C-FDRL: Context-Aware Privacy-Preserving Offloading Through Federated Deep Reinforcement Learning in Cloud-Enabled IoT

Yang Xu, *Member, IEEE*, Md Zakirul Alam Bhuiyan, *Senior Member, IEEE*, Tian Wang, Xiaokang Zhou, *Member, IEEE*, and Amit Kumar Singh, *Senior Member, IEEE*

***Abstract*—Recently, artificial intelligence approaches are widely suggested to optimize numerous offloading task-scheduling purposes. However, they confront difficulties in maintaining data privacy regarding the context of the data offloading during the course of offloading in the different stages. To address this problem, in this article we propose C-fDRL, a framework to provide context-aware federated deep reinforcement learning (fDRL) to maintain the context-aware privacy of the task offloading. We perform this in three stages (CloudAI, EdgeAI, and DeviceAI) of the overall system. C-fDRL checks whether the privacy of high-context-aware data with the task being offloaded is maintained locally at the DeviceAI, and low-context-aware data distributedly at the EdgeAI. When there is an offloading task request or a user needs to offload the data, C-fDRL uses a context-aware data management approach to decouple the context-aware (privacy) data from the tasks. This separates the context-aware data from the task for local computation and allows a new scheduling technique called "context-aware multilevel scheduler." This places high-context-aware data on local devices and low-context-aware data at the edge device for computation before the actual task execution. We performed experiments to evaluate the data privacy with the offloading tasks and the federated DRL. The results show that the proposed C-fDRL performs better than the existing framework.**

***Index Terms*—Cloud computing, context-aware data privacy, deep reinforcement learning (DRL), edge computing (EC), federated learning (FL), Internet of Things (IoT), security.**

I. Introduction

With the emergence of cloud-enabled Internet of Things (IoT) technologies, the more IoT devices are attached, the more data are produced, and the more communication overhead is incurred. Industrial IoT (IIoT) and smart city IoT devices contribute to the highest amount of data production. Many investigations show that data produced by IoT devices increases nearly 50 times in just seven years, raising some enormous challenges to the industries tasked with being stewards of this data. Edge computing (EC), the promising philosophy that emerges to improve these situations, where resource optimization is advanced from the cloud environment to the EC unit (ECU) servers [1], [2]. In the EC viewpoint, the IIoT devices at the edge are the data generator and the service contributor. With the help of EC services, a wide range of IIoT devices deeply achieve the demanded resources in real-time by migrating their computation-exhaustive jobs to ECU for execution, so-called computation offloading [3].

Recently, artificial intelligence (AI) approaches have been widely suggested to optimize offloading task scheduling in a real-edge-assisted IoT environment [4]–[11]. Since the ECU servers have features of resource limitations and dynamic changes, AI approaches, such as deep neural network (DNN), deep learning (DL), and deep reinforcement learning (DRL), are significantly utilized to design a suitable framework for data offloading and computational tasks scheduling in the edge. The DNN as the DL is suitable for handling IIoT processing tasks and scheduling since DNN learns features freely from the acquired data. The techniques of DNN have been employed in numerous angles of IIoT, such as resource monitoring, bandwidth monitoring and management, industrial manufacturing, etc. Recently, it is regularly seen that DNN has been employed in the edge-distributed units so as to solve computationally tricky tasks in the IIoT.

However, AI approaches confront numerous difficulties when applied in IoT, edge, and cloud computing environments. Most existing AI approaches substantially depend on Big Data for

Manuscript received 30 July 2021; revised 21 October 2021 and 24 December 2021; accepted 18 January 2022. Date of publication 8 February 2022; date of current version 13 December 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 62002113 and in part by the Natural Science Foundation of Hunan Province under Grant 2021JJ40122. Paper no. TII-21-3301. *(Corresponding author: Md Zakirul Alam Bhuiyan.)*

Yang Xu is with the College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China (e-mail: xuyangcs@hnu.edu.cn).

Md Zakirul Alam Bhuiyan is with the Department of Computer and Information Sciences, Fordham University, NY 10458 USA (e-mail: m.bhuiyan.dr@ieee.org).

Tian Wang is with the BNU-UIC Institute of Artificial Intelligence and Future Networks, Beijing Normal University (BNU Zhuhai), Guangdong 519088, China, and also with the Guangdong Key Lab of AI and Multi-Modal Data Processing, BNU-HKBU United International College (UIC), Guangdong 361021, China (e-mail: cs_tianwang@163.com).

Xiaokang Zhou is with the Faculty of Data Science, Shiga University, Shiga 522-8522, Japan (e-mail: zhou@biwako.shiga-u.ac.jp).

Amit Kumar Singh is with the National Institute of Technology Patna, Department of Computer Science and Engineering, National Institute of Technology Patna, Patna 800005, India (e-mail: amit.singh@nitp.ac.in).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TII.2022.3149335.

Digital Object Identifier 10.1109/TII.2022.3149335

1551-3203 © 2022 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: McMaster University. Downloaded on August 10,2024 at 01:43:20 UTC from IEEE Xplore. Restrictions apply.

centralized or global training. In some critical IoT applications, such as healthcare, such data are typically gathered and distributed by diverse healthcare organizations, collaborative shareholders, healthcare providers, and companies. The associations among the providers typically engage data exchange, data usage, and keeping copies of data locally, while the data may comprise personal privacy. As a result, potential privacy leakage issues appear. Healthcare organizations or providers are requested to evade data exchange by keeping the data locally to maintain personal privacy, making it challenging for them to train a model collaboratively.

Recently, there has been a large boost in the research of federated learning (FL), which seems promising in privacy-preserving and secure computation and addressing the data security issue of DL [4], [12]. With the FL approaches, the abovementioned collaborative problems can immensely be fixed [13]. FL functions as a learning approach for numerous data providers, allowing providers to build an effective model while preserving the data locally. Comprehensive and successful cases have demonstrated that FL can tradeoff between model performance and privacy [14], [15]. While FL approaches have been productively applied to numerous DL-based approaches, our investigation finds a few situations involving DNN-based approaches.

Furthermore, FL broadens threats to context-awareness, leaving insecurity issues in the learning process at the same point of learning. Here, the disclosure of context-aware data, including personal, industrial, and security-sensitive information, location, important business process, operation, etc., are the huge calamity, especially in IoT environments and operations. During the course of the transmission process between the IIoT devices, a range of unforeseen leaks of context-aware data, e.g., user location, location-oriented information (such as pictures, landscape, monument, and contact number), and private picture, leads the computation offloading more complex. With the FL, maintaining data privacy, especially the context of the data in the different stages when offloading from the IoT devices, confronts difficulty [16], [17]. Since the data produced by the IoT devices become more and more complex, this requires to withstanding context-aware data disclosure and breaches unavoidably. Reactive data providers may monitor transitional model conditions and provide random data, which is the piece of decentralized FL model training. For instance, attackers pretending as authentic consumers may send wrong parameters updates to compromise the trained model, reconstruct the information of authentic customers, or get authorization to model, even without making any valuable contribution. Existing FL approaches cover that attackers can control a large set of consumers in FL employment that may deliver poisoning attacks. Furthermore, in FL, machine learning (ML)/DL models are trained centrally on IoT devices. Traditional privacy-preserving methods used in FL are mainly based on the user-level differential privacy. However, such a method is defenseless to poisoning attacks, including Sybil attacks [18].

Considering the data offloading situations, resource usage of ECU servers can be somewhat inadequate, and the implementation is unsuccessful to some extent. There can be severe data privacy leakage, including id, system behavior, location, media data, etc. The course of the communication process from IoT health devices to ECU servers severely controls the usefulness of ECUs in IoT. Current FL approaches can just protect the privacy of data features in some respects. However, they do not handle many context-aware information. Guaranteeing the security and confidentiality of context-aware information should be significant. Furthermore, current work FL approaches can protect the privacy of data features in some respects. However, they do not handle many context-aware information.

To address this problem, we propose `C-fDRL`, a framework to provide context-aware federated DRL (fDRL) to maintain the context-aware privacy of the task offloading. We perform this in three stages (CloudAI, EdgeAI, and DeviceAI) of the overall system. `C-fDRL` checks whether the privacy of high-context-aware data with the task being offloaded is maintained locally at the DeviceAI, and low-context-aware data distributedly at the EdgeAI. When there is an offloading task request or a user needs to offload the data, `C-fDRL` uses a context-aware data management approach (CDMA) to decouple the context-aware (privacy) data from the tasks. This separates the context-aware data from the task for local computation and allows a new scheduling technique called "context-aware multilevel scheduler" that places high-context-aware data on local devices and low-context-aware data at the edge device for computation before the actual task execution. We performed experiments to evaluate the data privacy with the offloading tasks and the fDRL scheduling policy. The results show that the proposed `C-fDRL` has done better than the existing framework.

The major contributions of this article are as follows.

1) We propose `C-fDRL`, a novel context-aware task offloading in cloud-edge-assisted IoT. We provide the detailed design and architecture of the `C-fDRL`.
2) We define context-aware data and propose an fDRL and its detailed operation for context-aware data offloading.
3) We present a two-phase local DRL process, one is associated with the DeviceAI and the other one is with the EdgeAI. We present a CDMA for the local DRL process in the IIoT devices,
4) We have implemented the `C-fDRL` and conducted a comprehensive experiment. The result shows that the `C-fDRL` has a great potential to protect data privacy while learning.

The rest of this article is organized as follows. The design of the `C-fDRL` is provided in Section II. Section III presents the context-aware task offloading problem in cloud-edge-assisted IoT and reviews related work. Section IV proposes the fDRL for context-aware data offloading. Section V presents a two-phase local DRL process. We carry out the performance evaluation via simulations in Section VI. Finally, Section VII concludes this article.

II. Design of the `C-fDRL` Framework

In this section, we provide a quick overview of `C-fDRL`, a framework to provide fDRL to maintain the context-aware task offloading, which is illustrated in Fig. 1. Our target application

Authorized licensed use limited to: McMaster University. Downloaded on August 10,2024 at 01:43:20 UTC from IEEE Xplore. Restrictions apply.

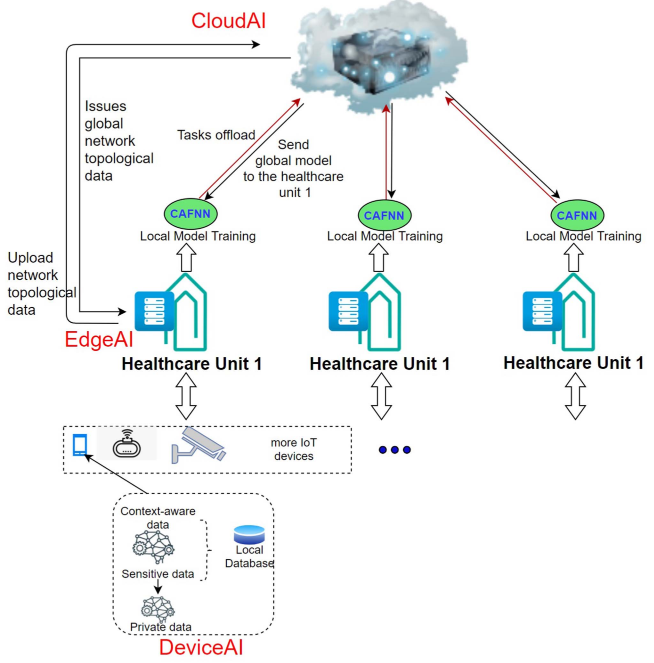

Fig. 1. C-fDRL framework.

is healthcare. However, this work can have a wide range of applications. We assume that a cloud-edge-enabled IIoT network is considered that contains a collection of IoT sensor devices, such as healthcare monitoring devices, which are distributed to cover different organizations, areas, or units. Each of them will process the complex offloading tasks for consumers/users. IIoT devices are resources constrained. To efficiently deliver the services for the applications with the quality and real-timelessness, the ECU server-based computation and communication together with limited computing power are employed and dispersed.

As shown in Fig. 1, task-offloading architecture for the cloud-EC enabled IIoT is illustrated. The computation and monitoring operations in the architecture are performed in stages, i.e., the IoT stage, the edge stage, and the cloud stage. We then define the three stages as DeviceAI, EdgeAI, and DeviceAI, respectively. Once the global cloud disperses the offloading tasks down to the networks, the ECU server disperses the tasks among the IIoT devices. Our primary interest is in the task at the IoT device stage, C-fDRL checks whether the privacy of high-context-aware data with the task being offloaded is maintained locally at the DeviceAI and low-context-aware data distributedly at the EdgeAI. To achieve this, the DeviceAI at each IIoT device is set to classify and preserve the most highly sensitive and context data locally. A local context-aware data classifier is devised, which permits the local models to take advantage of highly sensitive data and topological information to have quick classification. When there is an offloading task request or a user needs to offload the data, C-fDRL decouples the context-aware and sensitive (privacy) data from the tasks. This separates the context-aware data from the task for local computation and allows a new scheduling technique called "context-aware multilevel scheduler" that places high-context and privacy data on local devices for EdgeAI and low-context-aware data at the distributed devices for EdgeAI for computation before the actual task execution.

After separating the context-aware data from the task request, DeviceAI produces low-sensitive context-aware data. The IIoT device then forwards the data to the ECU for computation. ECU enables EdgeAI covering a particular healthcare unit that preserve the (low) context-aware data according to their context and policy and restrict context-aware data offloading towards the CloudAI. A distributed context-aware data classifier is devised following the local one, allowing the local models to take advantage of sensitive classified data and topological information to guarantee data offloading without context and privacy issues.

III. Context-Aware Task Offloading in Cloud-Edge-Assisted IoT

We introduce the design of the C-fDRL, the proposed privacy-protected task-offloading framework. First, we provide data-offloading architecture, task-offloading model, context-aware data model, fDRL model for context-aware data offloading model, and overview of the system.

A. Data-Offloading Architecture

With the emergence of cloud-enabled IIoT and edge-enabled IIoT, there is a significant interest in applying these technologies in industrial informatics, since they are promised to handle a big amount of data, improve communication challenges, provide real-time industrial monitoring, and improve the quality of service (QoS) [16], [19]–[22]. These are mainly associated with IIoT devices and emerging applications, such as healthcare, smart grid, and smart city applications. Particularly, in the cloud-enabled IoT healthcare applications, such as remote patient-monitoring devices, EKG, ECG, and pressure, depression, and mood monitoring devices. Various ingestible sensors are used for the purpose, which are mainly tiny and resources constrained. They can neither process all the data they collect nor reliably transmit the collected data using their limited bandwidth in real-time. This finally raises QoS, data security, and privacy issues. To reduce the computation-intensive tasks in intelligent healthcare, data are usually offloaded to the cloud through the ECU server and ECU servers are used to reduce the burden on the IoT devices.

We propose C-fDRL to address the data privacy context of cloud-EC and IoT system. This is comprised of multiple ECUs in different areas, each of which will process the complex tasks for users. To efficiently deliver services for the IoT system and applications with the quality and real-timelessness, the ECU server-based computation and communication with limited IIoT computing power are employed and dispersed to a healthcare application.

Fig. 1 illustrates the task system offloading architecture for cloud-EC-enabled IIoT. The computation and monitoring operations in the architecture are performed in levels or stages, i.e., the IoT device stage, the edge stage, and the cloud stage. A cloud-edge-enabled IIoT network contains a collection of IIoT sensor devices denoted by $S = \{s_1, s_2, \ldots, s_N\}$, where N is the given amount of IIoT devices in S. The devices collect data from

Authorized licensed use limited to: McMaster University. Downloaded on August 10,2024 at 01:43:20 UTC from IEEE Xplore. Restrictions apply.

the environment. We assume that each IIoT device yields one task, and the IIoT tasks are denoted by $T = \{t_1, t_2, \ldots, t_M\}$ and they forward the data offload toward an EdgeAI for computation. The number of tasks processed in a certain period, called time slots.

B. Context Awareness in the Task Offloading

While improving the QoS in IIoT applications, such as healthcare, IIoT devices (with mobility) produce privacy data encompassing private patient data where unauthorized processing or access to the data establishes the violation of data privacy. This can lead to disastrous consequences for a patient, hospital staff, and doctor, and financial loss to the industrial organization via rules and regulations, such as GDPR. Context-aware privacy protection demands consciousness of sensitive applications (medical databases), patients/users, identities, disease information, disease data classifications, and event types. The process of distinguishing every possible kind of health data, access, privileged and unauthorized user identities, healthcare applications, and events is a severe foremost step in executing context-aware privacy measures. Context-aware privacy protection is the use of supplemental information or technique to improve data security or privacy decisions at the time the context is interfered with or attempted to get accessed. This results in more accurate data security decisions capable of supporting data privacy in healthcare, as well as dynamic business and IT environments.

C. DRL Process for Task Offloading

AI approaches, including DL and DRL-based approaches, have been suggested and implemented for task scheduling, process scheduling, and code-offloading applications. They look to be efficient as compared to traditional approaches. However, the data privacy issue of DL and DRL is challenging since they have a significant impact on the overall system performance. There are diverse security and privacy attacks that surface with these approaches. Recently, edge AI (called EdgeAI) has attracted significant attention in the AI communities. Different AI approaches, including ML and DRL, have been seen used in IIoT systems and applications. DRL has received significant attention in some applications, including offloading and resource scheduling. Under the CloudAI stage, the EdgeAI can manage and process a significant amount of data, offloaded by the IIoT devices in real-time and produce real-time decisions, final data, semiprocessed data, raw data, or the final result into the CloudAI. The DRL approach is set to optimize the task offloading without any experience of the IoT networking statuses and enhance the long-term reward. However, when applying the DRL approach, the training agents in all the three stages (CloudAI, EdgeAI, and DeviceAI) are required to reveal their raw datasets or some sense of data. Particularly, data streaming from the IIoT devices towards the edge devices is often required to share at the EdgeAI level and CloudAI stage.

In `C-fDRL`, the process for training a DRL model at the IIoT devices engages streaming data to a server at the EdgeAI level and uses the data to train models. The method of training functions is just great as long as the context-aware privacy of the data is not a concern. However, when training the DRL model where individually recognizable data are engaged in the device with especially sensitive data, such as in healthcare, DRL may be inappropriate. Training the DRL model on a CloudAI (centralized server) also implies that we require a significant amount of data transmission and storage space and world-class security techniques. Furthermore, both EdgeAI and CloudAI can be connected to multiple domains. They can be controlled and managed by different controllers or healthcare units or organizations, as shown in Fig. 1, such that there can be the situation that sensitive data can be easily leaked due to be the suspicious controllers, agents, as well as cyber attackers.

IV. Federated DRL for the Context-Aware Data Offloading

A. fDRL Approach

The abovementioned problem can be overcome to some extent if it is possible to train DRL models with context-aware data, which is locally stored on a user's device and have the data processed locally and separately at each device. FL functions as a learning approach that can be used in individual IoT devices, letting each of the devices establish an efficient collaborative sharing model while possessing their contextual data at the device stage. Recent examples reveal that FL can do a tradeoff between the performance of the FL training model and the privacy [15].

However, some controllers or agents internally may not be willing to reveal their dataset collected from the IIoT devices, especially the healthcare or commercial data, which are very sensitive. While the FL approach maintains data privacy to some extent when not sharing the data, the protection is unclear when there are sophisticated cyberattacks and attackers attempt to infer the collected data from both the shared data, including membership inference attack [23]. In Fig. 1, the proposed `C-fDRL` framework combines the DRL model with FL making a new fDRL model, which we found to be reasonable to provide context-aware data privacy that should improve the situation to some respects. Fig. 1 describes the details of the fDRL workflow in `C-fDRL`.

In traditional DRL model-based cloud-edge computing, a cloud-based global controller delivers a unified interface. IIoT system users can request EdgeAI services for computation by accessing the interface, associating the ECU servers and the system users. As shown in Fig. 1, we devise the fDRL to optimize the context-awareness in `C-fDRL`, which will consider the network topological information, conditions, and task details. Also, this will optimize the task-scheduling procedure to minimize the maximum execution time of a task, therefore, leading to achieving reasonable computation latency. As shown in Fig. 1, the global controller will combine the distributed DRL process and diffuse the centralized DRL model into the EdgeAI agents in a distributed manner. The EdgeAI proceeds to synchronize and update the local model with the DeviceAI at each of the IIoT devices, and the processes continue until the global model at the CloudAI converges.

Authorized licensed use limited to: McMaster University. Downloaded on August 10,2024 at 01:43:20 UTC from IEEE Xplore. Restrictions apply.

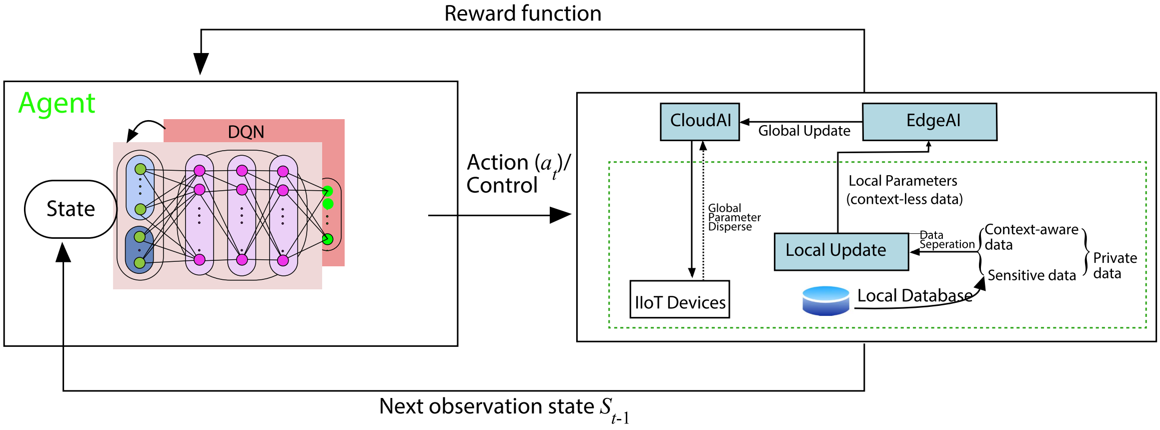

Fig. 2. Overall workflow of the proposed C-fDRL framework.

B. Employing Q-Network Algorithm

The deep Q-network algorithm is utilized to study the task-offloading strategy on the basis of environments and experiences, according to Q-network presented in [22], [24], and [25]. It is used to look for the best action-selection policy through Q-function. The aim here is to enhance the value function Q (simply Q-value). Q-table aids to determine the best action for each state of the fDRL. As shown in Fig. 2, the state transition can make the interaction between DRL and FL. The earlier state transition can be forming the Q-value, and then forecast the subsequent actions, and an EdgeAI function as a single DQN agent. In every round, the DQN agent gets the instant reward based on the ongoing state and relevant action, and the steady-state can then charge into the next state. To maintain the privacy of the offloading tasks in the local stage, and check whether the privacy with data being offloaded is maintained. When there is an offloading task request, or a user wants to offload the data, it decouples the tasks' context-aware (privacy) data. As shown in Fig. 2, this separates the context-aware data from the task for local deep Q-network. The fDRL training procedure is discussed as follows.

1) The data offloading task can be triggered by either the cloudAI or local device layer.
2) For the cloudAI, it retains a global deep Q-network model that covers the distributed agents and breaks down the tasks into the agent. A C-fDRL framework arbitrarily picks the distributed training agents in every round.
3) Every selected agent associated with the ECU server starts training a distributed deep Q-network model using the local dataset. The training is done fully in a distributed manner so that the neighboring agent is now allowed to learn the privacy context of this ECU server.
4) Every local device or a subset of local devices is tuned to start training. Proposed C-fDRL identifies the context-awareness of the task offloading, that is to say, whether the training and processing for the data will involve any data privacy context and privacy computation. If the offloading task involves context-awareness, tasks will be divided into two-phase training at the local layer. A few IIoT devices in the local layer get the data offloading task or a device itself needs to start data offloading. The one phase is about the basic training at the local IIoT devices. Another phase is about context-aware data.
5) The CloudAI gets the local deep Q-network models, which should be with context awareness free. It then aggregates them into the global deep Q-network model, and finally diffuses the global model to the selected agents [25].
6) The training and processes, free of context-awareness execution, continue until the global model converges.

Through the abovementioned fDRL, any context-related data privacy or user-restricted data can be preserved using the two-phase training, local training does not involve context-related data directly, and distributed agents do not reveal the context about the data. In the meantime, training or learning in a distributed and asynchronous manner will speed up the training rate. This is important to advance the real-time data classification locally regarding context-awareness. As it is found that the training in distributed and asynchronous can overcome the limitation of the optimization opportunities that we can get in the distributed model training.

C. Learning Process With Context Awareness

We consider maintaining data privacy from the top to the bottom of the IIoT network. We use three-level privacy protection with the fDRL approach. The first level is cloud stage privacy protection at the CloudAI. We apply the FL approach to train a centralized deep Q-network model to provide privacy protection at this level. This model is diffused across all the agents associated with the ECU servers (second-level privacy protection) at the EdgeAI, and then is further diffused across all of the local agents (third-level privacy protection) at the device level. Each level updates the model based on the dataset collected locally. Then, the first level at the CloudAI learning model agent gathers and accumulates the second-level (distributed) learning models, updates the first-level learning model, and finally diffuses it to all the downstream agents again. The abovementioned process can repeat while waiting for its convergences. That is to

Authorized licensed use limited to: McMaster University. Downloaded on August 10,2024 at 01:43:20 UTC from IEEE Xplore. Restrictions apply.

achieve expected accumulated rewards or reach an acceptable earned reward in different learning rates. Generally, to achieve an accumulated reward through the federated deep Q-network model and centralized Q-network model in an epoch, centralized deep Q-network is supposed to outperform the federated deep Q-network but may converge faster. This can be due to the lack of the centralized deep Q-network setting information, as federated Q-network relies on the local training process and dataset. The accumulated reward of federated deep Q-network may go above the centralized deep Q-network as training time surges. That is to say, a federated deep Q-network may get a near-optimal performance when training times increase and become sufficient. Allowing for privacy protection benefits, a near-optimal performance of the federated deep Q-network should be reasonable.

We use the local devices' collected datasets D and d to denote the local dataset at the agent on the ECU server p_i. We have the loss of function $F_i(\theta_i)$ of the agent pi. Subsequently, the global loss function of p_i based on the local dataset D_i. Here, it is worth noting that the third-stage training problem is to lessen the loss function for every dataset, and get associated parameters for the local DQN in order to provide privacy protection. Therefore, the global loss function can accumulate the local dataset and local data privacy protection. We think that the FL process will continue for training times, and we call it the local parameters of p_i in the iteration. When second-stage agents get all of its acquired local parameters and upload their local parameters, the third stage can accumulate the corresponding parameters. We apply the federated averaging method for third-stage parameter updates in this article. Here, the third-stage parameters equal the weighted sum of the local parameters obtained through the distributed agents. Repeatedly, an agent can download the global parameters from the controller. This applies to the local training process in the (t + 1)th iteration, as long as the global model does not converge.

The idea of the fDRL process is presented in Fig. 2. The input is the total federated training times, including all the three-stage learning process time, and the output is the third-stage DRL model. In the beginning, the third-stage agent initiates diffusion of the DRL model. As for the distributed agents, the agents grain the dispersed third-stage (cloud) model from the global agent and execute the algorithm to train the local deep Q-network model by using IIoT devices' collected dataset locally and asynchronously. On the other hand, the third-stage (cloud) agent retrieves the distributed deep Q-network models and updates the global model by federated accumulation. Finally, the third-stage models will be diffused further in the subsequent execution. The fDRL process can carry on for a given time, such that the centralized version converges.

V. Two-Phase Local DRL Process

A. Phase 1: Local DRL Process at the EdgeAI

In the local training at the device layer, we apply the DRL training method through the deep Q-network approach. Given a state s_t at time t, the `C-fDRL` execute action v and calculate the instantaneous reward $r(s_{t+1}, v_i)$, and the present state s_t is switched to s_{t+1}. Once we have all the elements of the DRL process, we emphasize the core objective, which is to get a policy ϕ that optimize the rewards, where $\phi = v_t$. This can be given as

$$R_t = \sum\nolimits_{t=0} \beta^t r_t \tag{1}$$

where β is a reduction factor $0 \leq \beta \leq 1$. By means of the Q-Network approach, a DRL is necessitated to estimate the optimum action-value function (Q) through a deep convolutional neural network (CNN) [26], which can be given by

$$Q * (\lambda, v) = \max_r E[e_t + \beta e_{t+1} + \beta^2 e_{t+1} + \cdots |\lambda_t = \lambda, v_t = v, r]. \tag{2}$$

The function action-value discerns the highest total of rewards r_t, which is abated by β at every time-step, made possible by the rule or policy $r = \text{Prob}(v|\lambda)$ after a given status (state) (λ) and action (v).

The CNN with weights (ω) is exploited to approximate the optimum function action-value, such that $Q(\lambda, v; \omega) \approx Q^*(\lambda, v)$. When the CNN is trained with the minimization of loss functions $F_i(\omega_i)$, which is updated at every time-step. We conduct N perception replay, in order to track the agent's perceptions $p_t = (\lambda_t, a_t, r_t, \lambda_{t+1})$ at every time-step t in a replay dataset $D_t = p_1, \ldots, p_t$. The dataset currently "perceived" transitions could be with the perception replay mechanism that are important for the incorporation of the RL and deep CNN [27]. The Q-network learning produce up-to-date information to apply on data samples of the training data $(\lambda, v, r, \lambda')$, which are taken drawn frequently from the perception replay storage S. The update of the Q-network learning in i can be calculated by the loss function, given as

$$F_i(\theta_i) = \mathrm{E}_{(\lambda,v,r,\lambda')\sim U(S)} \left[(\eta_i - Q(\lambda, v; \theta_{i-1}))^2 \right]$$

$$\eta_i = e + \beta \max_{v'} Q(\lambda,' v'; \theta_{i-1}) \tag{3}$$

where θ denotes the IoT network generative parameters in i, which utilized to calculate the target state η_i, i.e., $\eta = r + \beta \max_{v'} Q(\lambda,' v'; \theta_{i-1})$. The gradient of the loss function can be calculated in accordance with ω_i. The gradient descent values can be calculated and executed on $\Delta = [(\eta_i - Q(\lambda, v; \theta_{i-1}))^2]$.

B. Phase 2: Local DRL Process at the DeviceAI Through the CDMA

In this section, we attempt to decouple the computation and task-offloading scheduling regarding the healthcare data sensitivity context.

We briefly present a CDMA for the IIoT devices, particularly, mobile devices, which is aligned with the features of the task computation at the EdgeAI stage. The CDMA first shortly checks whether the requested or demanded task offloading concerns any context, such as a patient, person, or hospital personal information, such as ID, medical information, locations, etc. Based on the data engaged in task offloading, the CDMA decouples data and task-offloading scheduling and conducts a resource analysis before task offloading. First, before the actual runtime of task offloading, the CDMA utilizes a context-aware

Authorized licensed use limited to: McMaster University. Downloaded on August 10,2024 at 01:43:20 UTC from IEEE Xplore. Restrictions apply.

data classifier to perform a quick analysis on the tasks to be offloaded to see if the task offloading may engage and process any context-aware privacy data. Second, if the task offloading has no concern about the context-awareness, the CDMA performs the task schedule that allocates tasks on the devices. Third, if the task offloading involves context-awareness, the CDMA decouples offloading task engagements associated with some contextual features (that usually come from internal apps, tools, and databases that application contextual information, such as patient and hospital staff privacy context). Instead, the CDMA uses the "data placement stage" that places data on the source device considering the context. The CDMA leads to run decouple execution of the data locally, get the high-stage output of the data, and incorporate output with task offloading. Fourth, the CDMA uses a "runtime adaptation stage" to observe the quality stage of task offloading or local execution and adapts "data placement" in a control loop if needed. As a result, the CDMA may optimize the tradeoff between the execution latency and data privacy level based on the context.

VI. Evaluations

A. Evaluation Framework and Datasets

To justify the performance of the C-fDRL in the cloud-edge-enabled IoT environment, we have carried out evaluations using the Keras with Tensorflow as the backend. All evaluations are conducted on a 64 bit Surface Book 2 with core i7-8650 U, 4.2 GHz 16 GB RAM, 6 GB graphic memory. We utilize RMSProp as the training optimizer [28].

We utilize two sets of datasets for evaluations. The first dataset for evaluation comes from a real observation study. We have had five smartphones and given them to five lab users, who voluntarily used them for a period of one month. The objective is to gather sensing data, including accelerators, three-axis gyroscope readings, location sensors, and barometers from intelligent IoT devices under different conditions. The second dataset is the synthetic data that includes three types of wearable sensing values. These are a thermometer, pulse oximeter, and blood pressure monitor. The intention here is to analyze the collected data to characterize the sensing things or objects embedded in different sensor devices from a security and privacy point of view. Through this research and data may not benefit anyone personally at this stage, it will assist us to realize to what degree healthcare data can be vulnerable during AI learning. We utilize Z-score to normalize both kinds of collected data before inputting them into the learning models. Supervised learning is used to construct the training, validation, and testing sets, each containing 60%, 20%, and 20% of all the collected data, respectively.

B. Comparisons

To fully assess the performance of the C-fDRL, we conducted following three thorough case studies using the gathered datasets.

1) We investigated the privacy concern under no local model (no-fDRL).

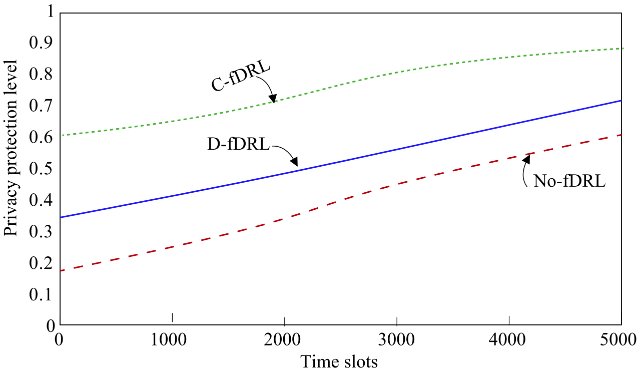

Fig. 3. Performance of the privacy-protection level in different frameworks.

2) We investigated the privacy protection when C-fDRL is fully active (C-fDRL).
3) We investigated the privacy concern under no local fDRL, but distributed version of it is in active (D-fDRL).

We have conducted a comparison with each other. Further, we have also compared our results with the results of four very recent works, including FASTGNN [29], ADDetector [4], FedGRU [30], and SpatioTemporal graph convolutional network (STGCN) [31].

C. Evaluation Results

The performance of data privacy while task offloading in C-fDRL is a mean of 250 time slots in the given task-offloading time, as shown in Fig. 3,

In the evaluation, local IoT devices process given data, classify the context-sensitive data, and process them locally. Context-less data are forwarded to the ECU servers for processing in a distributed manner. The experiment continues for 250 time slots. The privacy protection stage is calculated by the ratio of the total amount of data separately processed at the local IoT devices based on the context-sensitivity to the amount of data offloading toward the ECU server and cloud for computation. In an investigation, we have seen that there is a relationship between the collection data size (such as 20, 50, and 100 kb) and the privacy protection level. It is also seen that the privacy protection level increases as the amount of data collected by the healthcare IoT device increases. Furthermore, the privacy protection level increases as the number of healthcare IoT devices increases. That is to say, there is a different relationship between the privacy protection level and the amount of data collection together with the number of IoT devices involved in the data collection. In a study of the evaluation results, we have found that privacy concerns and costs increase as the amount of data collection increases. For example, in the study, we observe the case of the amount of 20 and 100 kb data collection. We have observed that when each of the IoT devices process 100 kb (on average) of healthcare data compared to that of 20 kb of healthcare data, the privacy level, the computational latency, and the energy cost of the IoT device with data offloading decrease by 37.12%, 46.12%, 1.61 times, and 26.22%, respectively. If the IoT device has to process 50 kb context-aware data locally in each time slot, as

Authorized licensed use limited to: McMaster University. Downloaded on August 10,2024 at 01:43:20 UTC from IEEE Xplore. Restrictions apply.

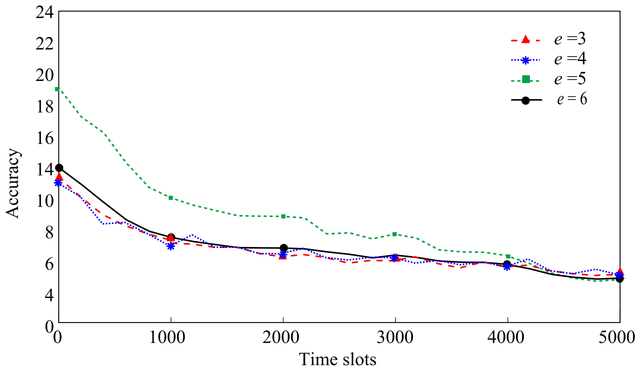

Fig. 4. Performance of C-fDRL in terms of accuracy in the context-aware task offloading.

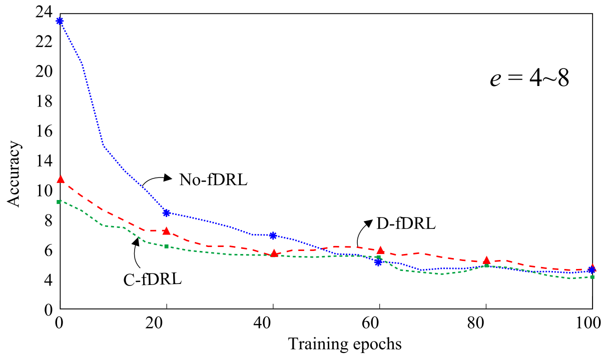

Fig. 5. Performance of the accuracy in the three evaluation settings.

shown in Fig. 4, the privacy level is further increased together with all other aspects.

With regards to the accuracy of FL when providing context-aware data locally, we adopt mean absolute percentage error (MAPE) as the metric to evaluate the privacy-protection accuracy in C-fDRL. Especially, MAPE is studied as the utmost referable metric. MAPE can be defined as

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{D_i - \widehat{D}_i}{D_i}\right| \times 100\% \tag{4}$$

where D_i and $\widehat{D}_i$ are the total amounts of privacy-related data elements and the total amount of local privacy-protected data processing, respectively. For the calculation of privacy context-related data elements in the given data input to the IoT devices, we adopt an element count algorithm that shows the total amount of privacy-related data classifications and isolation through the local fDRL process. Fig. 4 depicts the training process for 100 global epochs with a different number of ECU servers (e) with varying units of healthcare. The proposed C-fDRL is seen to obtain a good performance level compared to another setting, whose accuracy (the percentage of MAPE) surpasses others by at least 40%. This validates the effectiveness of the proposed technical framework for the context-aware FL.

Furthermore, we do compare the results with other frameworks, No-fDRL and D-fDRL, as shown in Fig. 5. We can see that C-fDRL outperforms the other two in terms of accuracy.

TABLE I
Comparison of Privacy Protection Accuracy

Framework	Accuracy (MAPE) (%)	Privacy Protection Level
FASTGNN	8.36	0.81
FedGRU	10.2	0.78
ADDetector	9.78	0.62
STGCN	8.56	0.73
C-fDRL	7.36	0.85

It can be observed that both No-fDRL and D-fDRL have performance degeneration, as f-DRL does perform well in terms of privacy protection in centralized and distributed versions. This leads to lower accuracy, for example, 56% penalty compared to C-fDRL. Moreover, C-fDRL is the only one among these frameworks that can both deal with local contextual information and obtain privacy protection of context locally through the local model training. In C-fDRL, only the local model can access the local network and databases for training, whereas the other two provide limited context information.

Finally, we investigate the accuracy of privacy-preserving when learning and data processing in different approaches. We have nominated a few recent works in the privacy-preserving area that also used FL and DRL approaches. First, we have considered a most closely related work, FASTGNN [29], which provides traffic speed forecasting with FL while guaranteeing privacy preservation. It incorporates a graph-based NN model for local training and a new FL method to protect the privacy of the shared topological data. The second one we have nominated for comparison is ADDetector [4] that proposes an FL-based privacy-preserving smart healthcare system. ADDetector detects Alzheimer's disease while preserving privacy. In addition, we have selected privacy-preserving traffic-flow prediction through FL [30] and STGCNs regarding their privacy-preserving levels [31]. To facilitate a fair comparison, we align the baseline framework with the default super parameters in their corresponding works. The performance in terms of privacy-preserving in these frameworks is depicted in Table I. Comparatively, the FL-based framework, such as FASTGNN, ADDetector, and C-fDRL, performs much better than the conventional framework by at least 7.25% (MAPE). Particularly, the proposed C-fDRL can obtain the same performance level as FASTGNN and STGCN, while C-fDRL outperforms all of them regarding the privacy-preserving level. One of the reasons is that FL with DRL shows high-level performance in privacy-preserving, and various privacy contexts have been considered in the C-fDRL.

VII. Conclusion

In this article, we have proposed C-fDRL, a framework to provide fDRL to maintain the context-aware privacy of the task offloading. We performed this in three stages (CloudAI, EdgeAI, and DeviceAI). C-fDRL checks whether the privacy of high-context-aware data with the task being offloaded is maintained locally at the DeviceAI and low-context-aware data distributedly at the EdgeAI. When there is an offloading task request, or a user needs to offload the data, C-fDRL decouples the context-aware (privacy) data from the tasks. Using a new scheduling technique

Authorized licensed use limited to: McMaster University. Downloaded on August 10,2024 at 01:43:20 UTC from IEEE Xplore. Restrictions apply.

called “context-aware multilevel scheduler”, we performed experiments to evaluate the data privacy with the offloading tasks. The results show that the proposed C-fDRL has done better than the existing framework. We have not worked on private data of the various edge models. We need the accuracy and FPR rate of the generated FL model over DQN for multiple data in various edges (local models) will be our future work. In addition, the stepwise algorithm may reflect the phases that are presented in this article that need further elaboration and may add the related computational complexity of it. Also, the proposed system will be validated on natural systems and have fat conveyance and applicable cases.

References

[1] A. Du, Y. Shen, Q. Zhang, L. Tseng, and M. Aloqaily, “CRACAU: Byzantine machine learning meets industrial edge computing in industry 5.0,” *IEEE Trans. Ind. Informat.*, to be published, doi: 10.1109/TII.2021.3097072.

[2] R.-H. Hsu, J. Lee, T. Q. S. Quek, and J.-C. Chen, “Reconfigurable security: Edge-computing-based framework for IoT,” *IEEE Netw.*, vol. 32, no. 5, pp. 92–99, Sep./Oct. 2018.

[3] T. Wang, G. Zhang, A. Liu, M. Z. A. Bhuiyan, and Q. Jin, “A secure IoT service architecture with an efficient balance dynamics based on cloud and edge computing,” *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4831–4843, Jun. 2019.

[4] J. Li *et al.*, “A federated learning based privacy-preserving smart healthcare system,” *IEEE Trans. Ind. Informat.*, vol. 18, no. 3, pp. 2021–2031, Mar. 2022.

[5] K. Zhang *et al.*, “Compacting deep neural networks for Internet of Things: Methods and applications,” *IEEE Internet Things J.*, vol. 8, no. 15, pp. 11935–11959, Aug. 2021.

[6] C. Sun *et al.*, “Task offloading for end-edge-cloud orchestrated computing in mobile networks,” in *Proc. IEEE Wireless Commun. Netw. Conf.*, 2020, pp. 1–6.

[7] J. Chen, S. Chen, Q. Wang, B. Cao, G. Feng, and J. Hu, “iRAF: A deep reinforcement learning approach for collaborative mobile edge computing IoT networks,” *IEEE Internet Things J.*, vol. 6, no. 4, pp. 7011–7024, Aug. 2019.

[8] X. Zhu, Y. Luo, A. Liu, W. Tang, and M. Z. A. Bhuiyan, “A deep learning-based mobile crowdsensing scheme by predicting vehicle mobility,” *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4648–4659, Jul. 2021.

[9] S. Deng *et al.*, “Dynamical resource allocation in edge for trustable Internet-of-Things systems: A reinforcement learning method,” *IEEE Trans. Ind. Informat.*, vol. 16, no. 9, pp. 6103–6113, Sep. 2020.

[10] R. Taheri, M. Shojafar, M. Alazab, and R. Tafazolli, “Fed-IIoT: A robust federated malware detection architecture in industrial IoT,” *IEEE Trans. Ind. Informat.*, vol. 17, no. 12, pp. 8442–8452, Dec. 2021.

[11] J. Zhang, M. Z. A. Bhuiyan, X. Yang, A. K. Singh, D. F. Hsu, and E. Luo, “Trustworthy target tracking with collaborative deep reinforcement learning in EdgeAI-aided IoT,” *IEEE Trans. Ind. Informat.*, vol. 18, no. 2, pp. 1301–1309, Feb. 2022.

[12] X. Zhu, Y. Luo, A. Liu, M. Z. A. Bhuiyan, and S. Zhang, “Multiagent deep reinforcement learning for vehicular computation offloading in IoT,” *IEEE Internet Things J.*, vol. 8, no. 12, pp. 9763–9773, Jun. 2021.

[13] Q. Yang, Y. Liu, T. Chen, and Y. Tong, “Federated machine learning: Concept and applications,” *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 1–19, 2019.

[14] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, “Differentially private asynchronous federated learning for mobile edge computing in urban informatics,” *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 2134–2143, Mar. 2020.

[15] B. Li, Y. Wu, J. Song, R. Lu, T. Li, and L. Zhao, “Deepfed: Federated deep learning for intrusion detection in industrial cyber-physical systems,” *IEEE Trans. Ind. Informat.*, vol. 17, no. 8, pp. 5615–5624, Aug. 2021.

[16] Y. Zhu, Y. Hu, T. Yang, T. Yang, J. Vogt, and A. Schmeink, “Reliability-optimal offloading in low-latency edge computing networks: Analytical and reinforcement learning based designs,” *IEEE Trans. Veh. Technol.*, vol. 70, no. 6, pp. 6058–6072, Jun. 2021.

[17] Y. Li, F. Qi, Z. Wang, X. Yu, and S. Shao, “Distributed edge computing offloading algorithm based on deep reinforcement learning,” *IEEE Access*, vol. 8, pp. 85204–85215, 2020.

[18] C. Fung, C. J. M. Yoon, and I. Beschastnikh, “Mitigating sybils in federated learning poisoning,” 2018, *arXiv: 1808.04866*.

[19] X. He, H. Lu, M. Du, Y. Mao, and K. Wang, “QoE-based task offloading with deep reinforcement learning in edge-enabled Internet of Vehicles,” *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 4, pp. 2252–2261, Apr. 2021.

[20] Z. Xu *et al.*, “Energy-aware inference offloading for DNN-driven applications in mobile edge clouds,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 4, pp. 799–814, Apr. 2021.

[21] H. Lu, X. He, M. Du, X. Ruan, Y. Sun, and K. Wang, “Edge QoE: Computation offloading with deep reinforcement learning for Internet of Things,” *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9255–9265, Oct. 2020.

[22] S. Song, Z. Fang, Z. Zhang, C.-L. Chen, and H. Sun, “Semi-online computational offloading by dueling deep-Q network for user behavior prediction,” *IEEE Access*, vol. 8, pp. 118192–118204, 2020.

[23] H. Lu, C. Liu, T. He, S. Wang, and K. S. Chan, “Sharing models or coresets: A study based on membership inference attack,” 2020, *arXiv: 2007.02977*.

[24] V. Mnih *et al.*, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.

[25] H.-H. Chang, L. Liu, and Y. Yi, “Deep echo state Q-network (DEQN) and its application in dynamic spectrum sharing for 5G and beyond,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 3, pp. 929–939, 2022.

[26] M. Mohammadi, A. Al-Fuqaha, S. Sorour, and M. Guizani, “Deep learning for IoT Big Data and streaming analytics: A survey,” *IEEE Commun. Surv. Tut.*, vol. 20, no. 4, pp. 2923–2960, Oct.–Dec. 2018.

[27] V. Mnih *et al.*, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, pp. 529–533, 2015.

[28] T. Tieleman and G. Hinton, “Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude,” *Neural Netw. Mach.*, vol. 4, pp. 2407–2416, 2021.

[29] C. Zhang, S. Zhang, J. J. Q. Yu, and S. Yu, “FastGNN: A topological information protected federated learning approach for traffic speed forecasting,” *IEEE Trans. Ind. Informat.*, vol. 17, no. 12, pp. 8464–8474, Dec. 2021.

[30] Y. Liu, J. J. Q. Yu, J. Kang, D. Niyato, and S. Zhang, “Privacy-preserving traffic flow prediction: A federated learning approach,” 2020, *arXiv: 2003.08725*.

[31] B. Yu, H. Yin, and Z. Zhu, “Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting,” in *Proc. 27th Int. Joint Conf. Artif. Intell.*, 2018, pp. 3634–3640.

Yang Xu (Member, IEEE) received the Ph.D. degree in computer science and technology from Central South University, Changsha, China, in 2019.

From 2015 to 2017, he was a Visiting Ph.D. Student with Texas A&M University, TX, USA. He is currently an Associate Professor with the College of Computer Science and Electronic Engineering, Hunan University, Changsha. He has authored or coauthored more than 40 articles in international journals and conferences, including the IEEE TRANSACTIONS ON SERVICES COMPUTING, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING, and IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING. His research interests include network computing, blockchain, and operating system.

Dr. Xu was the recipient of the Best Paper Award of IEEE International Conference on Internet of People (IoP 2018). He is a member of the CCF Technical Committee on Blockchain. He is the Program Committee Chair of the UbiSec 2021 and IWCSS 2020, and a reviewer of more than ten international journals.

Authorized licensed use limited to: McMaster University. Downloaded on August 10,2024 at 01:43:20 UTC from IEEE Xplore. Restrictions apply.

Md Zakirul Alam Bhuiyan (Senior Member, IEEE) received the B.Sc. degree in computer science and engineering from Int'l Islamic University Chittagong, Bangladesh, in 2005, the M.Eng. and Ph.D. degrees in computer science and technology from Central South University, China, in 2009 and 2013, respectively.

He is currently an Assistant Professor with the Department of Computer and Information Sciences and the Founding Director of Fordham Dependable and Secure System Lab (DependSys), Fordham University, Bronx, NY, USA. He has authored or coauthored in many prestigious journals/conferences, and his several research work got recognition of ESI highly cited papers. His research interests include cybersecurity, data-driven dependability, and IoT/CPS applications.

He is a member of the ACM.

Tian Wang received the B.Sc. and M.Sc. degrees from Central South University, Changsha, China, in 2004 and 2007, respectively, and the Ph.D. degree from the City University of Hong Kong, Hong Kong, in 2011, all in computer science.

He is currently a Professor with the Institute of Artificial Intelligence and Future Networks, Beijing Normal University, Guangdong, China, and United International College (UIC), Guangdong, China. He holds 27 patents and has authored or coauthored more than 200 papers in high-level journals and conferences. He has more than 9000 citations, according to Google Scholar. His H-index is 55. He has managed six national natural science projects (including two sub-projects) and four provincial-level projects. His research interests include Internet of Things, edge computing, and mobile computing.

Xiaokang Zhou (Member, IEEE) received the Ph.D. degree in human sciences from Waseda University, Tokyo, Japan, in 2014.

From 2012 to 2015, he was a Research Associate with the Faculty of Human Sciences, Waseda University. Since 2017, he has been a visiting Researcher with the RIKEN Center for Advanced Intelligence Project (AIP), RIKEN, Japan. He is currently an Associate Professor with the Faculty of Data Science, Shiga University, Shiga, Japan. His research interests include computer science and engineering, information systems, social and human informatics, ubiquitous computing, Big Data, machine learning, behavior, cognitive informatics, cyber-physical-social-system, cyber intelligence, and security.

Dr. Zhou is a member of the IEEE CS, ACM, USA, IPSJ, JSAI, Japan, and CCF, China.

Amit Kumar Singh (Senior Member, IEEE) received the B.Tech. and M.Tech. degrees from the Department of Computer Science and Engineering from the National Institute of Technology (NIT) Kurukshetra, Haryana, India, in 2005 and 2010, respectively, and the Ph.D. degree from the Department of Computer Engineering from NIT, in 2015.

He is currently an Assistant Professor with the Computer Science and Engineering Department, NIT Patna, Bihar, India. He has authored or coauthored more than 150 publications and his publications appeared in many top journals, including the IEEE Transactions on Industrial Informatics and IEEE Transactions on Dependable and Secure Computing. His research interests include multimedia security, data hiding, image processing, and cryptography.

Authorized licensed use limited to: McMaster University. Downloaded on August 10,2024 at 01:43:20 UTC from IEEE Xplore. Restrictions apply.