemes, namely DetFed

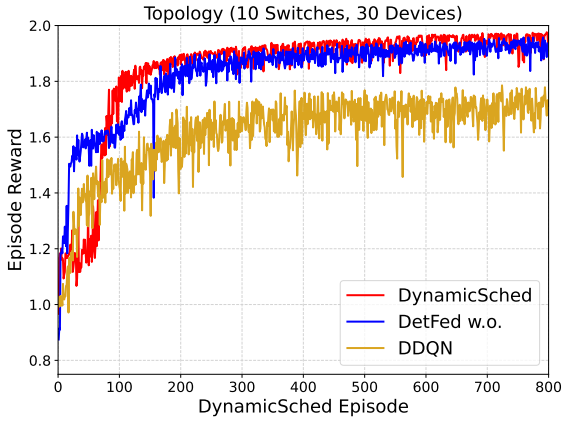


Fig. 8. Convergence performance comparison among different algorithms.

and DetFed w.o., only converge to a relatively low level (i.e., $\approx 90\%$), they can effectively save bandwidth resources of industrial buses and preserve privacy for industrial raw data. In particular, the DetFed scheme can reduce the number of training rounds until model convergence by 29.17% compared with the DetFed w.o. scheme. The reason is that directly aggregating model parameters at the factory server may lead to an unstable learning process due to the non-IID data distribution, which impedes the ACRNN model to obtain the global optimal parameters and slows down the convergence speed. On the contrary, the proposed DetFed scheme stabilizes the learning process by alternately aggregating model parameters among a factory server and field servers, such that the global optimal parameters can be efficiently obtained based on local optimal parameters and thus accelerating model convergence.

As shown in Fig. 6 and Fig. 7, the proposed DetFed framework can effectively accelerate the collaborative training process in the considered network topology. However, this result is achieved under the assumption that the IoT connections are stable. In harsh network scenarios, where serious interference and electrical noises exist, and the network bandwidth resources utilization is not high, the transmission delay for FL model parameters may not be satisfied. In particular,

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3303017

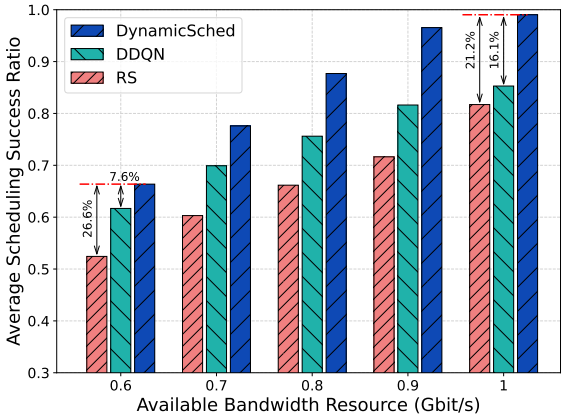


Fig. 9. Average scheduling success ratio comparison among the proposed scheme and benchmarks with respect to different bandwidth resources.

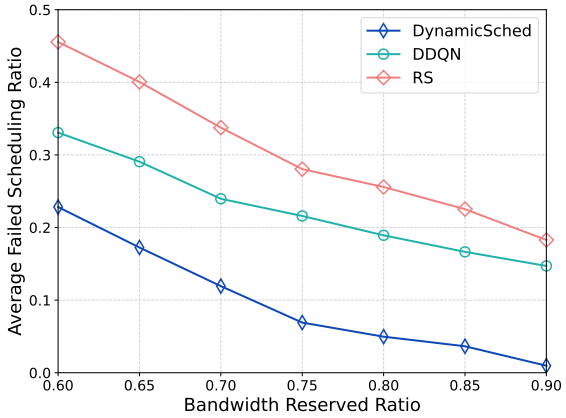


Fig. 11. Average failed scheduling ratio comparison among the proposed scheme and benchmarks with respect to different bandwidth reservation ratios.

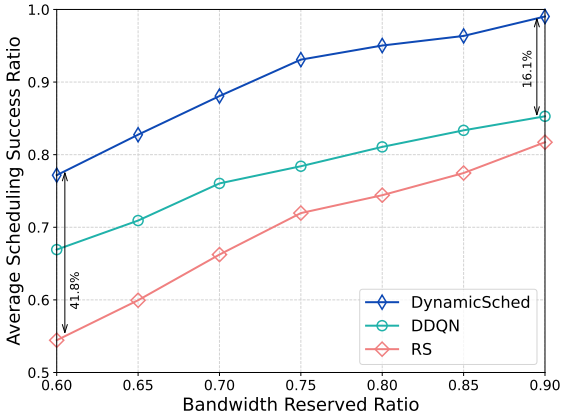


Fig. 10. Average scheduling success ratio comparison among the proposed scheme and benchmarks with respect to different bandwidth reservation ratios.

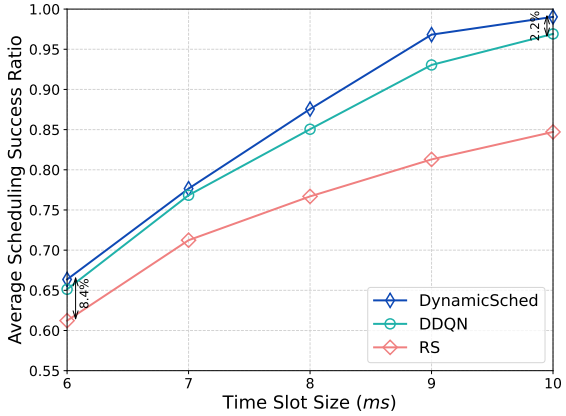


Fig. 12. Average scheduling success ratio comparison among the proposed scheme and benchmarks with respect to different time slot sizes.

when some IoT devices are disconnected, the updated model parameters will not be uploaded to the aggregation servers, which further slows down the convergence rate.

C. Evaluation of DynamicSched Algorithm

1) Convergence performance: As shown in Fig. 8, the convergence performance of the DynamicSched with respect to episodes is evaluated. It can be seen that all the learning-based resource scheduling algorithms have converged. In particular, the DynamicSched achieves a higher episode reward compared with other benchmarks, which indicates that the proposed DynamicSched is capable of making judicious allocation decisions for temporal resources of the considered TSN. This is because the DynamicSched adopts a D3QN-based architecture, in which an online-target *double* module and a value-advantage *dueling* module are utilized to improve algorithm performace. Specifically, the dueling module is helpful to capture which network states are (or are not) useful without having to learn the effect of each resource allocation decision for each network state, and thus identifying the appropriate decision more judiciously.

2) Impact of bandwidth resources: Fig. 9 shows average scheduling success ratio with respect to different bandwidth resources over 100 simulation runs. The result shows that the proposed algorithm can effectively improve average scheduling success ratio compared with the benchmark algorithms, which indicates that temporal resource allocation is optimized. In specific, when the amount of bandwidth resource is 1 Gbit/s, the proposed DynamicSched improves the average scheduling success ratio by 16.1% and 21.2% compared with the DDQN and RS benchmarks, respectively. This is because the proposed learning-based algorithm can allocate temporal resources for FL traffic more reasonably, which is helpful to satisfy the deterministic transmission requirements of FL traffic. In addition, the performance gain achieved in bandwidth-sufficient scenarios (e.g., 1 Gbit/s) is larger than that in bandwidth-limited scenarios (e.g., 0.6 Gbit/s), as compared to the DDQN benchmark. The result indicates that integrating the dueling module into the double module can enhance the decision-making capability of the proposed algorithm for temporal resource allocation, thereby improving the scheduling success ratio, especially when the bandwidth resource is sufficient.

3) Impact of reserved bandwidth ratio: Fig. 10 shows the average scheduling success ratio in terms of different bandwidth reservation ratios. Taking the value of 0.9 as an example, when the bandwidth reservation ratio is set to 0.9,

90% of bandwidth resources are utilized to support DetFed operation and 10% of bandwidth resources are reserved to support the co-transmission of burst traffic. As expected, the DynamicSched algorithm can effectively improve the average scheduling success ratio as compared with the benchmarks. This is because the proposed algorithm can better allocate bandwidth resources to support FL traffic transmission with satisfied deterministic requirements of delay, jitter, and packet loss. In addition, with the increase of the bandwidth reservation ratios, the average scheduling success ratios achieved by the DynamicSched, DDQN, and RS algorithms increase. In particular, the proposed algorithm improves the average scheduling success ratio by 15.3% to 16.1%, and by 41.8% to 21.2%, as compared with the DDQN and RS benchmarks, respectively. The underlying reason is that more available bandwidth resources can be utilized to determinately transmit FL traffic over the considered TSN scenario. To demonstrate the algorithm performance in the case of failed scheduling, we have presented the average failed scheduling ratio of the proposed DynamicSched algorithm with respect to bandwidth reserved ratios. As shown in Fig. 11, the DynamicSched achieves the lowest average failed scheduling ratio compared to all the benchmarks. In addition, the failed scheduling ratio decreases with the amounts of reserved bandwidth resources. The result indicates that the DynamicSched can achieve good performance due to effective resource utilization.

4) Impact of time slot sizes: Fig. 12 shows the impact of the time slot sizes on scheduling performance. The result shows that the learning-based algorithms, i.e., DynamicSched and DDQN, can achieve higher average scheduling success ratio than that of RS benchmark. The reason is that learning-based algorithms enable the deterministic requirements of FL traffic to be satisfied efficiently via judicious temporal resource scheduling. Moreover, with the increase of the time slot sizes, the average scheduling success ratio of the compared algorithms increase. This is because that more available temporal resources can be utilized to schedule FL traffic to satisfy their deterministic delay, jitter, and packet loss requirements. In particular, the proposed DynamicSched algorithm can achieve the highest average scheduling success ratio than the benchmarks. When the size of time slot is 10 ms, the DynamicSched can increase the average scheduling success ratio by 2.2% and 16.9%, as compared with DDQN and RS benchmarks, respectively.

VII. Conclusion

We have presented a new deterministic FL framework (i.e., DetFed) for the time-sensitive industrial IoT in combination with 6G-oriented ultra-reliable and low-latency communication techniques. We have proposed a DRL-based resource scheduling algorithm to make the optimal temporal resource allocation decisions for facilitating efficient DetFed, which aggregates more device-side models for obtaining an accurate global model. Experimental results based on real-world non-IID dataset have demonstrated that the proposed framework can significantly accelerate FL convergence and improve learning accuracy as compared with state-of-the-art benchmarks. By adopting the wireless and wired integrated TSN and the CQF mechanism, the DetFed can optimize the learning accuracy of FL without affecting the co-transmission of burst traffic. For the future work, we will study a mixed and online traffic scheduling scheme for facilitating efficient FL over a large-scale and distributed computing power network.

References

[1] H. Kang, Z. Li, and Q. Zhang, "Communicational and computational efficient federated domain adaptation," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 12, pp. 3678–3689, Dec. 2022.

[2] X. Shen, J. Gao, W. Wu, K. Lyu, M. Li, W. Zhuang, X. Li, and J. Rao, "AI-assisted network-slicing based next-generation wireless networks," *IEEE Open Journal of Vehicular Technology*, vol. 1, no. 1, pp. 45–66, Jan. 2020.

[3] B. Yang, X. Cao, J. Bassey, X. Li, and L. Qian, "Computation offloading in multi-access edge computing: A multi-task learning approach," *IEEE Transactions on Mobile Computing*, vol. 20, no. 9, pp. 2745–2762, Sept. 2021.

[4] L. U. Khan, W. Saad, Z. Han, E. Hossain, and C. S. Hong, "Federated learning for Internet of things: Recent advances, taxonomy, and open challenges," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1759–1799, June 2021.

[5] C. Zhang, M. Zhao, L. Zhu, W. Zhang, T. Wu, and J. Ni, "FRUIT: A blockchain-based efficient and privacy-preserving quality-aware incentive scheme," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 12, pp. 3343–3357, Dec. 2022.

[6] J. Kang, X. Li, J. Nie, Y. Liu, M. Xu, Z. Xiong, D. Niyato, and Q. Yan, "Communication-efficient and cross-chain empowered federated learning for artificial intelligence of things," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 5, pp. 2966–2977, Sept. 2022.

[7] Z. Qu, S. Guo, H. Wang, B. Ye, Y. Wang, A. Y. Zomaya, and B. Tang, "Partial synchronization to accelerate federated learning over relay-assisted edge networks," *IEEE Transactions on Mobile Computing*, vol. 21, no. 12, pp. 4502–4516, Dec. 2022.

[8] W. Y. B. Lim, Z. Xiong, C. Miao, D. Niyato, Q. Yang, C. Leung, and H. V. Poor, "Hierarchical incentive mechanism design for federated machine learning in mobile networks," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9575–9588, Oct. 2020.

[9] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Robust and communication-efficient federated learning from Non-i.i.d. data," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 9, pp. 3400–3413, Sept. 2020.

[10] Q. Tang, F. R. Yu, R. Xie, A. Boukerche, T. Huang, and Y. Liu, "Internet of intelligence: A survey on the enabling technologies, applications, and challenges," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 3, pp. 1394–1434, May 2022.

[11] E. Li, F. He, Q. Li, and H. G. Xiong, "Bandwidth allocation of stream-reservation traffic in TSN," *IEEE Transactions on Network and Service Management*, vol. 19, no. 1, pp. 741–755, Mar. 2022.

[12] M. Chen, D. Gündüz, K. Huang, W. Saad, M. Bennis, A. V. Feljan, and H. V. Poor, "Distributed learning in wireless networks: Recent progress and future challenges," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3579–3605, Dec. 2021.

[13] J. Prados-Garzon, T. Taleb, and M. Bagaa, "Optimization of flow allocation in asynchronous deterministic 5G transport networks by leveraging data analytics," *IEEE Transactions on Mobile Computing*, Early Access, 2021, DOI: 10.1109/TMC.2021.3099979.

[14] M. Vlk, Z. Hanzálek, K. Brejchová, S. Tang, S. Bhattacharjee, and S. Fu, "Enhancing schedulability and throughput of time-triggered traffic in IEEE 802.1Qbv time-sensitive networks," *IEEE Transactions on Communications*, vol. 68, no. 11, pp. 7023–7038, Nov. 2020.

[15] J. Prados-Garzon, T. Taleb, and M. Bagaa, "LEARNET: Reinforcement learning based flow scheduling for asynchronous deterministic networks," in *Proc. IEEE ICC 2020,* Dublin, Ireland, pp. 1–6, 2020.

[16] Z. Pang, X. Huang, Z. Li, S. Zhang, Y. Xu, H. Wan, and X. Zhao, "Flow scheduling for conflict-free network updates in time-sensitive software-defined networks," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 3, pp. 1668–1678, Mar. 2021.

[17] Y. Huang, S. Wang, B. Wu, T. Huang, and Y. Liu, "TACQ: Enabling zero-jitter for cyclic-queuing and forwarding in time-sensitive networks," in *Proc. IEEE ICC,* Montreal, QC, Canada, pp. 1–6, 2021.

[18] D. Yang, Z. Cheng, W. Zhang, H. Zhang, and X. Shen, "Burst-aware time-triggered flow scheduling with enhanced multi-CQF in time-sensitive networks," *IEEE/ACM Transactions on Networking*, Early Access, 2023, DOI: 10.1109/TNET.2023.3264583.

[19] X. Shen, J. Gao, W. Wu, M. Li, C. Zhou, and W. Zhuang, "Holistic network virtualization and pervasive network intelligence for 6G," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 1, pp. 1–30, Dec. 2021.

[20] M. N. Nguyen, N. H. Tran, Y. K. Tun, Z. Han, and C. S. Hong, "Toward multiple federated learning services resource sharing in mobile edge networks," *IEEE Transactions on Mobile Computing*, vol. 22, no. 1, pp. 541–555, Jan. 2023.

[21] W. Zhang, D. Yang, W. Wu, H. Peng, N. Zhang, H. Zhang, and X. Shen, "Optimizing federated learning in distributed industrial IoT: A multi-agent approach," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3688–3703, Dec. 2021.

[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*, vol. 20, no. 1, pp. 269–283, Jan. 2021.

[23] S. Wan, J. Lu, P. Fan, Y. Shao, C. Peng, and K. B. Letaief, "Convergence analysis and system design for federated learning over wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3622–3639, Dec. 2021.

[24] C. T. Dinh, N. H. Tran, M. N. H. Nguyen, C. S. Hong, W. Bao, A. Y. Zomaya, and V. Gramoli, "Federated learning over wireless networks: convergence analysis and resource allocation," *IEEE/ACM Transactions on Networking*, vol. 29, no. 1, pp. 398–409, Feb. 2021.

[25] M. Salehi and E. Hossain, "Federated learning in unreliable and resource-constrained cellular wireless networks," *IEEE Transactions on Communications*, vol. 69, no. 8, pp. 5136–5151, Aug. 2021.

[26] J. Zhang, N. Li, and M. Dedeoglu, "Federated learning over wireless networks: A band-limited coordinated descent approach," in *Proc. IEEE INFOCOM,* Vancouver, BC, Canada, pp. 1–10, 2021.

[27] D. Yang, K. Gong, J. Ren, W. Zhang, W. Wu, and H. Zhang, "TC-Flow: Chain flow scheduling for advanced industrial applications in time-sensitive networks," *IEEE Network*, vol. 36, no. 2, pp. 16–24, Mar. 2022.

[28] A. A. Atallah, G. B. Hamad, and O. A. Mohamed, "Routing and scheduling of time-triggered traffic in time-sensitive networks," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 7, pp. 4525–4534, Jul. 2020.

[29] J. Yan, W. Quan, X. Jiang, and Z. Sun, "Injection time planning: Making CQF practical in time-sensitive networking," in *Proc. IEEE INFOCOM,* Beijing, China, pp. 616–625, 2020.

[30] L. Zhao, P. Pop, Z. Zheng, H. Daigmorte, and M. Boyer, "Latency analysis of multiple classes of AVB traffic in TSN with standard credit behavior using network calculus," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 10, pp. 10291–10302, Oct. 2021.

[31] Y. Yuan, X. Cao, Z. Liu, C. Chen, and X. Guan, "Adaptive priority adjustment scheduling approach with response time analysis in time-sensitive networks," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 12, pp. 8714–8723, Dec. 2022.

[32] Y. Zhang, Q. Xu, X. Guan, C. Chen, and M. Li, "Wireless/wired integrated transmission for industrial cyber-physical systems: risk-sensitive co-design of 5G and TSN protocols," *Science China Information Sciences*, vol. 65, no. 1, pp. 1–16, Dec. 2021.

[33] Z. Cheng, D. Yang, W. Zhang, J. Ren, H. Wang, and H. Zhang, "DeepCQF: Making CQF scheduling more intelligent and practicable," in *Proc. IEEE ICC,* Seoul, Korea, pp. 1–6, 2022.

[34] A. A. Atallah, G. B. Hamad, and O. A. Mohamed, "Routing and scheduling of time-triggered traffic in time-sensitive networks," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 7, pp. 4525–4534, Jul. 2020.

[35] "IEEE standard for local and metropolitan area networks–bridges and bridged networks – amendment 31: Stream reservation protocol (SRP) enhancements and performance improvements," *IEEE Std 802.1Qcc-2018 (Amendment to IEEE Std 802.1Q-2018 as amended by IEEE Std 802.1Qcp-2018)*, pp. 1–208, 2018.

[36] H. Sami, H. Otrok, J. Bentahar, and A. Mourad, "AI-based resource provisioning of IoE services in 6G: A deep reinforcement learning approach," *IEEE Transactions on Network and Service Management*, vol. 18, no. 3, pp. 3527–3540, Sept. 2021.

[37] Q. Ye, W. Shi, K. Qu, H. He, W. Zhuang, and X. Shen, "Joint RAN slicing and computation offloading for autonomous vehicular networks: A learning-assisted hierarchical approach," *IEEE Open Journal of Vehicular Technology*, vol. 2, pp. 272–288, June 2021.

[38] S. Wang, Y. Guo, N. Zhang, P. Yang, A. Zhou, and X. Shen, "Delay-aware microservice coordination in mobile edge computing: A reinforcement learning approach," *IEEE Transactions on Mobile Computing*, vol. 20, no. 3, pp. 939–951, Mar. 2021.

[39] Y. Wang, M. Chen, T. Luo, W. Saad, D. Niyato, H. V. Poor, and S. Cui, "Performance optimization for semantic communications: An attention-based reinforcement learning approach," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 9, pp. 2598–2613, Sept. 2022.

[40] W. Shi, S. Zhou, Z. Niu, M. Jiang, and L. Geng, "Joint device scheduling and resource allocation for latency constrained wireless federated learning," *IEEE Transactions on Wireless Communications*, vol. 20, no. 1, pp. 453–467, Jan. 2021.

[41] W. Zhang, D. Yang, H. Wang, X. Huang, and M. Gidlund, "CarNet: A dual correlation method for health perception of rotating machinery," *IEEE Sensors Journal*, vol. 19, no. 16, pp. 7095–7106, Aug. 2019.

[42] R. Livni, S. Shalev, and O. Shamir, "On the computational efficiency of training neural networks," in *Proc. NeurIPS,* Montréal, Canada, vol. 27, pp. 855–863, 2014.

Dong Yang (Member, IEEE) received his B.S. degree from Central South University, Hunan, China, in 2003 and Ph.D. degree in Communications and Information Science from Beijing Jiaotong University, Beijing, China, 2009. From March 2009 to June 2010, he was a Post-Doctoral Research Associate with Jonkoping University, Jonkoping, Sweden. In August 2010, he joined the School of Electronic and Information Engineering, Beijing Jiaotong University. Since 2017, he is a Full Professor of Communication Engineering in Beijing Jiaotong University. His research interests include network architecture, wireless sensor networks, industrial network and Internet of Things.

Weiting Zhang (Member, IEEE) earned the Ph.D. degree in Communication and Information Systems with the Beijing Jiaotong University, Beijing, China, in 2021. From Nov. 2019 to Nov. 2020, he was a visiting Ph.D student with the Department of Electrical and Computer Engineering, University of Waterloo, Canada. Starting from Dec. 2021, he works as an associate professor with the School of Electronic and Information Engineering, Beijing Jiaotong University. His research interests include industrial Internet of Things, deterministic networks, edge intelligence, and machine learning for network optimization.

Qiang Ye (Senior Member, IEEE) Qiang Ye received the PhD degree in Electrical and Computer Engineering from the University of Waterloo, ON, Canada, in 2016. Since Sept. 2021, he has been an Assistant Professor with the Department of Computer Science, Memorial University of Newfoundland, NL, Canada. Before joining Memorial, he had been with the Department of Electrical and Computer Engineering and Technology, Minnesota State University, USA, as an Assistant Professor from Sept. 2019 to Aug. 2021 and with the Department of Electrical and Computer Engineering, University of Waterloo as a Postdoctoral Fellow and a Research Associate from Dec. 2016 to Sept. 2019. He has published over 50 research articles on top-ranked IEEE Journals and Conference Proceedings. He is/was General and TPC co-chairs for different international conferences and workshops, e.g., IEEE VTC'22, IEEE INFOCOM'22, and IEEE IPCCC'21. He serves/served as associate editors of IEEE Transactions on Cognitive Communications and Networking, IEEE Open Journal of the Communications Society, Peer-to-Peer Networking and Applications, ACM/Wireless Networks, and International Journal of Distributed Sensor Networks. He is a Senior Member of IEEE.

Chuan Zhang (Member, IEEE) Chuan Zhang received his Ph.D. degree in computer science from Beijing Institute of Technology, Beijing, China, in 2021. From Sept. 2019 to Sept. 2020, he worked as a visiting Ph.D. student with the BBCR Group, Department of Electrical and Computer Engineering, University of Waterloo, Canada. He is currently an assistant professor at School of Cyberspace Science and Technology, Beijing Institute of Technology. His research interests include secure data services in cloud computing, applied cryptography, machine learning, and blockchain.

Ning Zhang (Senior Member, IEEE) is an Associate Professor in the Department of Electrical and Computer Engineering at University of Windsor, Canada. He received the Ph.D degree in Electrical and Computer Engineering from University of Waterloo, Canada, in 2015. After that, he was a postdoc research fellow at University of Waterloo and University of Toronto, Canada, respectively. His research interests include connected vehicles, mobile edge computing, wireless networking, and machine learning. He is a Highly Cited Researcher. He serves as an Associate Editor of IEEE Internet of Things Journal, IEEE Transactions on Cognitive Communications and Networking, and IEEE Systems Journal; and a Guest Editor of several international journals, such as IEEE Wireless Communications, IEEE Transactions on Industrial Informatics, IEEE Transactions on Intelligent Transportation Systems, and IEEE Transactions on Cognitive Communications and Networking. He also serves/served as a general chair for IEEE SAGC 2021, TPC chair for IEEE VTC 2021 and IEEE SAGC 2020, a track chair for several international conferences including IEEE ICC 2022, CollaborateCom 2021, IEEE VTC 2020, AICON 2020 and CollaborateCom 2020, and a co-chair for numerous international workshops. He received an NSERC PDF award in 2015 and 6 Best Paper Awards from IEEE Globecom in 2014, IEEE WCSP in 2015, IEEE ICC in 2019, IEEE ICCC in 2019, IEEE Technical Committee on Transmission Access and Optical Systems in 2019, and Journal of Communications and Information Networks in 2018, respectively.

Chuan Huang (Member, IEEE) received the Ph.D. degree in electrical engineering from Texas A&M University, College Station, USA, in 2012. From August 2012 to July 2014, he was a Research Associate and then a Research Assistant Professor with Princeton University and Arizona State University, Tempe, respectively. He is currently an Associate Professor with the Chinese University of Hong Kong, Shenzhen. His current research interests include wireless communications and signal processing. He served as a Symposium Chair for IEEE GLOBECOM 2019 and IEEE ICCC 2019 and 2020. He has been serving as an Editor for IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, IEEE ACCESS, Journal of Communications and Information Networks, and IEEE WIRELESS COMMUNICATIONS LETTERS.

Hongke Zhang (Fellow, IEEE) received the M.S. and Ph.D. degrees in Electrical and Communication Systems from the University of Electronic Science and Technology of China, Chengdu, China, in 1988 and 1992, respectively. From 1992 to 1994, he was a Postdoctoral Researcher with Beijing Jiaotong University, Beijing, China, where he is currently a Professor with the School of Electronic and Information Engineering and the Director of the National Engineering Lab on Next Generation Internet Technologies. His research has resulted in many papers, books, patents, systems, and equipment in the areas of communications and computer networks. He is the author of more than ten books and the holder of more than 70 patents. Dr. Zhang is the Chief Scientist of the National Basic Research Program of China (973 Program) and has also served on the editorial boards of several international journals.

Xuemin (Sherman) Shen (Fellow, IEEE) received the Ph.D. degree in electrical engineering from Rutgers University, New Brunswick, NJ, USA, in 1990. He is a University Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Canada. His research focuses on network resource management, wireless network security, Internet of Things, 5G and beyond, and vehicular ad hoc and sensor networks. Dr. Shen is a registered Professional Engineer of Ontario, Canada, an Engineering Institute of Canada Fellow, a Canadian Academy of Engineering Fellow, a Royal Society of Canada Fellow, a Chinese Academy of Engineering Foreign Member, and a Distinguished Lecturer of the IEEE Vehicular Technology Society and Communications Society.

Dr. Shen received the R.A. Fessenden Award in 2019 from IEEE, Canada, Award of Merit from the Federation of Chinese Canadian Professionals (Ontario) in 2019, James Evans Avant Garde Award in 2018 from the IEEE Vehicular Technology Society, Joseph LoCicero Award in 2015 and Education Award in 2017 from the IEEE Communications Society, and Technical Recognition Award from Wireless Communications Technical Committee (2019) and AHSN Technical Committee (2013). He has also received the Excellent Graduate Supervision Award in 2006 from the University of Waterloo and the Premier's Research Excellence Award (PREA) in 2003 from the Province of Ontario, Canada. He served as the Technical Program Committee Chair/Co-Chair for IEEE Globecom'16, IEEE Infocom'14, IEEE VTC'10 Fall, IEEE Globecom'07, and the Chair for the IEEE Communications Society Technical Committee on Wireless Communications. Dr. Shen is the President Elect of the IEEE Communications Society. He was the Vice President for Technical & Educational Activities, Vice President for Publications, Member-at-Large on the Board of Governors, Chair of the Distinguished Lecturer Selection Committee, Member of IEEE Fellow Selection Committee of the ComSoc. Dr. Shen served as the Editor-in-Chief of the IEEE IoT JOURNAL, IEEE Network, and IET Communications.