Deep Reinforcement Learning-Based Joint Optimization of Delay and Privacy in Multiple-User MEC Systems

Ping Zhao, *Member, IEEE*, Jiawei Tao, Kangjie Lui, Guanglin Zhang, *Member, IEEE*, and Fei Gao

Abstract—Multi-access Edge Computing (MEC) enables mobile users to run various delay-sensitive applications via offloading computation tasks to MEC servers. However, the location privacy and the usage pattern privacy are disclosed to the untrusted MEC servers. The most related work concerning privacy-preserving offloading schemes in MEC either consider an impractical MEC scenario consisting of a single user or take a large amount of computation and communication cost. In this article, we propose a deep reinforcement learning based joint optimization of delay and privacy preservation during offloading for multiple-user wireless powered MEC systems, preserving users' both location privacy and usage pattern privacy. The main idea is that, to protect both the two kinds of privacy, we propose to disguise users' offloading decisions and deliberately offloading redundant tasks along with the actual tasks to the MEC servers. On this basis, we further formalize the task offloading as an optimization problem of computation rate and privacy preservation. Then, we design a deep reinforcement learning based offloading algorithm to solve such an non-convex problem, aiming to obtain the better tradeoff between the computation rate and the privacy preservation. Finally, extensive simulation results demonstrate that our algorithm can maintain a high level of computation rate while protecting users' usage pattern privacy and location privacy, compared with two learning-based methods and two Baselines.

Index Terms—Mobile edge computing, task offloading, privacy preservation, deep reinforcement learning, computation rate

1 Introduction

Multi-access Edge Computing (MEC), as an alternative solution of the centralized mobile cloud computing, enables the resource-constrained mobile devices to offload the computation tasks generated by various delay-sensitive applications, e.g., face recognition, interactive gaming, augmented reality and healthcare monitoring [1], [2], to resource-rich MEC servers, thereby significantly reducing both the workload and the execution latency of mobile devices. Nonetheless, the task offloading in MEC entails privacy risks of mobile users [3], [4], [5], [6], [7].

- *Ping Zhao is with the College of Information Science and Technology, Donghua University, Shanghai 201620, China, and also with the State key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China. E-mail: pingzhao2014ph@gmail.com.*
- *Jiawei Tao, Kangjie Lui, and Guanglin Zhang are with the College of Information Science and Technology, Donghua University, Shanghai 201620, China. E-mail: 2191477@mail.dhu.edu.cn, 2466765750@qq.com, glzhang@dhu.edu.cn.*
- *Fei Gao is with the State key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China. E-mail: gaof@bupt.edu.cn.*

Manuscript received 14 September 2021; revised 27 December 2021; accepted 29 December 2021. Date of publication 4 January 2022; date of current version 7 June 2023.
This work was supported in part by the National Natural Science Foundation of China under Grant 61902060, in part by the Open Foundation of State key Laboratory of Networking and Switching Technology (Beijing University of Posts and Telecommunications) under Grant SKLNST-2021-1-06, and in part by Shanghai Sailing Program under Grant 19YF1402100.
(Corresponding author: Guanglin Zhang.)
Recommended for acceptance by C. A. Ardagna.
Digital Object Identifier no. 10.1109/TCC.2022.3140231

Users' *location privacy* and *usage pattern privacy* are breached by the untrusted MEC servers during task offloading [3]. Specifically, as shown in Fig. 1, the MEC server S_1 can infer that the mobile user u_3 is moving away from it during time period (t_1, t_3) via analyzing the size of the offloaded tasks from u_3, since a user away from the MEC server, is very likely to locally computes the computation tasks under the severe radio channel condition [3], [4]. More seriously, more accurate moving trajectory of u_3 can be inferred when several MEC servers collude with each other, e.g., S_1 and S_2. As a result, the location privacy of mobile user u_3 is disclosed to the untrusted MEC servers S_1 and S_2. To make matters worse, the user u_3 may be thereby vulnerable to serious attacks, e.g., spams, or even blackmails and physical violence, etc. In addition, MEC servers S_1 and S_2 can evaluate the usage pattern of users u_2 and u_5 respectively via estimating the size of the offloaded tasks, when u_2 and u_5 offload the tasks to the servers under good radio channel state [3]. Therefore, it is necessary to design the privacy-preserving offloading scheme in MEC systems.

While a great deal of studies concerning offloading in MEC have concentrated on minimizing both the energy consumption and the delay, there is less attention in the equally important problem of privacy preservation in task offloading. Specifically, most related work investigated the offloading schemes in single-user [8], [9], [10], [11], [12], [13] or multiple-user [14], [15], [16], [17], [18], [19], [20], [21] MEC systems. However, these work neither considered the privacy disclosure during task offloading nor proposed the corresponding privacy-preserving algorithms to defend such privacy risks. Another kind of work [22], [23], [24], [25], [26],

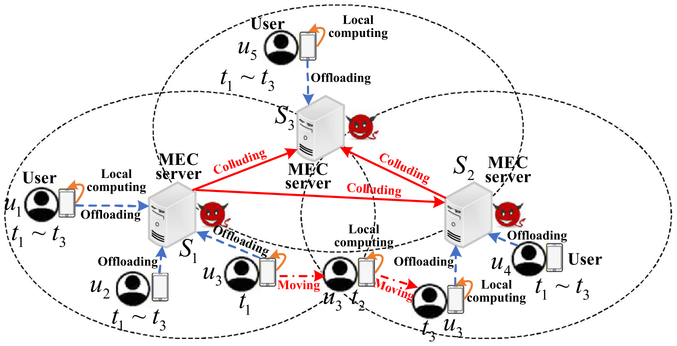


Fig. 1. The illustration of mobile edge computing (MEC) and the privacy disclosure during the task offloading in MEC.

[27], [28], [29], [30] focused on privacy-preserving MEC systems. Nevertheless, these work mainly concentrated on privacy preservation inherited from the conventional cloud computing architecture, without studying the privacy issues unique to MEC (i.e., the optimization of computation delay and the privacy preservation in task offloading). The third kind of related work concerning privacy-preserving offloading scheme in MEC is largely classified into studies based on encryption [31], [32], [33], [34], physical-layer techniques [12], [35], [36], and learning [3], [37], [38]. Unfortunately, these schemes based on encryption take a large amount of computation and communication cost, which is not applicable to the resource-constrained mobile devices in practical MEC scenarios. Studies based on physical-layer techniques prevent the privacy disclosure from another perspective, physical layer, which is another research topic. Studies based on learning only considered the MEC system consisting of a single user and a single edge server, which is quite impossible in the practical MEC scenarios. Overall, it is necessary to design privacy-preserving offloading algorithm that is lightweight, specifically intended for the MEC architecture, and applicable to more practical MEC scenarios consisting of more users and servers.

To address the above-mentioned problems, we propose a deep reinforcement learning based joint optimization of delay and privacy preservation during offloading for multiple-user MEC systems, preserving users' both location privacy and usage pattern privacy. The main idea is that, to protect both the two kinds of privacy, we propose to disguise users' offloading decisions especially when users are moving away from the MEC server and suffering from the severe radio channel condition, and moreover deliberately send redundant tasks (i.e., redundant information) along with the actual tasks to the MEC server. On this basis, we further formalize the task offloading as an optimization problem to obtain the better tradeoff between the computation rate and the privacy preservation.

Specifically, we first take into account the wireless channel power gains between the MEC server with multiple devices, the computation capacities and energy constraints of multiple devices, and further build the local computation model and edge computation model. Then, we formalize the privacy protection model via disguising users' offloading decisions and deliberately offloading redundant tasks. Thereafter, on this basis, we formalize a joint optimization problem of delay and privacy, and design a deep reinforcement learning based privacy-aware task offloading scheme to solve such a problem, aiming to achieve a larger computation rate and a larger privacy level. At last, we evaluate the performance of our algorithm via comparing our work with the two learning-based methods and two Baselines.

However, we are facing the following two *challenges*:

- We formalize the joint optimization of computation rate and privacy preservation as a non-convex mixed integer programming problem, since the offloading decisions are unknown.
- It is difficult to deal with the continuous channel gains while preserving privacy.

To addressing these challenges, we make the following main *contributions*:

- To deal with the first challenge, we propose deep reinforcement learning based privacy-aware task assignment algorithm to joint optimize the computing mode decision, the privacy protection and the system time allocation. Specifically, the decision and the size of redundant information can be obtained by the proposed neural network, and then, the optimization problem can be reduced to a convex one.
- To addressing the second challenge, we introduce a new reward for privacy protection. To concrete, the reward for protecting privacy depends on the lost computation rate and the weights of the location privacy and the usage pattern privacy.

The remainder of this paper is organized as follows. Section 2 reviews the related work. Then, Section 3 presents some preliminary knowledge. Section 4 introduces the system models and the problem formalization. Thereafter, Section 5 presents the proposed offloading scheme in detail, following by the performance evaluation in Section 6. Finally, Section 7 concludes the paper.

2 Related Work

2.1 Task Offloading in MEC Systems

Task Offloading in Single-User MEC Systems. The work [8] designed a partial computation offloading model, minimizing the latency and energy cost. Likewise, the follow-up work [9], [10], [11] took the task queues and decomposing the tasks into consideration respectively. The latest work [12], [13] proposed to use physical-layer approaches and a powerful hierarchical layered offloading mechanism respectively.

Task Offloading in Multiple-User MEC Systems. The work [16], [17], [18], [19] considered task dependency between the two devices, code-oriented partitioning, cached data, and users' mobility in offloading respectively. The works [20], [21], [39] proposed fine-grained task offloading and studied the offloading in hierarchical MEC architecture respectively. The latest works [40], [41], [42] focused on the multi-server multi-access edge computing environment.

However, both the two kind of work above only investigated the offloading schemes in MEC systems, and neither considered the privacy disclosure during task offloading nor proposed the corresponding privacy-preserving algorithms to defend such privacy risks. In contrast, this paper proposed the privacy-preserving offloading scheme that optimizes both the privacy preservation and the cost in task offloading.

2.2 Privacy-Preserving MEC Systems

Work [22], [23] and the references therein aimed at the private information risk imparted from the traditional MEC computing structure. The follow-up studies [24], [25] utilized chaff services to guard against eavesdropping. Thereafter, literatures [26], [43], [44], [45] studied machine learning based privacy preservation, and another work [27] used asymmetric encryption to guarantees data privacy, authentication, and integrity. The work [28] introduced an economics model for MEC bearing physical layer offloading intuition. The latest work [30] used feature learning deduced from the social graph.

However, these work above mainly concentrated on privacy preservation imparted from the traditional cloud offloading structure, without studying the privacy disclosure risks distinctive to MEC (i.e., the optimization of computation delay and the privacy preservation in task offloading). In contrast, in our work, we take both the location privacy and the usage pattern privacy in task offloading into consideration, then formalize the computation delay and the two kind of privacy during task offloading as an joint optimization problem, and finally design an deep reinforcement learning-based algorithm to solve such an optimization problem.

2.3 Privacy-Preserving Offloading in MEC Systems

Several work focused on privacy-preserving offloading scheme in MEC, which is largely classified into three kinds: studies based on encryption, physical-layer techniques, and learning. Studies based on encryption [31], [32], [33], [34] *take a large amount of computation and communication cost, which is not applicable to the resource-constrained mobile devices in practical MEC scenarios. In this work, we design an lightweight deep reinforcement learning-based algorithm, concentrating on reducing the computation delay of task offloading and guarantee both the usage pattern privacy and the location privacy. After the training, the machine learning of our algorithm can get the offloading decision quickly.* Studies based on physical-layer techniques [12], [35], [36] *prevent the privacy disclosure from another perspective, physical layer, which is another research topic. In contrast, our work aims at protecting private information in the data flow when tasks are offloaded to MEC servers.* Studies based on learning [3], [37], [38] *considered the MEC system with only one user and one edge server, which is quite impossible in the practical MEC scenarios. In contrast, we consider the multiple-user MEC system which consists of multiple users, and one edge server.*

3 Preliminary

3.1 Privacy Disclosure in MEC

In MEC, users' location privacy and usage pattern privacy may be breached by the untrusted MEC servers. To make matters worse, users will suffer from various attacks, in the event of the disclosure of the location privacy and usage pattern privacy.

Disclosure of Location Privacy. As shown in Fig. 1, the mobile user u_3 offloads tasks to the MEC server S_1 at time t_1, and locally computes tasks at time t_2. Then, the MEC server S_1 can infer that the mobile user u_3 is moving away from it during the time period (t_1, t_2), since a user far away from the MEC server is likely to locally compute the tasks under bad radio channel condition [46]. To make matters worse, when MEC servers S_1, S_2 and S_3 collude, the accurate trajectory of u_3 can be inferred. Specifically, u_3 offloads tasks to the MEC server S_2 at time t_3. In such a case, MEC servers S_1, S_2 and S_3 can infer that u_3 moves from the coverage area of MEC server S_1 to the coverage area of MEC server S_2 without passing the coverage area of MEC server S_3. As a result, the trajectory of u_3 is disclosed to MEC servers S_1, S_2 and S_3.

Disclosure of Usage Pattern Privacy. As shown in Fig. 1, the MEC server S_1 can record and analyse the size of the offloaded tasks from mobile users u_1, u_2 within time $t_1 \sim t_3$ and u_3 at time t_1, since users running different applications on mobile devices exhibit different usage pattern privacy [47]. For example, a specific pregnant woman running the baby-care apps will have different usage patterns with the office workers. Moreover, a young man addicted to games exhibits different usage pattern with a patient who runs the health monitoring APP on mobile phone. In a nutshell, users exhibit the unique usage patterns, and such unique usage patterns can help the untrusted MEC servers to identify a specific user from a set of anonymous users.

3.2 Adversary Model

In this paper, we assume that the MEC server is untrusted and regarded as an adversary, as in the existing work [48], [49], [50], [51], [52]. It means that, on one hand, the MEC server honestly receives the offloaded tasks from users, performs the computations, and returns the corresponding results to users. On the other hand, it may try to reveal the location privacy and usage pattern privacy of users for, e.g., commercial interests, and so on. In the following, the background knowledge and goal of the adversary are presented in detail.

Adversary's Knowledge. Users offload their tasks to the adversary (i.e., MEC server), and the adversary computes the results and returns these results to users. In this process, the adversary is assumed to know the source device which sends the tasks to the adversary, and also get the knowledge of the size of the arrival tasks (in bits). To concrete, when the adversary receives a task, it can know where the task is sent from, and also know the the size of the task.

Adversary's Goal. The adversary dedicates to reveal the two kinds of privacy information of users, i.e., usage pattern privacy and location privacy. Specifically, when the adversary receives a task, it first has to obtain the current channel condition. If the observed channel condition is good enough, the adversary could reveal the usage pattern privacy of the user, with the help of the background information, i.e., the source device and the size of the arrival task. In such a case, the level of privacy leakage depends on the size of the offloaded task. When the channel condition is bad and the adversary does not receive the task, the adversary could infer the user's location privacy. Namely, the adversary can infer that the user is far away from it, since the user is very likely to locally computes the task with the bad channel condition.

3.3 Goal of Design

In this paper, our goal is to achieve the joint optimization of delay and privacy preservation in the process of task

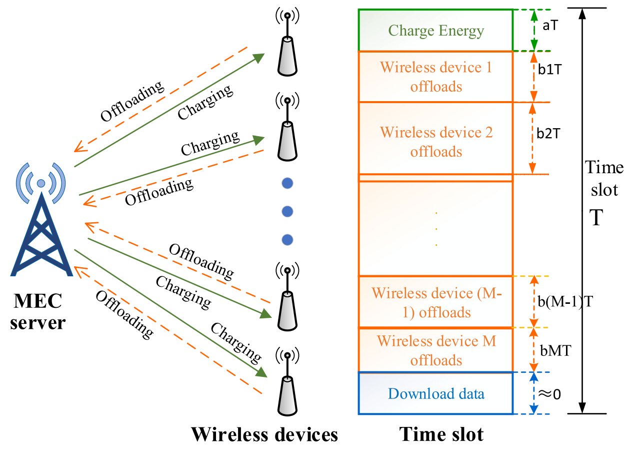


Fig. 2. Illustration of system model and time allocation.

offloading in MEC systems. Specifically, the usage pattern privacy and location privacy may be revealed in task offloading and further be utilized by the adversary to infer more personal information of users. For example, the usage pattern privacy of a specific user will enable the adversary to identify the user from a set of anonymous users. The patient who runs the health monitoring APP on mobile phone tends to own the unique usage pattern in contrast to other average users [35]. Likewise, when the channel condition is bad and the adversary does not receive the task, the adversary could infer the user's location privacy. Then, with the help of location privacy, the adversary can infer users' religious belief, habits, health status, and so on [47]. Therefore, it is important to protect the usage pattern privacy and location privacy. In addition, in process of task offloading, locally computation and task transmission definitely incur the computation delay. Therefore, we design the optimal offloading scheme to minimize delay and obtain the highest privacy level. This algorithm can be deployed to run on an MEC server with rich computation resources.

4 System Model And Problem Formulation

As shown in Fig. 2, the MEC system consists of one MEC server and M wireless mobile devices. The MEC server links to a stable power supply, and the wireless mobile devices can be powered by the MEC server through Radio Frequency(RF) wireless charging technology. It is worth noting that the MEC server can charge multiple wireless mobile devices at the same time, and the wireless devices can store the power for computation tasks. For the stability of our system, the wireless devices only harvest energy once at the begin of each time slot. We assume that the length of a time slot is T, and the set of time slots can be denoted by $\mathcal{T} = \{0, 1, \ldots, \}$. The set of wireless devices is denoted by $\mathcal{M} = \{1, 2, \ldots, M\}$. At each time slot, all wireless devices will generate computation tasks locally, and these tasks need to be computed locally or be offloaded to the MEC server. We assume that the tasks created by the wireless devices in our system are indivisible for the sake of data security. So tasks can be only executed locally or be offloaded to the MEC server. We define $\mathcal{M}_0$ and $\mathcal{M}_1$ as two mutually exclusive sets, where $\mathcal{M}_0$ contains the wireless devices executing tasks locally and $\mathcal{M}_1$ consists of the wireless devices offloading tasks to the MEC server. Therefore, we can get $\mathcal{M}_0 \cup \mathcal{M}_1 = \mathcal{M}$.

TABLE 1
List of Notations

Symbol	Description
$\mathcal{M}$	Set of wireless devices
$\mathcal{T}$	Set of time slots
h_m	Channel power gain at device m
α_m^L	Device m executes tasks locally
α_m^O	m offloads task to protect usage pattern privacy
α_m^P	m offloads task to protect two kinds of privacy
E_m	Energy obtained by the mth wireless device
$r_{L,m}^*$	Maximum local computation rate in a time slot
D_m	Size of offloaded tasks of device m
b_m	Time the device m needs to offload tasks
P_m^o	The transmit power of the device m
$r_{O,m}^*$	Maximum computation rate at MEC server
$P_{l,m}$	The reward of protecting location privacy
$P_{\mu,m}$	The reward of protecting usage pattern privacy
P_m	Level of privacy preservation for device m

Each device in our system can have five kinds of offloading decisions at one time slot. We introduce five indicators to denote the five decisions, i.e., α_m^1, α_m^2, α_m^3, α_m^4 and α_m^5 respectively. Offloading decision $\alpha_m^1 = 1$ means that the wireless device m executes the tasks locally. Offloading decision α_m^2 means that the wireless device m offloads tasks to the MEC server but does not protect privacy. Offloading decision α_m^3 means that the wireless device m offloads tasks to the MEC server and only protects the location privacy. The offloading decision α_m^4 means that the wireless device m offloads tasks to the MEC server and only protects the usage pattern privacy. The offloading decision α_m^5 means that the wireless device m offloads tasks to the MEC server and protects both the usage pattern privacy and the location privacy simultaneously. So the decision set of the wireless device m at one time slot can be described as $\boldsymbol{\alpha}_m \triangleq \{\alpha_m^1, \alpha_m^2, \alpha_m^3, \alpha_m^4, \alpha_m^5\}$, where $\alpha_m^1 \cup \alpha_m^2 \cup \alpha_m^3 \cup \alpha_m^4 \cup \alpha_m^5 = 1$ and $\alpha_m^1 \cap \alpha_m^2 \cap \alpha_m^3 \cap \alpha_m^4 \cap \alpha_m^5 = \varnothing$. Furthermore, the decision indicators should satisfy the following operation constraints.

$$\sum_{m \in \mathcal{M}} \alpha_m = M, \tag{1}$$

where $\alpha_m \in \boldsymbol{\alpha}_m$.

At the beginning of each time slot, wireless devices harvest energy for computation or transmitting tasks. The energy obtained by the mth wireless device is

$$E_m = \mu P h_m a T \quad (m = 1, \ldots, M), \tag{2}$$

where μ denotes the effectiveness factor of harvesting energy, P is the energy power transported by wireless charging equipment at one time slot, h_m denotes the wireless channel power gain between the mth device and the MEC server, $a(0 < a \leqslant 1)$ represents the protortion of charging time in a time slot, and aT denotes the charging time. We assumed that h_m is same for downlink and uplink, and that h_m is static within one time slot. Wireless devices can execute tasks locally while charging energy. The important notations in this paper are listed in Table 1.

4.1 Local Computation Model

At a certain time slot, when the tasks are executed locally, according to the Law of Conservation of Energty, the consumed energy $k_m f_m^3 \tau_m$ must be less than or equal to E_m, where k_m denotes energy efficiency factor, f_m is the local CPU's process rate, and τ_m denotes the time spent on computation. Then, we can easily get the amount of data (i.e., tasks) locally computed $f_m \tau_m / \phi$ where ϕ is the needed CPU cycles for computing one bit of data. Thus, we could get the local computation rate in one time slot

$$r_{L,m} = \frac{f_m \tau_m}{\phi T}. \tag{3}$$

In order to maximize the amount of data computed locally as well as the local computing rate, we assume the wireless device exhausts its harvested energy, i.e., $k_m f_m^3 \tau_m = E_m$, and it can continuously run all over the time slot as a device can harvest energy and compute tasks at the same time, i.e., $\tau_m = T$. Therefore, on the basis of Eq. (3), by substituting $f_m = (\frac{E_m}{k_m \tau_m})^{\frac{1}{3}}$ and $\tau_m = T$ into Eq. (3), we can get the local computation rate

$$r^*_{L,m} = \frac{(\frac{E_m}{k_m \tau_m})^{\frac{1}{3}}}{\phi} = \frac{(\frac{\mu P h_m a}{k_m})^{\frac{1}{3}}}{\phi} = \eta_1 (\frac{h_m}{k_m})^{\frac{1}{3}} a^{\frac{1}{3}}, \tag{4}$$

where $\eta_1 = \frac{1}{\phi}(\mu P)^{\frac{1}{3}}$.

4.2 Edge Computation Model

We assume that all devices communicate with the MEC server in the same frequency band. One MEC server just can receive the tasks from one wireless device at the same time, because of the limitation of time-division-multiplexing. So, after harvesting energy, these wireless devices offload their tasks to the MEC server one by one. At a certain time slot, the time that wireless device m needs to offload tasks is denoted by $b_m T$ where $b_m \in [0, 1)$ represents the proportion of offloading time in a time slot. Therefore, we can obtain the size of the offloaded tasks of the wireless device m (in bits)

$$D_m = \frac{B b_m T}{v_m} \log_2 (1 + \frac{P^o_m h_m}{N_0}), \tag{5}$$

where B denotes the communication bandwidth between the wireless device m and the MEC server, P^o_m denotes the transmit power of the device m, N_0 denotes the power of noise, and $v_m \geq 1$ denotes the redundancy of offloaded tasks, such as overhead message. Obviously, the interference free environment is unrealistic. So we simulate the channel quality in different environments through different h_m. When we encrypt the transmitted tasks, the redundancy will increase inevitably.

Based on our research, we assume that both the computing resources and the transmitting ability of the MEC server are much better than that of these wireless devices. So we ignore the time of computation on MEC server, the time of downloading feedback from the MEC server, and the consumed energy for receiving the feedback. In a word, it follows the relationship between a and b_m:

$$\sum_{m \in \mathcal{M}_1} b_m + a \leq 1. \tag{6}$$

Hence, we can see that the MEC server's computation rate that can be supplied to one device equals to the task offloading rate. To maximize the task offloading rate, we assume that the device uses up the harvested energy in offloading. This suggests that $P^o_m = \frac{E_m}{b_m T}$, and we can obtain the maximum computation rate of the wireless device m

$$\begin{aligned} r^*_{O,m} = \frac{D_m}{T} &= \frac{B b_m}{v_m} \log_2 (1 + \frac{\mu P^o_m a h^2_m}{b_m N_0}) \\ &= \frac{\varepsilon b_m}{v_m} \ln (1 + \frac{\eta_2 a h^2_m}{b_m}), \end{aligned} \tag{7}$$

where $\eta_2 \triangleq \frac{\mu P^o_m}{N_0}$, $\varepsilon = \frac{B}{\ln 2}$. Therefore, the whole computation rate of all M wireless devices is

$$R = \sum_{m \in \mathcal{M}_0} \eta_1 (\frac{h_m}{k_m})^{\frac{1}{3}} a^{\frac{1}{3}} + \sum_{m \in \mathcal{M}_1} \frac{\varepsilon b_m}{v_m} \ln (1 + \frac{\eta_2 a h^2_m}{b_m}). \tag{8}$$

To unify the equation above, we introduce one indicator variable x_m

$$x_m = \begin{cases} 0, m \in \mathcal{M}_0, \\ 1, m \in \mathcal{M}_1. \end{cases} \tag{9}$$

In this way, we can turn Eq. (8) into the following equation:

$$\begin{aligned} R = \sum_{m=1}^{M} ((1 - x_m) \eta_1 (\frac{h_m}{k_m})^{\frac{1}{3}} a^{\frac{1}{3}} \\ + x_m \frac{B b_m}{v_m} \ln (1 + \frac{\eta_2 a h^2_m}{b_m})). \end{aligned} \tag{10}$$

4.3 Privacy Protection Model

4.3.1 Location Privacy

Based on the system model we proposed above, users are more likely to offload the tasks to MEC server when wireless channel gain between the MEC server and the devices are large enough. So, wireless devices' location information is associated with the wireless channel gain. Since the untrusted MEC server knows the offloaded tasks from a specific wireless device, it can obtain the distance between the wireless device and the MEC server. Moreover, by comparing the offloaded tasks of the wireless user at different time, the untrusted MEC server can obtain the moving trajectory of the device. To make matters worse, when several MEC servers collude, they can get the accurate location of the wireless device. To protect the location privacy, the wireless devices can offload some tasks deliberately when the channel condition is not good enough to offload tasks. Accordingly, the reward of protecting location privacy can be formalized as

$$P_{l,m} = \mathbb{E} \frac{B b_m}{v_m} \ln (1 + \frac{\eta_2 a h^2_m}{b_m}) \cdot \mathbb{I}(x_m = 1) \cdot \mathbb{I}(x'_m = 0), \tag{11}$$

where x'_m denotes the task allocation strategy without privacy protection. Parameter x'_m is only related to the wireless channel gain h_m. $\mathbb{E}$ means the mathematical expectation of the improvement of computation rate from $x'_m = 0$ to $x_m = 1$ divided by the offload computation rate. The process of obtaining parameter $\mathbb{E}$ is shown in Algorithm 1. The computation complexity of Algorithm 1 is mainly affected by the two for loops. Obviously, the computation complexity is $O(|\mathcal{T}||M|)$ at most, where $|\mathcal{T}|$ and $|M|$ mean the number of parameters t and m respectively.

Algorithm 1. How to get $\mathbb{E}$

```
Require: Wireless channel gain h, task allocation strategy $x'_m$
1: Create an empty array $E$
2: for $t$ in $\mathcal{T}$ do
3:     for $m$ in $M$ do
4:         if $x'_m = 0$ then
5:             Compute $\Delta R$ between $x'_m = 0$ and $x_m = 1$
6:             Add $\frac{\Delta R}{r_{O,m}}$ to the array $E$
7: $\mathbb{E}$ is equal to the mathematical expectation of all elements
   in $E$
```

4.3.2 Usage Pattern Privacy

During a time slot, the channel energy gain between the wireless device m and the MEC server is relatively stable. Moreover, the tasks created by one function or APP have the same format. As a result, the wireless device m's usage pattern information is associated with the size of the offloaded tasks. Thus, by analyzing the tasks' size of the device m within a time period, the untrusted MEC server can obtain the user's usage pattern easily. To protect the user's usage pattern privacy, we propose to deliberately transmit some redundant tasks when the user offloads tasks to the MEC server. Note that the redundant tasks are the exact historical tasks of users. Accordingly, the reward of protecting usage pattern privacy can be formalized as

$$\begin{aligned} P_{u,m} &= (\frac{1}{v'_m} - \frac{1}{v_m})\varepsilon b_m \ln(1 + \frac{\eta_2 a h_m^2}{b_m}) \cdot \mathbb{I}(x_m = 1) \\ &= \Delta\varepsilon b_m \ln(1 + \frac{\eta_2 a h_m^2}{b_m}) \cdot \mathbb{I}(x_m = 1), \end{aligned} \tag{12}$$

where $\Delta = \frac{1}{v'_m} - \frac{1}{v_m}$. Parameter v'_m denotes the initial redundancy without privacy protection.

In summary, on the basis of the location privacy and the usage pattern privacy, the level of privacy preservation for the wireless device m can be formalized as

$$P_m = \beta_1 P_{l,m} + \beta_2 P_{u,m}, \tag{13}$$

where β_1 and β_2 denote the weight of location privacy and usage pattern privacy. By adjusting β_1 and β_2, we can balance the computing rate and the level of privacy protection, which will be introduced in detail in the following.

4.4 Problem Formulation

Based on the models above, enhancing the privacy preservation definitely decreases the computation rate, and vice versa. To obtain the better balance between the privacy preservation and the computation rate, we first formalize the weighted sum of the computation rate and the privacy level at a certain time slot

$$\begin{aligned} Q(\mathbf{h}, \mathbf{x}, \mathbf{v}, \mathbf{b}, a) = \sum_{m=1}^{M} &((1 - x_m)\eta_1(\frac{h_m}{k_m})^{\frac{1}{3}} a^{\frac{1}{3}} \\ &+ x_m \frac{\varepsilon b_m}{v_m} \ln(1 + \frac{\eta_2 a h_m^2}{b_m}) \\ &+ \beta_1 x_m (1 - x'_m)\mathbb{E}\frac{B b_m}{v_m} \ln(1 + \frac{\eta_2 a h_m^2}{b_m}) \\ &+ \beta_2 x_m \Delta\varepsilon b_m \ln(1 + \frac{\eta_2 a h_m^2}{b_m}) \end{aligned} \tag{14}$$

$$s.t.\ Eq.\ (6), v_m \geqslant 1, b_m \geqslant 0, a \geqslant 0, x_m \in \{0, 1\},$$

where $\mathbf{h} = \{h_m | m \in \mathcal{M}\}$, $\mathbf{x} = \{x_m | m \in \mathcal{M}\}$, $\mathbf{v} = \{v_m | m \in \mathcal{M}\}$ and $\mathbf{b} = \{b_m | m \in \mathcal{M}\}$ are vectors of h_m, x_m, v_m and b_m respectively, and other parameters are fixed (e.g., P, B and so on). As it can be seen from the problem above, there is a tradeoff between the privacy protecting and the computation rate. So, in our work, the objective is to find a better balance which achieves a higher computation rate and a higher privacy level at the same time. Therefore, with the given wireless channel gain $\mathbf{h}$, we formulate the problem that maximizes the sum of local computation rate, offloading computation rate, and the reward of privacy protection as

$$P1: \quad \max_{\mathbf{x},\mathbf{v},\mathbf{b},a} Q(\mathbf{h}, \mathbf{x}, \mathbf{v}, \mathbf{b}, a), \tag{15}$$

$$s.t.\ (6), v_m \geqslant 1, b_m \geqslant 0, a \geqslant 0, x_m \in \{0, 1\}.$$

Therefore, Equ. (15) can take into account the constraints on energy and delay while improving the computing power of the system and protecting user privacy. For example, a larger a seems to lead to a larger result. But in fact, a larger a means more charging time, i.e., delay. At the same time, this also means that the device can harvest more energy. In the scenario of MEC, this will make the device tend to local computing rather than offloading to the edge server with stronger computing power, which will lead to the decline of computing power. But $P1$ is a mixed integer programming non-convex problem. It's very difficult to solve such an optimization problem. Nevertheless, once $\mathbf{x}$ and $\mathbf{v}$ are given, $P1$ can be modified to a convex problem

$$P2: \max_{\mathbf{b},a} Q(\mathbf{h}, \mathbf{b}, a) \tag{16}$$

$$s.t.\ (6), v_m \geqslant 1, b_m \geqslant 0, a \geqslant 0, x_m \in \{0, 1\}.$$

This observation motivates us to propose a deep reinforcement learning based privacy-aware task assignment algorithm to solve such an optimization problem, which will be introduced in the next section.

5 Deep RL Based Privacy-Aware Task Offloading Algorithm

In this section, to solve the optimization problem $P2$, we propose a deep reinforcement learning based privacy-aware task offloading algorithm. Specifically, as shown in Fig. 3, the proposed offloading algorithm consists of two steps. The first step is to seek appropriate $\mathbf{b}$ and a that maximize $Q(\mathbf{h}, \mathbf{b}, a)$, with the given $\mathbf{h}_t$, $\mathbf{x}$, and $\mathbf{v}$. The corresponding $\mathbf{h}_t$ and optimal output $\{\mathbf{x}, \mathbf{v}\}$ of Deep Neural Networks (DNNs) that maximize $Q(\mathbf{h}, \mathbf{b}, a)$ will be stored in training

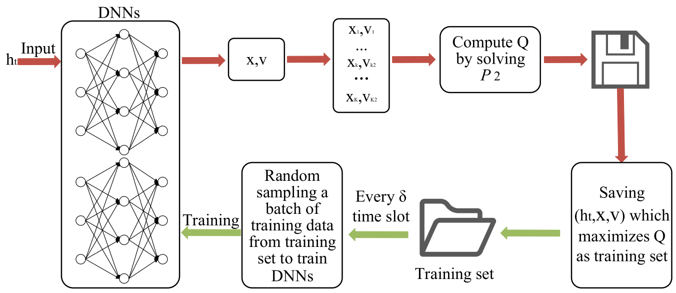

DNNs
Input
h_t
x,v
x1,v1
...
xk2,vk2
Compute Q by solving P2
Saving (ht,x,v) which maximizes Q as training set
Training set
Every δ time slot
Random sampling a batch of training data from training set to train DNNs
Training

Fig. 3. Work flow of the deep reinforcement learning based privacy-aware task offloading algorithm.

set. The second step is to train the DNNs with the training set obtained in the first step every δ time slots.

5.1 Find Out b and a

In this part, we will find out $\mathbf{b}$ and a by solving $P2$, given $\mathbf{x}$ and $\mathbf{v}$. To solve $P2$, we introduce a Lagrangian multiplier ξ to constrain Eq. (6). Then, we can obtain the Lagrangian form of $P2$

$$\begin{aligned} L(a,\mathbf{b},\xi) = &\sum_{m\in\mathcal{M}_0} \eta_1 (\frac{h_m}{k_m})^{\frac{1}{3}} a^{\frac{1}{3}} \\ &+ \sum_{m\in\mathcal{M}_1} (\beta_2\Delta + \frac{1}{v_m})\varepsilon b_m \ln(1+\frac{\eta_2 a h_m^2}{b_m}) \\ &+ \beta_1 \sum_{m\in\mathcal{M}_1} (\mathbb{E}\frac{Bb_m}{v_m}\ln(1+\frac{\eta_2 a h_m^2}{b_m})) \cdot \mathbb{I}(x'_m = 0) \\ &+ \xi(1 - a - \sum_{m\in\mathcal{M}_1} b_m); \\ &\quad s.t. \quad a, b_m \geqslant 0. \end{aligned} \tag{17}$$

The Lagrangian dual function is

$$\min_\xi\{\max_{a,\mathbf{b}} L(a,\mathbf{b},\xi) | \xi \geqslant 0\}. \tag{18}$$

Given $\mathbf{v}$ and $\mathbf{x}$, $P2$ is a convex problem, and thereby the dual problem in Eq. (18) can be solved. The final result satisfies the following conditions:

$$\textstyle\sum_{m\in\mathcal{M}_1} b_m + a = 1. \tag{19}$$

Therefore, we can obtain the following results. With the help of the following results, I can further get the relationship between a and b_m, and find out the value of a and b_m.

Theorem 1. *The relationship between parameters b_m, a and ξ is*

$$\frac{b_m}{a} = \frac{\eta_2 h_m^2}{-W^{-1}(\frac{-1}{\exp(1+\frac{\xi}{\varepsilon\gamma_m})}) - 1}, \forall m \in \mathcal{M}_1, \tag{20}$$

where $\gamma_m = 1/v_m + \beta_2\Delta + \beta_1\mathbb{E}/v_m \cdot \mathbb{I}(x'_m = 0)$, $W(x)$ *denotes the Lambert-W function.*

Proof. See Appendix A, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TCC.2022.3140231. □

Then we get the proportional relationship between a and b_m. It helps us find out the value of a and b_m. By taking a simple transformation of Eq. (20), we can obtain b_m

$$b_m = \eta_2 h_m^2 a \cdot \Phi_m(\xi), \tag{21}$$

where $\Phi_m(\xi) \triangleq (-W^{-1}(\frac{-1}{\exp(1+\frac{\xi}{\varepsilon\gamma_m})}) - 1)^{-1}$. It can be seen that $\sum_{m\in\mathcal{M}_1} b_m + a = 1$ holds based on the analysis above. Accordingly, by combining Eqs. (19) and (21), we achieve a closed-form a

$$a = \frac{1}{1+\eta_2\sum_{m\in\mathcal{M}_1} h_m^2\Phi_m(\xi)} \triangleq \Gamma(\xi). \tag{22}$$

We can see that both $\Phi_m(\xi)$ and $\Gamma(\xi)$ are functions about ξ. There is no doubt that the key to find out the value of a and b_m is to find ξ. The following Theorem 2 can help us find ξ.

Theorem 2. The optimal value of ξ should satisfy the following equation:

$$\begin{aligned} \Pi(\xi) \triangleq &\frac{1}{3}(\Gamma(\xi))^{-\frac{2}{3}}\eta_1 \sum_{m\in\mathcal{M}_0} (\frac{h_m}{k_m})^{\frac{1}{3}} \\ &+\gamma_m\varepsilon\eta_2 \sum_{m\in\mathcal{M}_1} \frac{h_m^2}{1+1/\Phi_m(\xi)} - \xi = 0. \end{aligned} \tag{23}$$

Proof. See Appendix B, available in the online supplemental material. □

By solving Eqs. (23), we can get the value of ξ and further get the values of a and $\mathbf{b}$.

5.2 Create Training Set

Based on the analysis above, we can see that $\mathbf{x}$ and $\mathbf{v}$ are crucial in solving $P2$. Hence, we train two deep neural networks that can provide best $\mathbf{x}$ and $\mathbf{v}$ based on $\mathbf{h}$ respectively. When $t = 1$, the parameters of both DNNs are randomly initialized following a zero-mean normal distribution, and the biases of both DNNs are initialized to 0.1. There is no doubt that $\mathbf{x}^{t=1}$ and $\mathbf{v}^{t=1}$ are not good solutions. But, based on the parameters $\mathbf{x}^t$ and $\mathbf{v}^t$, we can obtain more suitable solutions. In the following, we will propose an algorithm to find better $\mathbf{x}^t$ and $\mathbf{v}^t$ base on the wrong ones.

5.2.1 *Find a Better* v

When the user needs to protect its usage pattern privacy, the wireless device m can generate two sizes of redundant data which correspond to two different levels of usage privacy protection. So, we use sigmoid as the activation function in the output layer, i.e., $S(x) = \frac{1}{1+e^{-x}}$, and we can get the output $v_m \in (0,1)$. Consequently, we can divide the results into two categories, i.e., low and high levels of usage pattern privacy preservation. Based on the parameter $\mathbf{v}^t$, i.e., the output of the deep neural network at time slot t, we can create K_1 different alternative optimal $\mathbf{v}_{k1}$, and we will find the best solution from these different $\mathbf{v}_{k1}$. In theory, K_1 meets the constraint $K_1 \leqslant M+1$. The first privacy decision $\mathbf{v}^t$ can be obtained through the $\mathbf{v}^t$

$$v_{1,m} = \begin{cases} m_1 & \text{if} \quad v_m^t > 0.5, \\ m_0 & \text{if} \quad v_m^t \leqslant 0.5, \end{cases}$$
$$for\ m = 1, \ldots, M. \tag{24}$$

m_0 and m_1 represent two different usage pattern privacy protection levels respectively. Then, based on the first decision, the remaining $\mathbf{v}_{k1}$s can be generated though $\mathbf{v}_1$, which is formalized as

$$v_{k1,m} = \begin{cases} m_1 & \text{if} \quad v_m^t > v_{k_1-1}^t, \\ m_1 & \text{if} \quad v_m^t = v_{k_1-1}^t \; and \; v_m^t \leqslant 0.5, \\ m_0 & \text{if} \quad v_m^t = v_{k_1-1}^t \; and \; v_m^t \geqslant 0.5, \\ m_0 & \text{if} \quad v_m^t < v_{k_1-1}^t, \end{cases}$$
$$for \; m = 1, \ldots, M; \; k_1 = 2, \ldots, K_1. \tag{25}$$

For example, suppose that $\mathbf{v}^t = [0.1, 0.3, 0.6, 0.9]$ and $K_1 = 4$. Accordingly, the 4 usage privacy protection actions generated form $\mathbf{v}^t$ are $\mathbf{v}_1 = [m_0, m_0, m_1, m_1]$, $\mathbf{v}_2 = [m_1, m_1, m_1, m_1]$, $\mathbf{v}_3 = [m_0, m_0, m_1, m_1]$, and $\mathbf{v}_4 = [m_0, m_0, m_0, m_1]$. Compared with the exhaustive method which can generates 2^M actions at most, the complexity of our method has been greatly reduced while maintaining the similar performance. The reason is that the distance between $\mathbf{v}^t$ and $\mathbf{v}_{k_1}$ in our method is more large than that of exhaustive method. Thus, we can find a better $\mathbf{v}_{k_1}$ more easily than the exhaustive method.

5.2.2 *Find a Better* $\mathbf{x}$

The activation function of the deep neural network that outputs $\mathbf{x}$ is identical with that of the deep neural network that outputs $\mathbf{v}$. Thus, the method to find an optimal $\mathbf{x}$ is same to $\mathbf{v}$. Therefore, based on the $\mathbf{x}^t$, i.e., the output of our deep neural network at time slot t, we can create K_2 different $\mathbf{x}_{k2}$. We could find the best $\mathbf{x}^*$ from these K_2 solutions.

$$x_{k2,m} = \begin{cases} 1 & \text{if} \quad x_m^t > 0.5 \; and \; k_2 = 1, \\ 0 & \text{if} \quad x_m^t \leqslant 0.5 \; and \; k_2 = 1, \\ 1 & \text{if} \quad x_m^t > x_{k_2-1}^t, \\ 1 & \text{if} \quad x_m^t = x_{k_2-1}^t \; and \; x_m^t \leqslant 0.5, \\ 0 & \text{if} \quad x_m^t = x_{k_2-1}^t \; and \; x_m^t \geqslant 0.5, \\ 0 & \text{if} \quad x_m^t < x_{k_2-1}^t, \end{cases}$$
$$for \; m = 1, \ldots, M; \; k_2 = 2, \ldots, K_2. \tag{26}$$

Generally, a larger K_1 or a larger K_2 is more likely to find a better solution. But, it will also bring more computation difficulty, and vice versa. Hence, we choose large K_1 and K_2 at first to find the best solution among the solutions that we can find. However, after long training period, small K_1 and K_2 are enough to achieve a good computation rate. Therefore, it is desirable to gradually reduce K_1 and K_2 with the progress of training. For example, at the beginning, we set K to be the maximum value of 10, and we will record the value of k which obtain the optimal solution. At each time slot, K is set to be $k_{\max} + 1$, where $k_{\max}$ is the maximum value of k in the previous 50 time slots.

5.2.3 *Create Training Set*

We can achieve $Q(\mathbf{h}, \mathbf{x}, \mathbf{v}, \mathbf{b}, a)$ by solving $P2$ through candidate $\mathbf{x}_{k1}$ and $\mathbf{v}_{k2}$. Then, the $(\mathbf{h}, \mathbf{x}^*, \mathbf{v}^*)$ which maximizes $Q(\mathbf{h}, \mathbf{x}, \mathbf{v}, \mathbf{b}, a)$ will be stored in or be used to update the training set.

5.3 Learning Training Set

In our system, there are two DNNS which are used to learn the distribution of $\mathbf{x}$ and $\mathbf{v}$ respectively with the given $\mathbf{h}$. Once the number of data in the training set is larger than the batch size, the two DNNs will be trained immediately. We train the DNNs with a batch of training data to improve the efficiency of learning. When the training set is full, the newly generated training data replace the previously generated training data. Hence, the DNNs only learn from the latest training data which is better than the old one. As it can be seen that there is almost no difference between the adjacent time slots, because we just add or update one training data at one time slot. It will be very inefficient and useless to train the DNNs every time slot. Therefore, the pseudocode of our algorithm can be expressed as Algorithm 2. The for loop takes the majority computation cost of Algorithm 2, and thus the computation complexity of Algorithm 2 is $O(|\mathcal{T}|)$ at most.

Algorithm 2. Online Deep RL Algorithm to Solve the Offloading Decision Problem

```
Require: Wireless channel gain $\mathbf{h}$
Ensure: $\mathbf{x}, \mathbf{v}, \mathbf{b}, a$ that maximize $Q(\mathbf{h}, \mathbf{x}, \mathbf{v}, \mathbf{b}, a)$
 1: Initialize DNNs' parameters
 2: for t in $\mathcal{T}$ do
 3:     Select right value of $K_1$ and $K_2$ according to $t$
 4:     Select suitable batch size $\epsilon$ and learning interval $\delta$
 5:     Input $\mathbf{h}$ to DNNs to get $\mathbf{x}^t$ and $\mathbf{v}^t$
 6:     Generate spare $\mathbf{x}_{k_1}$ and $\mathbf{v}_{k_2}$
 7:     Compute $Q$ for all $\mathbf{x}_{k_1}$ and $\mathbf{v}_{k_2}$ by solving $P2$
 8:     Select $\left(\mathbf{x}_{k_1}^*, \mathbf{v}_{k_2}^*\right)$ that maximize $Q$
 9:     Use $(\mathbf{h}, \mathbf{x}_{k_1}^*, \mathbf{v}_{k_2}^*)$ to update training set
10:     if $t \geqslant \epsilon$ AND Remainder$(t/\delta) = 0$ then
11:         Randomly choose $\epsilon$ training datas
12:         Train the DNNs with chosen batch of training datas
```

6 Simulation Results

In this section, we evaluate the performance of our proposed task offloading scheme by using tensorflow in python.

6.1 Experimental Setup

Existing Work for Comparisons. We compare our method with the existing work [53] (hereafter *DROO*) which utilized reinforcement learning to investigate the task offloading in MEC without privacy preservation. Moreover, we adapt our algorithm to the neural network, and hereafter we call such method as *Neural Network*. Neural Network preserves both the two kinds of privacy, and it is based on neural network while our work is based on deep reinforcement learning. In addition, we also compare our work with two Baselines, i.e., *Baseline1* and *Baseline2*. Specifically, Baseline1 means that at each time slot, the user device executes computation tasks locally. Likewise, Baseline2 means that the tasks are offloaded to MEC server at each time slot.

Metrics. We mainly use the following metrics to evaluate the performance of our proposed method. Specifically, we first use the metrics, Loss of learning $\mathbf{x}$, Normalized Computation Rate $\widehat{R}$, Normalized $\widehat{Q}$, to evaluate the convergence of our algorithm. Then, we also investigate the offloading

TABLE 2
Simulation Parameter Setting

Parameter	Numerical value	Parameter	Numerical value
M	5	N	10
μ	0.7	k	10^{-26}
P	3W	ϕ	100 cycles/bit
B	2×10^6 Hz	N_0	10^{-10} W
A_d	4.11	f_c	915 MHZ
d_e	2.8 Hz	δ	40

decisions, level of privacy preservation, and computation rate of our proposed method, and further compare our work with the two learning-based methods and two Baselines.

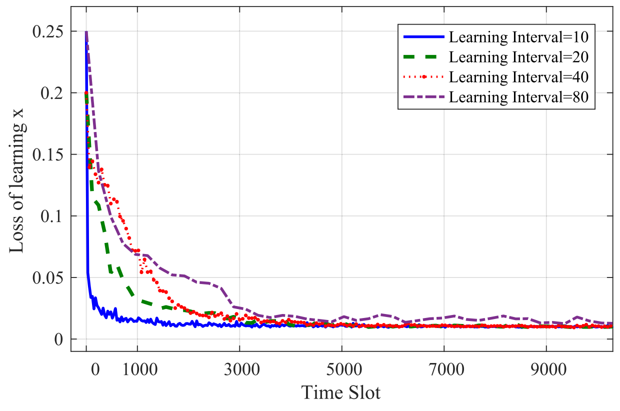
Parameter Setting. We consider ten wireless devices communicating with the MEC server at the same time. The computation tasks generated by these devices can be offloaded to MEC server or be executed locally. There are 30000 time slots in our system, and we set $\delta = 40$, $K = 10$ and $K_2 = 5$ in the default setting. The wireless channel gain $\mathbf{h}$ follows the free space path loss model $\bar{h}_m = A_d(\frac{3\cdot 10^8}{4\pi f_c d_i})^{d_e}$, where $\bar{h}_m$ denotes the average channel gain, $A_d = 4.11$ denotes the antenna gain, $f_c = 915$ MHZ denotes the carrier frequency, and $d_e = 2.8$ denotes the path loss exponent. The wireless channel gain of each device at one time slot is generated from a Rayleigh fading channel model $h_m = \bar{h}_m\lambda_m$, where λ_m is the independent random channel fading factor following an exponential distribution with unit mean. The values of other simulation parameters are listed in the following Table 2.

6.2 Simulation Results

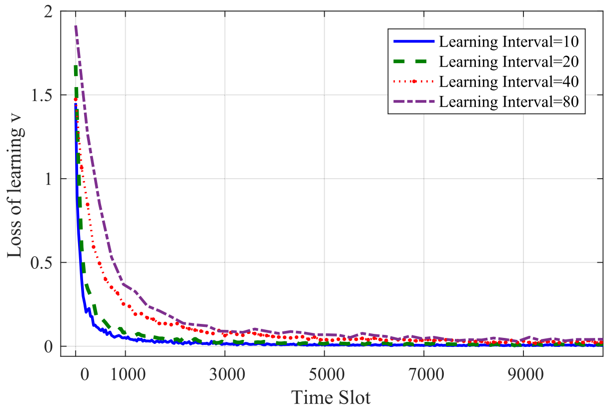
6.2.1 Convergence

We first investigate the learning speed of our DNNs by adjusting the learning interval δ. The so-called learning interval means that our algorithm will be trained once every certain time slot. So, we can choose a certain δ to balance the relationship between the algorithm efficiency and the learning speed, via comparing the learning loss under different δ. In Fig. 4a, with the increase of δ, the decrease of the loss of learning $\mathbf{v}$ slow down gradually. Likewise, in Fig. 4b, the loss of learning $\mathbf{x}$ decreases with the increasing δ. When $\delta = 10$, the loss of learning $\mathbf{v}$ and $\mathbf{x}$ decrease to 0 quickly, but the operation efficiency of our algorithm will be at a low standard. When $\delta = 80$, the loss of learning $\mathbf{v}$ and $\mathbf{x}$ decrease slowly, and the loss of learning $\mathbf{v}$ can not be reduced to 0 at about 9000 time slot. But the operation efficiency of our algorithm can be improved at present. So, choosing a medium value for δ, e.g, 40, is a good idea. Thus, we will set δ to 40, set the learning rate $\mathbf{v}$ to 0.1, and the learning rate $\mathbf{x}$ to 0.02 in the default settings. It is important to noted that we set the weights of privacy ω_1 and ω_2 to 50% to balance the location privacy and the usage pattern privacy.

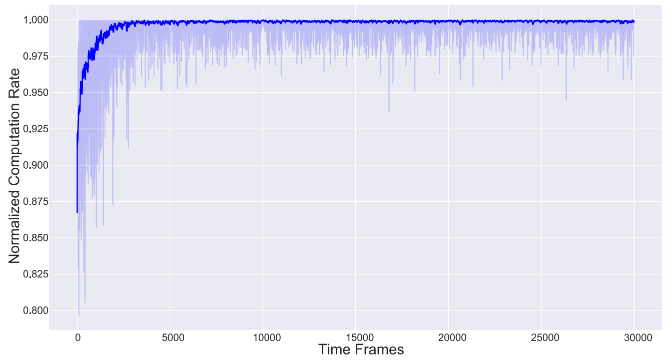
In different scenarios, the wireless channel gains will alter greatly, which can have a great effect on the results of the whole computation rate R. So, it is wise to introduce $\widehat{R} = \frac{R}{\max R}$ to make the results easier observed, where the best solution $\max R$ can be obtained by the exhaustive algorithm in advance. As depicted in Fig. 4c, with the progress of the



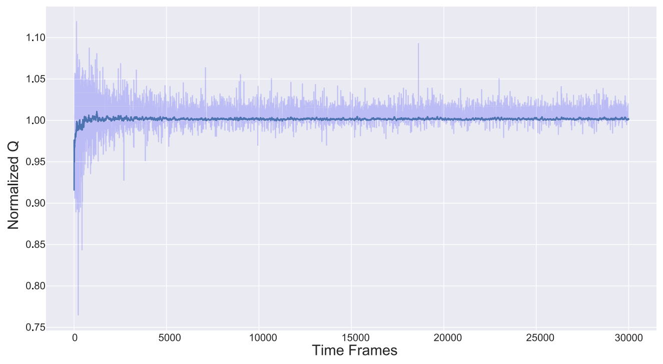
(a) Loss of learning $\mathbf{x}$ versus δ.



(b) Loss of Learning $\mathbf{v}$ versus δ.

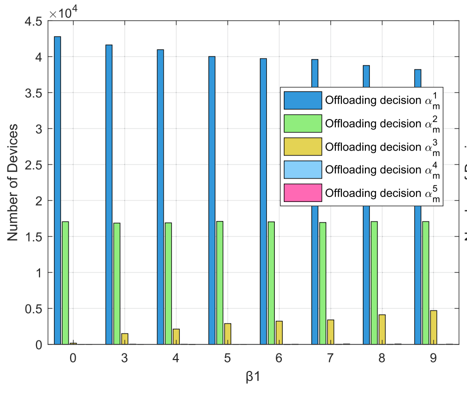


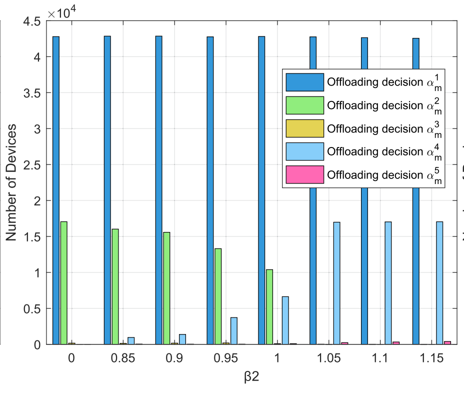
(c) Normalized Computation Rate $\widehat{R}$ versus t.



(d) Normalized $\widehat{Q}$ versus t.

Fig. 4. The convergence of our method in terms of loss of learning $\mathbf{x}$, normalized Computation Rate $\widehat{R}$, and normalized weighted sum of computation rate and privacy level $\widehat{Q}$.


(a) Impact of privacy weight $\beta 1$, $\beta 2$.


(b) Impact of parameter ϕ.

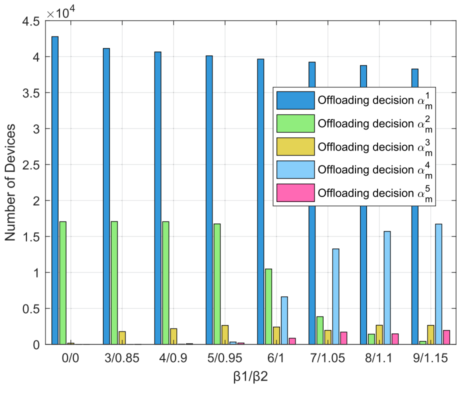

(c) Impact of parameter ϕ.

Fig. 5. The impact of privacy weight $\beta 1$, $\beta 2$ and needed cycles for processing one bit data ϕ on the offloading decisions.

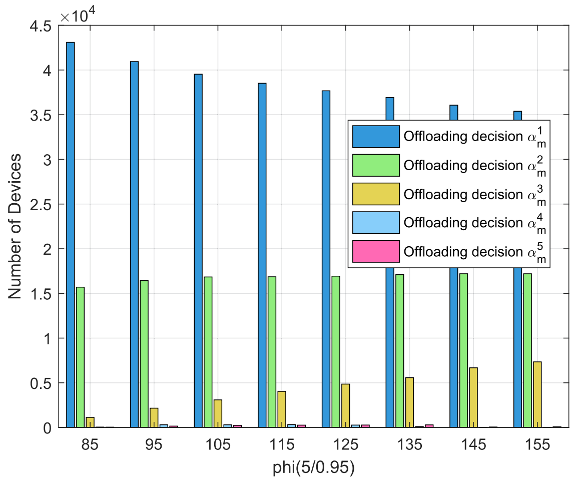
learning of DNNs, the average normalized $\widehat{R}$ tends to 1 gradually. We can observe that $\widehat{R}$ is higher than 95% after about 3000 time slots. In addition, the light purple shadow in Fig. 4c denotes the range of $\widehat{R}$ over the past 50 time slots. With the time slot moving on, the fluctuation of purple shadow decreases gradually. To better observe the variation of Q, we introduce $\widehat{Q} = \frac{Q}{\max R}$. As it can be seen in Fig. 4d, the value of $\widehat{Q}$ fluctuates greatly at first to balance computation rate and the weighted privacy. But, with the processing of our deep reinforcement learning, we find the balance between computation rate and the weighted privacy. As a result, the swing of the $\widehat{Q}$ stabilized gradually and keep slightly greater than 1. It proves that our algorithm can obtain as much computation rate as possible while protecting the privacy of wireless devices.

6.2.2 Offloading Decisions

Fig. 5 shows the impact of privacy weight β_1/β_2 and the need cycles of one bit ϕ on the offloading decisions α_m^1, α_m^2, α_m^3, α_m^4, α_m^5. As it can be seen in Fig. 5a, in the last 6000 time slots of test set, the number of devices that choose the offloading decision α_m^2 has not changed a lot with the varying weights. But, with the enlarging weight of location privacy, the proportion of devices that choose the offloading decision α_m^1 decreases and devices that choose the offloading decision α_m^3 increases gradually. In Fig. 5b, with the enlarging weight of usage pattern privacy, the proportion of devices that choose the offloading decision α_m^2 decreases and devices that choose the offloading decision α_m^4 increases gradually. In Fig. 5c, with the enlarging weights of the two kind of privacy, the number of devices that choose the offloading decision α_m^1, α_m^2 decreases and devices that choose the offloading decision α_m^3, α_m^4, α_m^5 increases gradually. This suggests that our algorithm can adjust the devices' offloading decisions to maximize $\widehat{Q}$, with respect to users' different demands of privacy preservation. In addition, as depicted in Figs. 6a and 6b, with the increase of the needed CPU cycles for processing one bit, less devices choose to execute locally, and more devices choose to offload tasks. But, when task offloaded, the more CPU cycles required, more devices will choose to offload without privacy protection.

6.2.3 Level of Privacy Preservation

We change the value of β_1, β_2 or ϕ and keep other factors fixed to evaluate the level of privacy preservation of our algorithm. Since literature [53] only concentrated on the computation rate without privacy, we compare our work with Neural Network, Baseline1, Baseline2. As it can be


(a) Impact of parameter ϕ.

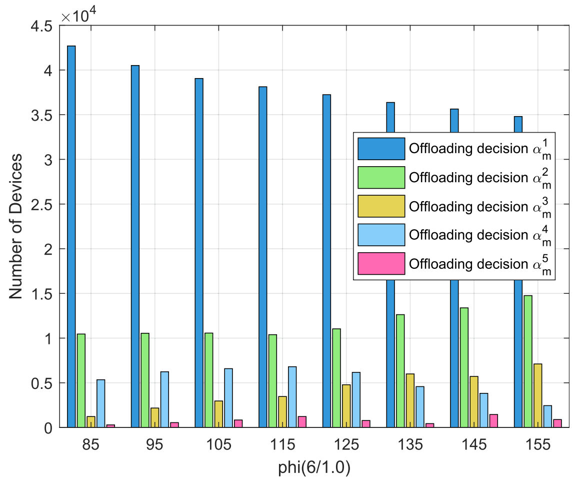

(b) Impact of parameter ϕ.

Fig. 6. The impact of privacy weight $\beta 1$, $\beta 2$ and needed cycles for processing one bit data ϕ on the offloading decisions.

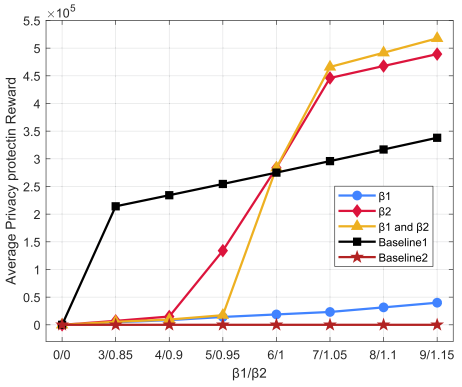


Fig. 7. The impact of parameter $\beta1/\beta2$ on average privacy protection reward.

seen in Fig. 7, the average privacy protection reward of the test set increases with the increment of the value of β_1, β_2 or β_1 and β_2. With the change of the value of β_1, the average privacy protection reward has little change. However, with the change of the value of β_2, the reward changes a lot, which means that usage pattern privacy has a greater effect on the average privacy protection reward. Obviously, increasing the value of β_1 and β_2 at the same time can get the maximum average privacy protection reward. As it can be seen in Fig. 8, in the case of $\beta_1 = 6$, $\beta_2 = 1.0$, the average privacy protection reward of the test set increases with the increment of the required CPU cycles ϕ for executing one bit data at first. However, when the needed cycle ϕ continues to increase, the average privacy protection reward decreases and gradually close to line baseline2. The reasons for this phenomenon is that with the increases of the privacy weights, our strategy tends to offload tasks, and with the increases of local computation energy, our strategy tends to offload tasks too, which makes the average privacy protection reward more and more close to that of baseline2. In the case of $\beta_1 = 5$, $\beta_2 = 0.95$, a low privacy weight case, the average privacy protection reward increases continuously in the limited range of ϕ and gradually close to line baseline2 too. That is because tasks are more likely to be executed locally in a low privacy weight. These experimental results show that our algorithm can adjust the offloading decisions to adapt to the different privacy and computing capabilities. Both our algorithm and Neural Network have the same trend. But more importantly, average privacy protection reward in our algorithm is larger. Moreover, the average privacy protection reward in Baseline1 and Baseline2 is not affected by the increasing needed cycles ϕ. That is owing to that the two methods do not update offloading decisions with the increasing needed cycles ϕ.

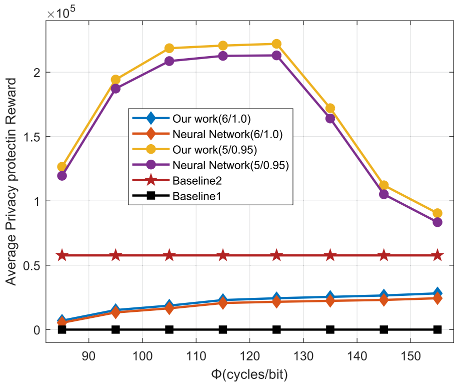


Fig. 8. The impact of ϕ on the normalized weighted sum of computation rate and privacy level $\widehat{Q}$.

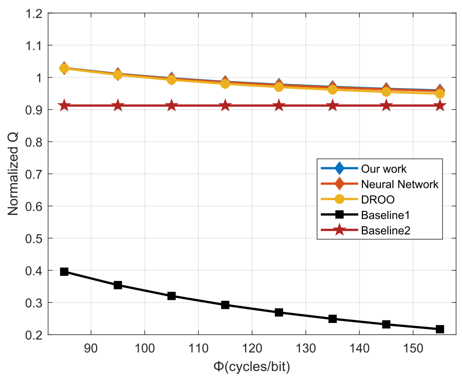


Fig. 9. The impact of ϕ on average privacy protection reward.

6.2.4 Computation Rate

We investigate the impact of the required CPU cycles ϕ for executing one bit data. We compare our work with Neural Network, Baseline1, Baseline2, and DROO. As depicted in Fig. 9, the normalized weighted sum of computation rate and privacy level $\widehat{Q}$ in these five algorithms decreases with the increasing ϕ. Moreover, it can be seen that the normalized $\widehat{Q}$ in our algorithm, Neural Network, and DROO is more robust to the increasing ϕ. In addition, the normalized $\widehat{Q}$ in our algorithm is larger than that in the other algorithms, which means that our algorithm has better performance in joint optimization of privacy protection and computation rate. It is worth noting that the $\max R$ in $\widehat{Q} = \frac{Q}{\max R}$ is obtained when ϕ equals to 100. In addition, it can be seen that $\widehat{Q}$ in Baseline1 decreases continuously, and $\widehat{Q}$ in Baseline2 is not affected by ϕ. The reasons are that Baseline1 is only related to the local computing power, and that Baseline2 is not constrained by the local computing power.

6.2.5 Computation Delay

In our deep reinforcement learning, the structure of neural network is $[10, 120, 80, 10]$, and the number of training sets is 30000. We train with 10 training sets as a batch. The computation delay of our work is 7.5 ms, 9.6 ms, 11.8 ms, 14 ms, 15 ms, 15.5 ms, 14.9 ms, and 14.5 ms respectively when parameter Φ is 85 cycles/bit, 95 cycles/bit, 105 cycles/bit, 115 cycles/bit, 125 cycles/bit, 135 cycles/bit, 145 cycles/bit, and 155 cycles/bit. We can see that the computation delay is linearly increasing with respect to the increment of parameter Φ at first, and then is little affected by the increasing Φ. The reasons are that the computing capacity of edge server is powerful, and that the computation delay is mainly affected by the amount of the tasks.

7 Conclusion

In this paper, we investigated the tradeoff of computation rate and privacy preservation for task offloading in a multi-user MEC system. We first formalize the task offloading as an optimization problem of computation rate and privacy preservation. Then, we design a deep reinforcement learning based offloading algorithm to solve such an non-convex problem, aiming to obtain the better tradeoff between the computation rate and the privacy preservation. Finally, extensive simulation results demonstrate that our algorithm can maintain a high level of computation rate while protecting users' privacy, compared with existing work and Baselines.

Note that the untrusted MEC server may have other kinds of background information, e.g., social relationship, home address, and so on, which may enable the untrusted MEC server to infer a special user's more sensitive information. What's more, based on these side information, there may be other kinds of attacks. So in terms of the future work, we plan to explore the privacy-preserving offloading schemes against the untrusted MEC server with more kinds of background information, and investigate other kinds of attacks in task offloading.

References

[1] F. Sangoleye, N. Irtija, and E. E. Tsiropoulou, "Data acquisition in social Internet of Things based on contract theory," in *Proc. IEEE Int. Conf. Commun.*, 2021, pp. 1–6.

[2] N. Patrizi, E. E. Tsiropoulou, and S. Papavassiliou, "Health data acquisition from wearable devices during a pandemic: A techno-economics approach," in *Proc. IEEE Int. Conf. Commun.*, 2021, pp. 1–6.

[3] M. Min, X. Wan, and L. Xiao, "Learning-based privacy-aware offloading for healthcare iot with energy harvesting," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4307–4316, Jun. 2019.

[4] Y. Bai, L. Chen, L. Song, and J. Xu, "Risk-aware edge computation offloading using Bayesian stackelberg game," *IEEE Trans. Netw. Service Manage.*, vol. 17, no. 2, pp. 1000–1012, Jun. 2020.

[5] S. Mao, J. Wu, L. Liu, D. Lan, and A. Taherkordi, "Energy-efficient cooperative communication and computation for wireless powered mobile-edge computing," *IEEE Syst. J.*, to be published, doi: 10.1109/JSYST.2020.3020474.

[6] L. Qian, W. Wu, W. Lu, Y. Wu, B. Lin, and T. Q. S. Quek, "Secrecy-based energy-efficient mobile edge computing via cooperative non-orthogonal multiple access transmission," *IEEE Trans. Commun.*, vol. 69, no. 7, pp. 4659–4677, Jul. 2021.

[7] T. Wang, Y. Li, and Y. Wu, "Energy-efficient UAV assisted secure relay transmission via cooperative computation offloading," *IEEE Trans. Green Commun. Netw.*, vol. 5, no. 4, pp. 1669–1683, Dec. 2021.

[8] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. S. Quek, "Offloading in mobile edge computing: Task allocation and computational frequency scaling," *IEEE Trans. Commun.*, vol. 65, no. 8, pp. 3571–3584, Aug. 2017.

[9] G. Zhang, W. Zhang, Y. Cao, D. Li, and L. Wang, "Energy-delay tradeoff for dynamic offloading in mobile-edge computing system with energy harvesting devices," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4642–4655, Oct. 2018.

[10] A. Hekmati, P. Teymoori, T. D. Todd, D. Zhao, and G. Karakostas, "Optimal mobile computation offloading with hard deadline constraints," *IEEE Trans. Mobile Comput.*, vol. 19, no. 9, pp. 2160–2173, Sep. 2020.

[11] H. Ko and S. Pack, "Distributed device-to-device offloading system: Design and performance optimization," *IEEE Trans. Mobile Comput.*, vol. 20, no. 10, pp. 2949–2960, Oct. 2021.

[12] X. He, R. Jin, and H. Dai, "Physical-layer assisted secure offloading in mobile-edge computing," *IEEE Trans. Wireless Commun.*, vol. 19, no. 6, pp. 4054–4066, Jun. 2020.

[13] K. Guo, M. Sheng, and J. Tang, "Hierarchical offloading for delay-constrained applications in fog RAN," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4257–4270, Apr. 2020.

[14] Y. Li, X. Wang, X. Gan, H. Jin, L. Fu, and X. Wang, "Learning-aided computation offloading for trusted collaborative mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 19, no. 12, pp. 2833–2849, Dec. 2020.

[15] C. Yi, J. Cai, and Z. Su, "A multi-user mobile computation offloading and transmission scheduling mechanism for delay-sensitive applications," *IEEE Trans. Mobile Comput.*, vol. 19, no. 1, pp. 29–43, Jan. 2020.

[16] J. Yan, S. Bi, and Y. J. Zhang, "Optimal task offloading and resource allocation in mobile-edge computing with inter-user task dependency," *IEEE Trans. Wireless Commun.*, vol. 19, no. 1, pp. 235–250, Jan. 2020.

[17] Y. Ding, C. Liu, X. Zhou, Z. Liu, and Z. Tang, "A code-oriented partitioning computation offloading strategy for multiple users and multiple mobile edge computing servers," *IEEE Trans. Ind. Informat.*, vol. 16, no. 7, pp. 4800–4810, Jul. 2020.

[18] H. Wei, H. Luo, Y. Sun, and M. S. Obaidat, "Cache-aware computation offloading in IoT systems," *IEEE Syst. J.*, vol. 14, no. 1, pp. 61–72, Mar. 2020.

[19] W. Zhan, C. Luo, G. Min, C. Wang, Q. Zhu, and H. Duan, "Mobility-aware multi-user offloading optimization for mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 3341–3356, Mar. 2020.

[20] C. Shu, Z. Zhao, Y. Han, G. Min, and H. Duan, "Multi-user offloading for edge computing networks: A dependency-aware and latency-optimal approach," *IEEE Internet Things J.*, vol. 7, no. 3, pp. 1678–1689, Mar. 2020.

[21] J. Ren, K. M. Mahfujul, and F. Lyu, "Joint channel allocation and resource management for stochastic computation offloading in MEC," *IEEE Trans. Veh. Technol.*, vol. 69, no. 8, pp. 8900–8913, Aug. 2020.

[22] S. Yi, Z. Qin, and Q. Li, "Security and privacy issues of fog computing: A survey," in *Proc. Int. Conf. Wireless Algorithms Syst. Appl.*, 2015, pp. 1–10.

[23] J. Ni, K. Zhang, X. Lin, and X. Shen, "Securing fog computing for Internet of Things applications: Challenges and solutions," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 1, pp. 601–628, First Quarter 2018.

[24] T. He, E. N. Ciftcioglu, S. Wang, and K. S. Chan, "Location privacy in mobile edge clouds," in *Proc. IEEE 37th Int. Conf. Distrib. Comput. Syst.*, 2017, pp. 2264–2269.

[25] T. He, E. N. Ciftcioglu, and S. Wang, "Location privacy in mobile edge clouds: A chaff-based approach," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 11, pp. 2625–2636, Nov. 2017.

[26] M. Du, K. Wang, Y. Chen, X. Wang, and Y. Sun, "Big data privacy preserving in multi-access edge computing for heterogeneous Internet of Things," *IEEE Commun. Mag.*, vol. 56, no. 8, pp. 62–67, Aug. 2018.

[27] X. Li, S. Liu, F. Wu, S. Kumari, and J. J. P. C. Rodrigues, "Privacy preserving data aggregation scheme for mobile edge computing assisted IoT applications," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4755–4763, Jun. 2019.

[28] L. Li, M. Siew, and T. Q. S. Quek, "Learning-based pricing for privacy-preserving job offloading in mobile edge computing," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2019, pp. 4784–4788.

[29] M. Qiucheng, J. Weipeng, and S. Houbing, "Differential privacy-based location privacy enhancing in edge computing," *Concurrency Comput. Pract. Experience*, vol. 38, no. 7, 2018, Art. no. e4735.

[30] P. Zhao, H. Huang, X. Zhao, and D. Huang, "P3: Privacy-preserving scheme against poisoning attacks in mobile-edge computing," *IEEE Trans. Comput. Soc. Syst.*, vol. 7, no. 3, pp. 818–826, Jun. 2020.

[31] J. Liu, K. Kumar, and Y. Lu, "Tradeoff between energy savings and privacy protection in computation offloading," in *Proc. ACM/IEEE Int. Symp. Low-Power Electron. Des.*, 2010, pp. 213–218.

[32] J. Liu and Y. H. Lu, "Energy savings in privacy-preserving computation offloading with protection by homomorphic encryption," in *Proc. Int. Conf. Power Aware Comput. Syst.*, 2010, pp. 1–7.

[33] F. Wang, Y. Xu, L. Zhu, X. Du, and M. Guizani, "LAMANCO: A lightweight anonymous mutual authentication scheme for -times computing offloading in IoT," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4462–4471, Jun. 2019.

[34] S. Chen, X. Zhu, H. Zhang, C. Zhao, G. Yang, and K. Wang, "Efficient privacy preserving data collection and computation offloading for fog-assisted IoT," *IEEE Trans. Sustain. Comput.*, vol. 5, no. 4, pp. 526–540, Fourth Quarter 2020.

[35] X. He, R. Jin, and H. Dai, "Peace: Privacy-preserving and cost-efficient task offloading for mobile-edge computing," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 1814–1824, Mar. 2020.

[36] X. He and R. Jin, "Physical-layer assisted privacy-preserving offloading in mobile-edge computing," in *Proc. IEEE Int. Conf. Commun.*, 2019, pp. 1–6.

[37] X. He, J. Liu, R. Jin, and H. Dai, "Privacy-aware offloading in mobile-edge computing," in *Proc. IEEE Global Commun. Conf.*, 2017, pp. 1–6.

[38] X. He, R. Jin, and H. Dai, "Deep PDS-learning for privacy-aware offloading in MEC-enabled IoT," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4547–4555, Jun. 2019.

[39] P. A. Apostolopoulos, E. E. Tsiropoulou, and S. Papavassiliou, "Risk-aware data offloading in multi-server multi-access edge computing environment," *IEEE/ACM Trans. Netw.*, vol. 28, no. 3, pp. 1405–1418, Jun. 2020.

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

[41] W. Yahya, E. Oki, Y.-D. Lin, and Y.-C. Lai, "Scaling and offloading optimization in pre-cord and post-cord multi-access edge computing," *IEEE Trans. Netw. Service Manage.*, vol. 18, no. 4, pp. 4503–4516, Dec. 2021.

[42] J. Zhang, W. Shi, R. Zhang, and W. Liu, "Computation offloading and shunting scheme in wireless wireline internetwork," *IEEE Trans. Commun.*, vol. 69, no. 10, pp. 6808–6821, Oct. 2021.

[43] Y. Sun, N. Li, and X. Tao, "Privacy preserved secure offloading in the multi-access edge computing network," in *Proc. IEEE Wireless Commun. Netw. Conf. Workshops*, 2021, pp. 1–6.

[44] Z. Xue *et al.*, "A resource-constrained and privacy-preserving edge-computing-enabled clinical decision system: A federated reinforcement learning approach," *IEEE Internet Things J.*, vol. 8, no. 11, pp. 9122–9138, Jun. 2021.

[45] G. Zhang, S. Ni, and P. Zhao, "Learning-based joint optimization of energy-delay and privacy in multiple-user edge-cloud collaboration MEC systems," *IEEE Internet Things J.*, vol. 9, no. 2, pp. 1491–1502, Jan. 2022.

[46] C. Yi, J. Cai, and Z. Su, "A multi-user mobile computation offloading and transmission scheduling mechanism for delay-sensitive applications," *IEEE Trans. Mobile Comput.*, vol. 19, no. 1, pp. 29–43, Jan. 2020.

[47] P. Zhao, H. Jiang, and J. C. S. Lui, "P3-LOC: A privacy-preserving paradigm-driven framework for indoor localization," *IEEE/ACM Trans. Netw.*, vol. 26, no. 6, pp. 2856–2869, Dec. 2018.

[48] H. Jiang, J. Li, P. Zhao, F. Zeng, Z. Xiao, and A. Iyengar, "Location privacy-preserving mechanisms in location-based services: A comprehensive survey," *ACM Comput. Surv.*, vol. 54, no. 1, pp. 1–36, 2021.

[49] D. Liu, Z. Cao, M. Hou, H. Rong, and H. Jiang, "Pushing the limits of transmission concurrency for low power wireless networks," *ACM Trans. Sensor Netw.*, vol. 16, no. 4, pp. 1–29, 2020.

[50] Z. Xiao *et al.*, "Vehicular task offloading via heat-aware MEC cooperation using game-theoretic method," *IEEE Internet Things J.*, vol. 7, no. 3, pp. 2038–2052, Mar. 2020.

[51] H. Jiang *et al.*, "Fly-navi: A novel indoor navigation system with on-the-fly map generation," *IEEE Trans. Mobile Comput.*, vol. 20, no. 9, pp. 2820–2834, Sep. 2021.

[52] H. Jiang, M. Wang, P. Zhao, Z. Xiao, and S. Dustdar, "A utility-aware general framework with quantifiable privacy preservation for destination prediction in lbss," *IEEE/ACM Trans. Netw.*, vol. 29, no. 5, pp. 2228–2241, Oct. 2021.

[53] L. Huang, S. Bi, and Y. J. A. Zhang, "Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks," *IEEE Trans. Mobile Comput.*, vol. 19, no. 11, pp. 2581–2593, Nov. 2020.