DQN-based QoE Enhancement for Data Collection in Heterogeneous IoT Network

Hansong Zhou*, Sihan Yu†, Xiaonan Zhang*, Linke Guo† and Beatriz Lorenzo‡
*Department of Computer Science, Florida State University, Tallahassee, FL 32306, USA
†Department of Electrical and Computer Engineering, Clemson University, Clemson, SC 29634, USA
‡Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, Amherst, MA 15221, USA
Email: hz21e@my.fsu.edu, sihany@g.clemson.edu, xzhang@cs.fsu.edu, linkeg@clemson.edu, blorenzo@umass.edu

2022 IEEE 19th International Conference on Mobile Ad Hoc and Smart Systems (MASS) | 978-1-6654-7180-0/22/$31.00 ©2022 IEEE | DOI: 10.1109/MASS56207.2022.00032

***Abstract*—Sensing data collection from the Internet of Things (IoT) devices lays the foundation to support massive IoT applications, such as patient monitoring in smart health and intelligent control in smart manufacturing. Unfortunately, the heterogeneity of IoT devices and dynamic environments result in not only the life-cycle latency but also data collection failures, affecting the quality of experience (QoE) for all the users. In this paper, we propose a recovery mechanism with a dynamic data contamination method to handle the failure. To further enhance the long-term overall QoE, we allocate the spectrum resources and make contamination decisions for each device using a deep reinforcement learning method. Particularly, a lightweight decentralized State-sharing Deep-Recurrent Q-Network (SDRQN) is proposed to find the optimal collection policies. Our simulation results indicate that the recurrent unit in SDRQN gives rise to 10% lower waiting time and 60% lower task drop rate than the fully-connected design. Compared to a centralized DQN scheme, SDRQN achieves a similar ultra-low drop rate of 0.29% but requires only 1% GPU memory, demonstrating the effectiveness of SDRQN in the large-scale heterogeneous IoT network.**

***Index Terms*—Heterogeneous IoT Network, Data Collection, Quality of Experience, Recovery Mechanism, Deep-Q Network**

I. Introduction

Internet of Things (IoT) is expected to reshape the way we interact with the world, as evidenced by innovations such as wearable computing in healthcare, intelligent controls of home appliances, and automatic irrigation in smart farming [1]. As the foundation of supporting various IoT applications, different types of sensing data are collected from various IoT devices and processed by IoT gateways. As an example of smart health applications, the smartphone plays the role of an IoT gateway to process different health parameters of patients' vital signs collected by various wearable devices. It then sends control messages such as the scheduling and the configurations to these devices [2].

Ensuring all the data is sent to IoT gateways with a bearable latency is an essential ingredient in the life-cycle data collection that is satisfied by users. Existing works rely on several assumptions, including homogeneous network structure and static wireless channel [3], [4]. In practice, large-scale heterogeneous IoT devices with different protocols are deployed for the above emerging IoT applications. Coexisting in a small space, those devices are likely to incur severe interference, causing a long life-cycle latency and a high possibility of data collection failures. Existing approaches [5]–[7] achieve a lower latency roughly from the conventional Quality of Service (QoS) perspective. However, they make few efforts to address the failure problem. Taking into account the failure that occurs in data collection, we introduce Quality of Experience (QoE), defined as the degree of delight or annoyance of the user of an application [8], for evaluating the data collection efficiency in our paper.

Aiming at enhancing the QoE for all the users, in the paper, we propose to employ a recovery mechanism along with the dynamic data contamination to design a reliable data collection scheme. The major challenge lies in that: it is hard to construct a mathematical model of task recovery with the conditions of the dynamics in wireless environments together with the heterogeneity in IoT devices and sensing data. To address this issue, we deploy the deep reinforcement learning (DRL) approach to maximize the long-term overall QoE for all users. Specifically, the Deep Q-Network (DQN) is adopted to find the optimal decisions on whether to collect the data, batch size, and spectrum allocation. To avoid the large searching space in a centralized DQN, we design a decentralized DQN scheme allowing multiple IoT gateways to make decisions for their managed IoT devices. Our contribution is as follows:

- We propose a data recovery mechanism with the dynamic data contamination in order to mitigate the collection failure as well as decrease the life-cycle latency.
- We design a light-weight decentralized **S**tate-sharing **D**eep-**R**ecurrent **Q**-**N**etwork (SDRQN) to learn the data collection policy, including the spectrum resource allocation and the contamination decision, without any prior knowledge of environment dynamics.
- Our experimental results demonstrate that compared to several benchmarks, SDRQN enhances the long-term QoE by significantly reducing the drop rate and average waiting time for all the users.

The rest of our paper is organized as follows The related works are briefly reviewed in Section II. In Section III, we describe the system model of the data collection. Our proposed task recovery mechanism as well as the QoE optimization problem are explained in Section IV. A lightweight SDRQN

The work of L. Guo was supported by National Science Foundation under grant IIS-1949640 and CNS-2008049. The work of B. Lorenzo was supported by National Science Foundation under grant CNS-2008309.

2155-6814/22/$31.00 ©2022 IEEE
DOI 10.1109/MASS56207.2022.00032

is then used in Section V. We evaluate the performance of our proposed data collection scheme in section VI, followed by the conclusion in Section VII.

II. Related Work

Interference Mitigation in Heterogeneous IoT Network. Effectively mitigating the interference is of significant importance in improving transmission reliability as well as decreasing transmission latency in a heterogeneous IoT network. Conventional schemes usually leverage either MAC layer protocols such as CSMA/CA or energy detection methods to avoid interference from a different wireless protocol [9]. Unfortunately, these schemes will be less effective when there is a dense device deployment. Recent efforts in Cross-Technology Communication (CTC) [10]–[12] enable direct communication among different protocols, which effectively mitigates interference. However, CTC usually replaces original spectrum bands with another one, which exacerbates the spectrum scarcity issue. Different from them, we dynamically allocate the spectrum resources to IoT devices based on the current environment and the task arrival status, for which the interference is alleviated. The transmission reliability and latency are further improved.

Enhancement on QoE. Existing works focus on maximizing the QoE for users with a proper resource allocation scheme in wireless network. Among different application scenarios, QoE evaluation and its enhancement methods vary as well. In hierarchical edge-cloud computing [13], they define the QoE as the amount of cost reduction achieved by the users when offloading the task. A suboptimal resource allocation mechanism is then proposed to obtain the Nash Equilibrium to enhance the QoE. Mean opinion score (MOS) is a common QoE evaluation in video stream [14]. He et al. [15] propose the shortest path tree (SPT) algorithm and a DQN-based algorithm to maximize the MOS. For the edge computing network with the queueing system [16], round trip workload transmission time and queueing delay at the edge layer are two major metrics of QoE. In our work, we specifically consider the QoE during the data collection, in which not only the life-cycle latency but also the potential data collection failures are taken into account.

Resource Allocation Methods in IoT Network. The increasing number of devices in the IoT network challenges the design of an optimal resource allocation scheme to improve the wireless system performance. Lu et al. [17] propose a cooperative spectrum sharing method to guarantee the target rate of the primary system in cognitive IoT networks. Yang et al. [18] increase the spectrum efficiency of the fog IoT network with the multi-armed bandit algorithm. In the NOMA (Non-orthogonal multiple access)-enabled IoT network, a Karush-Kuhn-Tucker (KKT) conditions-based spectrum resource allocation scheme is adopted [19] to maximize the spectrum efficiency. DRL-based resource allocation scheme [20] [21] is trending in recent research. Shi et al. [22] studies the problem of spectrum resource sharing in the Industry IoT (IIoT) system with multiple IoT sensors and gives a DRL-based solution. In our paper, we adopt the DQN-based algorithm to address the data collection failures and its long life-cycle latency in the heterogeneous IoT network.

III. System Model

Our system model is shown in Fig. 1, where a set of IoT devices $\mathcal{M} = \{1, 2, \cdots, M\}$ collect sensing data. They operate under K different wireless protocols. Each device is denoted as m_k. The sensing data are then sent to J multi-protocol IoT gateways (MPGs) over the air. Each MPG j manages $\mathcal{M}_j$ IoT devices where $\sum_j \mathcal{M}_j = \mathcal{M}$. These devices share the spectrum with a total bandwidth of B, which is divided into N channel units with a unit bandwidth of B_{min}. Suppose that device m_k uses N_{m_k} channel units for transmission. The total channel bandwidth occupied by m_k is $B_{m_k} = N_{m_k} B_{min}$. We focus on a stationary environment in a quasi-static scenario [23] where the channel is static in a time slot but dynamic through a long-term period. The spectrum efficiency of IoT device m_k is given as follows

$$\eta_{m_k j} = \log_2\{1 + \frac{\alpha_{m_k j} P_k \sum_{n \in B_{m_k}} |h_{m_k n j}|^2}{I_{m_k} + \mathcal{B}_{m_k} N_o}\}, \quad (1)$$

where $I_{m_k} = \sum_{m_i \in \mathcal{M}_{sub,k}} P_i \sum_{n' \in \mathcal{B}_{m_k}} \beta_{m_i n'} |h_{m_i n' j}|^2$ represents the interference from these devices. $\mathcal{M}_{sub,k}$ is a set of IoT devices working on the channel units overlapped with the IoT device m_k. We use a binary symbol $\beta_{m_i n'} = 1$ to indicate that the channel unit $\{n' | n' \in \mathcal{B}_{m_k}\}$ is shared with the device m_i, $m_i \in \mathcal{M}_{sub,k}$. Otherwise, $\beta_{m_i n'} = 0$. $\alpha_{m_k j} = 1$ denotes that device m_k is sending data to MPG j. Otherwise, $\alpha_{m_k j} = 0$. N_o is the Additive white Gaussian noise.

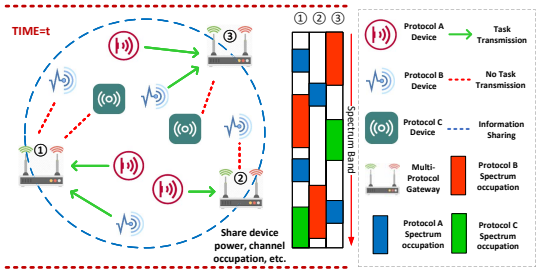


Fig. 1: Data collection model in heterogeneous IoT network.

At the start of a time slot, each IoT device continuously transmits the pilot signal to all MPGs through all available channel units before sending the data. The MPG estimates the channel gain [24] and then generates a channel gain table $\mathbf{H}_j$ with the size of $M \times N$, which is

$$\mathbf{H}_j = \begin{bmatrix} h_{11j} & h_{12j} & \dots & h_{1Nj} \\ h_{21j} & h_{22j} & \dots & h_{2Nj} \\ \dots & \dots & \dots & \dots \\ h_{M1j} & h_{M2j} & \dots & h_{MNj} \end{bmatrix}. \quad (2)$$

Assume that $L^i_{m_k}$ refers to the i-th data sensed by the IoT device m_k. It follows the Poisson process with the arrival rate λ, denoted as $L^i_{m_k} \sim Poisson(\lambda)$. Each IoT device m_k keeps a queue Q_{m_k} with max length V_{m_k} for storing the data in a

FIFO (First-In-First-Out) manner. Assume that ν_{m_k} tasks are concatenated as a batch in each transmission. The size of the entire batch is denoted as $L_{m_k} = \sum_{i=1}^{\nu_{m_k}} L_{m_k}^i$.

IV. QoE Maximization

A. Factors Affecting QoE Performance

1) Life-Cycle Latency: The life-cycle latency $\tau_{m_k}^g$ for the IoT device m_k contains the transmission latency $\tau_{m_k}^{tr}$, processing latency $\tau_{m_k}^{rmt}$, and feedback latency τ^{fd} as follows

$$\begin{aligned}\tau_{m_k}^g &= \tau_{m_k}^{tr} + \tau_{m_k}^{rmt} + \tau^{fd} \\ &= L_{m_k} R_{m_k}^{-1} + \sum_{i=1}^{\nu_{m_k}} \zeta_{u_{m_k}^i} L_{m_k}^i \frac{\rho_m}{f_{m_k}^g} + \tau^{fd},\end{aligned} \tag{3}$$

where $R_{m_k j} = B_{m_k} \eta_{m_k j}$ denotes the data transmission rate; the binary symbol $\zeta_{u_{m_k}^i}$ indicates whether the data is successfully processed by MPG; ρ_m is the processing density and is same for all MPGs; $f_{m_k}^g$ is the computation resource allocated to the current data batch, which is proportional to the batch size L_{m_k}. Because of the high transmit power of the MPG, τ^{fd} is ignored in computing the latency.

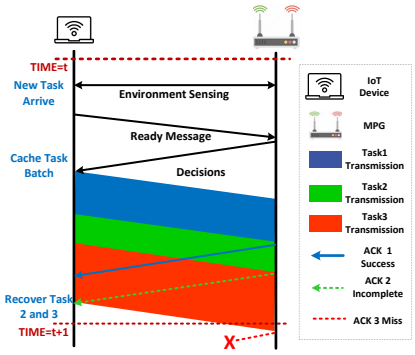


Fig. 2: Handshake procedure for data collection

2) Task Failure: The handshake procedure for each time slot is shown in Fig. 2. The Ready message from IoT devices includes the device number and the queue status. Meanwhile, the IoT device will start a countdown timer starting from T. According to the Ready message and the observed channel status, the MPG makes the data collection decisions for managed IoT devices and notifies them for scheduling the transmission. The IoT devices then will start transmitting data. If the data batch is processed successfully, the MPG will return the results with an ACK1 signal to the IoT device. However, there are three failure cases resulting from unpredictable dynamic environments as follows:

- *Case 1: Task Processing Incomplete.* The processing of the tasks cannot be completed by the end of the time slot. The MPG will send back an ACK 2 signal as well as the failed task number to the device.
- *Case 2: Task Missing.* If the MPG does not receive the scheduled tasks, the device will receive ACK 3 signal by the end of the time slot.
- *Case 3: Transmission Collision.* Batches from different devices are sent to the same MPG through the overlapped channel. The MPG will return a Collision message to all the above devices for re-transmission.

B. Task Recovery with Dynamic Contamination

To address the above task failures, we propose a task recovery mechanism with dynamic data contamination depicted in Fig.3. After receiving the Ready message, the MPG will decide the data batch size, which is compatible with the current wireless environment, available computation resources, and the recovery queue status. For instance, in time slot t in Fig.3, the data batch size is set to 4 since those 4 tasks are small. As a comparison, in time slot $t+1$, the data batch size becomes 2 because the second task is large. For those failed tasks, they will be attached to the tail of the recovery queue.

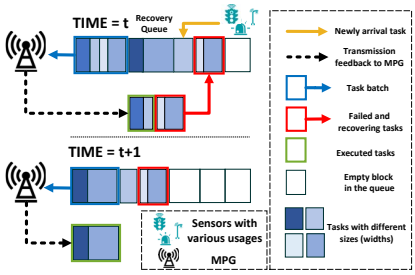


Fig. 3: Recovery queue example

C. QoE Optimization

Denote δ_{m_k} as the current queue length of IoT device m_k. If $\delta_{m_k} > V_{m_k}$ by the end of the current time slot, the task which arrives in the next time slot will be dropped due to the limited storage, denoted by $O_{m_k}^l = 1$. Although the recovery mechanism is able to mitigate the task failures, it introduces an extra delay that negatively affects the QoE for users. We quantize each user's QoE $O_{m_k}^d, m_k \in \mathcal{M}$, in the following

$$O_{m_k}^d = \sqrt{1 - \frac{\min(\tau_{m_k}^g, T) + \delta_{m_k} T}{(V_{m_k} + 1)T}} - O_{m_k}^l, \tag{4}$$

which indicates that both the short life-cycle latency and the short length of the recovery queue lead to a higher QoE.

We would like to maximize the QoE for all the users as

$$\max_{J_{m_k}, B_{m_k}^l, \nu_{m_k}} \sum_{j=1}^{J} \sum_{m_k \in \mathcal{M}_j} O_{m_k}^d, \tag{5a}$$

$$\text{s.t.} \quad \nu_{m_k} \in [1, \nu_{m_k}^{max}], \forall m_k \in \mathcal{M} \tag{5b}$$

$$\mathcal{B}_{m_k} \subseteq \mathbf{B}, \forall m_k \in \mathcal{M} \tag{5c}$$

$$R_{m_k}^{max} T > L_{m_k} \tag{5d}$$

where $B_{m_k}^l$ denotes the first channel unit the device m_k should use. J_{m_k} indicates whether the device should transmit data. (5b) constrains the maximum number of tasks that can be contaminated in a single batch. Constraint (5c) limits the channel units which can be allocated to the IoT device. Constraint (5d) indicates that each IoT device m_k is supposed to successfully transmit its batch with the maximum data rate when no interference exists.

V. DQN-BASED TASK COLLECTION DECISION MAKING

Maximizing the QoE in (5a) is a Mixed Integer Nonlinear Programming problem (MINLP), which is NP-hard [25]. To handle this, we propose to train the Deep Q-Network (DQN) on the MPGs to find the optimal policy for making decisions. Nonetheless, when a single DQN is deployed, increasing the number of devices will dramatically enlarge the searching space. Facing this challenge, we decompose the enormous DQN into M lightweight DQNs on J MPGs, where each DQN decides a combinatorial action for a single IoT device. In addition, the DQNs on the same MPG share the same channel state H_j in (2). We will show in Section VI that sharing channel state will improve the long-term QoE.

As for the network structure, it has been proved that taking into account the historical information can achieve a better performance than the traditional DQN [26]. In our paper, a Long short-term memory (LSTM) unit is applied in each DQN. We name the network structure as **S**tate-sharing **D**eep-**R**ecurrent **Q**-**N**etwork (SDRQN) according to the following specifications.

A. State Space

For each IoT device m_k trained on MPG j, the state $S_{m_k^j}$ consists of the queue length $V_{m_k^j}$, arrival task size $L_{m_k^j}$, and the shared channel gain table $\mathbf{H}_j$. We write the state as $S_{m_k^j} = \{V_{m_k^j}, L_{m_k^j}, \mathbf{H}_j\}$. The first two elements describe the intrinsic state of the device whereas the last one reflects the channel status between the transmitter IoT device and receiver MPG.

B. Action Space

Each MPG decides the actions for its managed IoT devices, which are composed of whether to send the collected data, channel unit selection, and the data batch size. The action is denoted as $a_{m_k} = \{J_{m_k}, B^l_{m_k}, \nu_{m_k}\}$ for each device m_k.

C. Reward Function

We define the reward $R(t)$ as the total QoE value after all devices take action at the end of each time slot

$$R(t) = \sum\nolimits_{j=1}^{J} R_j(t) = \sum\nolimits_{j=1}^{J} \sum\nolimits_{m_k \in \mathcal{M}_j} O^d_{m_k}(t). \quad (6)$$

The value of $R_j(t)$ could be close to 0 or even negative, which happens when the actions result in a high task drop rate.

D. Training Algorithm

In our neural network model, each SDRQN aims to learn a policy π_j from each state-action pair to a Q-value by updating the parameter $\theta^j_{m_k}$, denoted as $Q(S_{m_k^j}(t), a_{m_k^j}(t); \theta^j_{m_k})$. After the SDRQN outputs the Q-value of all possible actions, the MPG will dispatch the decisions $a_{m_k^j}$ to each IoT device based on the $\varepsilon - greedy$ policy shown as follows

$$a_{m_k^j}(t) = \begin{cases} argmax_{a_{m_k^j}(t)} Q_{\pi_j}(t) & 1-\varepsilon \\ \text{random action} & \varepsilon, \end{cases} \quad (7)$$

where ε is a small probability between 0 and 1. This policy encourages the exploration for a global optimum solution [27].

Each SDRQN includes a target DQN network with parameter θ' and a training DQN with parameter θ in the same structure. In the training stage, a mini-batch of experience is used to train the training DQN for updating the parameter θ based on the following loss function

$$L = (R_t + \gamma \max Q^*_\pi(\mathbf{S}'_t, \alpha'_t; \theta') - Q_\pi(\mathbf{S_t}, \alpha_t; \theta))^2. \quad (8)$$

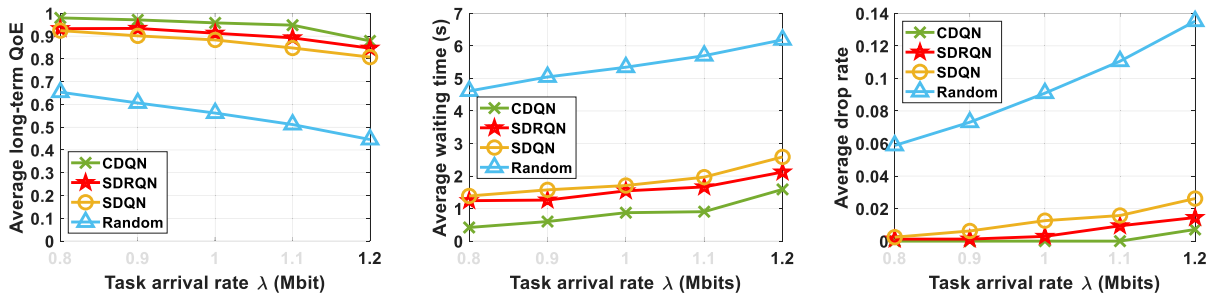
After D steps of training, the parameter θ' is renewed by setting it to θ. The proposed DQN-based algorithm is summarized specifically in Algorithm 1.

Algorithm 1 DQN-based long-term QoE optimization

1: Initialize the training network and target network with same random parameter $\theta^j_{m_k} = \theta'^j_{m_k}$ for each device m_k .
2: Set a empty replay buffer with size G for the SDRQN.
3: **for** $m = 1, 2, \ldots, M$ **do** episodes
4: Initialize the state S of the system.
5: **for** $n = 1, 2, \ldots, N$ **do** steps
6: Each MPG observes the state for its managed IoT devices $S_{m_k^j}(t), m_k^j = 1, 2, \ldots, \mathcal{M}_j$;
7: Input each state into the SDRQN of each device to train the parameter $\theta^j_{m_k}$;
8: Take action $a_{m_k^j}(t)$ based on the $\varepsilon - greedy$ policy and interact with the environment;
9: Calculate the corresponding reward $R_{m_k^j}(t)$;
10: Observe the environment at the next time slot $t+1$ and obtain the next state $S_{m_k^j}(t+1)$;
11: Update the experience replay buffer with the experience $\varsigma(t) = \{S_{m_k^j}(t), a_{m_k^j}(t), R_{m_k^j}(t), S_{m_k^j}(t+1)\}$;
12: Randomly select a mini-batch Ω of experiences from the replay buffer;
13: Train the training network with Ω with using loss function (8) and update the parameter θ_j of network;
14: **if** Training steps = D **then**
15: Update $\theta'^j_{m_k}$ of target network equals to $\theta^j_{m_k}$
16: **end if**
17: **end for**
18: **end for**

E. Network Structure

The SDRQN is a four-layer neural network where the first hidden layer is a fully connected layer with 32 neurons. It extracts the common feature of input states and reduces the dimension of the sequence to match the input interfaces of the next layer. The second layer is the LSTM layer with 32 features in the hidden state. This layer remembers the historical information about the input feature. It has been proved in [28] that introducing recurrent unit strengths the conventional multiple dense layers networks with partially observed input state. The LSTM layer is followed by two fully-connected layers with 32 and 16 neurons, separately. The output layer gives the Q-value of the combinatorial action. The output size varies based on the number of available channels and recovery batch size of the IoT devices.



(a) Average long-term QoE (b) Average waiting time (c) Average drop rate

Fig. 4: Data collection performance under different arrival task rates

VI. Performance Evaluation

In this section, we evaluate the performance of the proposed data collection scheme. The simulations are conducted using the Pytorch 1.10.0 and CUDA 11.3 on a desktop with the GeForce RTX 3060 graphic card.

A. Experiment Settings

We place 4 MPGs and 12 heterogeneous IoT devices by default in an indoor area with the size of $20m \times 20m$. The MPGs are placed at the coordinates $(5,5),(5,15),(15,5),(15,15)$, respectively whereas IoT devices are randomly distributed. The spectrum between 2400 MHz and 2420 MHz is divided into 20 orthogonal channel units with a unit bandwidth of 1MHz. These IoT devices are classified into 3 types working under different physical regulations, which are listed as follows

Device	Tx power(mW)	Total channels	channel units
Type A	100	5	4
Type B	20	3	6
Type C	50	2	10

TABLE I: Benefits of state sharing in data collection

We utilize the WINNER2 channel model [20] to simulate the wireless channel. Since we investigate the efficiency of the proposed recovery mechanism, we set a large task arrival rate $\lambda = 1$Mbits in default. As for the computational resource of the MPG, based on previous work [29], the process density ρ_m is set to 0.297 Gigacycles per Mbits (Gcycles/Mbs). The total computation capability of an MPG is 30 Gcycles/s.

B. DQN Description

The DQN of each device trained on the MPG shares the exact same settings. The learning rate is set to 0.001. The size of the mini-batch is 32. The ε is initialized as 0.9, decreasing 0.001 by each step to 0.01. We apply the Adam optimizer with the MSE loss function [30] to update the parameters. The reward discount factor is equal to 0.9. The network trains 33 episodes, each of which comprises 1000 steps. The first 3 episodes are used to fulfill the replay buffer with a size of 3000 experiences. The frequency D of updating the parameter of the target DQN is set to 100 steps.

We compare the SDRQN with the random decision scheme (Random) and the other two DQNs. Their differences from SDRQN is listed in the following:

1) Random: The IoT devices take random actions.
2) Centralized Deep Q-Network (CDQN): MPG trains a DQN for its managed devices in a centralized manner. The output is the action combination for all devices.
3) Decentralized State-Sharing Deep Q-Network (SDQN): SDQN has a similar structure with the proposed SDRQN, but the recurrent unit is replaced by a conventional fully-connected layer with the same size.

Meanwhile, three major metrics are considered: the average reward that each IoT device obtains; the average waiting time of all IoT devices including the queueing time before a successful task transmission and the life-cycle latency; the average drop rate of all the tasks.

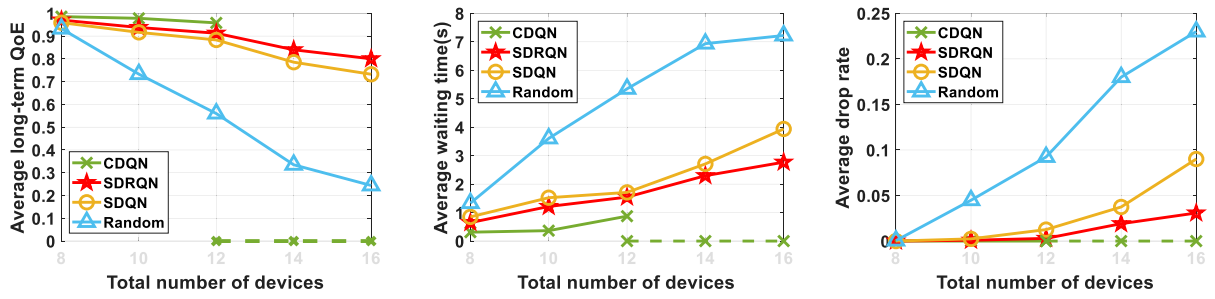
C. Result Analysis

Benefit of state sharing. We compare the decentralized neural network performance w/o the channel state sharing. The networks without channel state sharing are denoted as **I**ndependent-**S**tate DQN (ISDQN) and ISDRQN in Table.II, respectively. Specifically, in both ISDQN and ISDRQN, the input is the channel state between the IoT device itself and its manager MPG. The MPG trains the network for each of its managed IoT devices independently.

Structure	Average QoE	Waiting time(s)	Drop rate
SDRQN	**0.9127**	**1.5450**	**0.0029**
ISDRQN	0.8716	2.0260	0.0062
SDQN	0.8895	1.7190	0.0101
ISDQN	0.8381	1.9880	0.0239

TABLE II: Benefits of state sharing in data collection

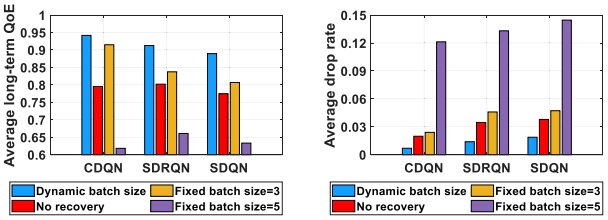
Among the four network structures in Table.II, SDRQN obtains the highest reward as well as the lowest waiting time and task drop rate. Without both the channel state sharing and the recurrent unit, ISDQN results in the lowest reward because of the high drop rate of 2.39%. As shown in the last column, via sharing channel state, the average drop rates in SDRQN and SDQN reduce to 0.29% and 1.01%, respectively, which are around 55% of those in ISDRQN and ISDQN. As for the waiting time, the SRDQN with the LSTM benefits more from the channel state sharing compared to the SDQN. Specifically, It reduces more than 10% of the waiting time compared to that of the SDQN.



(a) Average long-term QoE (b) Average waiting time (c) Average drop rate

Fig. 5: Data collection performance under different IoT device numbers

Impact of arrival task rate. Fig.4 depicts the data collection performance with different task arrival rates from 0.8Mbits/s to 1.2Mbits/s. When the task arrival rate is less than the unit data size, $\lambda \leq \tau$, the average drop rate is lower than 1%. The average QoE is higher than 0.9, indicating that all three DQNs have a satisfactory performance with a small λ. When the λ is larger than 1.1τ, the rewards of both decentralized DQN decrease dramatically to 0.85 and 0.8 separately. However, when the rate increases to 1.2τ, the waiting time is over 2 time slots in decentralized DQNs. Note that the τ can be easily adjusted to fit other networks.

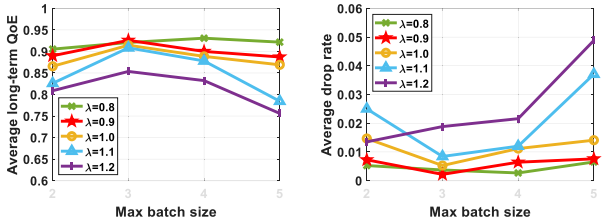


(a) Average long-term QoE (b) Average drop rate

Fig. 6: Performance with different recovery mechanisms

Effectiveness of proposed contamination mechanism. We compare data collection performance among the recovery mechanism with the dynamic contamination, the recovery mechanism with the fixed batch [31], and without the recovery mechanism. In the fixed batch size setting, the IoT device always packs a certain number of tasks in a batch. The IoT device working without the recovery mechanism only attempts to send a single task and will drop it if any failure occurs. According to the results in Fig. 6, when the recovery batch size is fixed to 5, the average drop rate is around 13%, which is even higher than that without the recovery mechanism. The reason is that the large batch size leads to the long life-cycle latency and further causes the collection failure. The recovery mechanism with the dynamic batch size reduces around 67% drop rate from that without the recovery mechanism in all three DQNs. In addition, the recovery mechanism with the dynamic contamination always achieves the highest reward among all three DQN structures.

Impact of max batch size in the dynamic contamination. Fig.7a illustrates the averaged long-term QoE and the task drop rate in SDRQN, where the curves are all convex. It demonstrates that there always exists an optimal max batch size given different task arrival rates. When $\lambda = 0.8$ Mbits/s, the optimal max batch size is 4 and the drop rate is only 0.26%. As λ increases, the optimal size is 3 in most cases. When the λ increases to 1.2 Mbits/s, the average drop rate is about 1.4% when $v_{m_k}^{max} = 2$, lower than 1.9% when $v_{m_k}^{max} = 3$. With this being said, we can tell that the optimal max batch size presents a decreasing trend as the arrival rate gets larger.



(a) Average long-term QoE (b) Average drop rate

Fig. 7: SDRQN performance under different max batch sizes

Scalability and necessaries of the SDRQN. To demonstrate the feasibility of SDRQN in the large-scale IoT network, we increase the number of IoT devices from 8 to 16. The number of IoT devices in type A, B, C are $[2, 3, 3]$, $[3, 3, 4]$, $[4, 4, 4]$, $[4, 5, 5]$, and $[4, 5, 7]$, respectively. As shown in Fig.5, all three DQN structures achieve the ultra-low drop rates, 0% when 8 heterogeneous IoT devices coexist and at most 0.24% with 10 devices. This is attributed to the sparse distribution of the IoT devices with weak interference most of the time. When the IoT network continues to scale up, the proposed SDRQN still maintains a great performance. Compared to the Random policy, SDRQN reduces 83% of the drop rate to about 3.9%. It also outperforms the 9% drop rate realized by SDQN.

Structure	SDRQN	SDQN	CDQN
Number of parameters	9768	10120	5121792
Memory allocated (MBytes)	0.16	0.17	144.54
Training memory (MBytes)	3.01	2.53	1177.14

TABLE III: Source utilization of different network structure

Table.III lists the computation and memory resources required for training different DQNs, where Memory allocated indicates the GPU memory allocated for initializing the neural network. Training memory represents the memory required for each update of the network parameters. As we can see, the SDRQN and SDQN only need around 10 thousand parameters, which is far less than over 5 million parameters in centralized CDQN. Meanwhile, there is a much higher level memory usage in CDQN than that in both SDRQN and SDQN. This demonstrates that the decentralized structure of DQN must be considered in a large-scale IoT network.

VII. Conclusion

In this paper, we proposed a sensing data collection scheme to enhance the long-term overall QoE in the heterogeneous IoT network, where task failures and the life-cycle latency are taken into consideration. A recovery mechanism with dynamic data contamination is proposed to address those challenges. Technically, to maximize the long-term overall QoE, a DRL algorithm with a lightweight DQN structure SRDQN is deployed for finding the optimal data collection policy. Our simulation results show that SRDQN has a better performance than CDQN, SDQN, and Random schemes. Furthermore, our data collection scheme is demonstrated to be efficient and effective in the large-scale heterogeneous IoT network.

References

[1] M. Ayaz, M. Ammad-Uddin, Z. Sharif, A. Mansour, and E.-H. M. Aggoune, "Internet-of-things (iot)-based smart agriculture: Toward making the fields talk," *IEEE Access*, vol. 7, pp. 129 551–129 583, 2019.

[2] H. H. Nguyen, F. Mirza, M. A. Naeem, and M. Nguyen, "A review on iot healthcare monitoring applications and a vision for transforming sensor data into real-time clinical feedback," in *2017 IEEE 21st International conference on computer supported cooperative work in design (CSCWD)*. IEEE, 2017, pp. 257–262.

[3] Y. Li, M. Sheng, C. Yang, and X. Wang, "Energy efficiency and spectral efficiency tradeoff in interference-limited wireless networks," *IEEE Communications Letters*, vol. 17, no. 10, pp. 1924–1927, 2013.

[4] S. Zhang, H. Zhang, B. Di, and L. Song, "Cellular uav-to-x communications: Design and optimization for multi-uav networks," *IEEE Transactions on Wireless Communications*, vol. 18, no. 2, pp. 1346–1359, 2019.

[5] Q. Li, S. Wang, A. Zhou, X. Ma, F. Yang, and A. X. Liu, "Qos driven task offloading with statistical guarantee in mobile edge computing," *IEEE Transactions on Mobile Computing*, vol. 21, no. 1, pp. 278–290, 2020.

[6] Y. Mao, J. Zhang, S. Song, and K. B. Letaief, "Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems," *IEEE Transactions on Wireless Communications*, vol. 16, no. 9, pp. 5994–6009, 2017.

[7] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Transactions on Wireless Communications*, vol. 16, no. 3, pp. 1397–1411, 2016.

[8] B. Kjell, A. B. Sergio, d. M. Katrien, D. Ann, and E. Sebastian, "Qualinet White Paper on Definitions of Quality of Experience," https://hal.archives-ouvertes.fr/hal-00977812/document, 2013.

[9] J.-S. Han, H.-S. Kim, J.-S. Bang, and Y.-H. Lee, "Interference mitigation in ieee 802.15. 4 networks," in *2011 IEEE Global Telecommunications Conference-GLOBECOM 2011*. IEEE, 2011, pp. 1–5.

[10] S. Wang, S. M. Kim, and T. He, "Symbol-level cross-technology communication via payload encoding," in *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2018, pp. 500–510.

[11] Z. Li and T. He, "Webee: Physical-layer cross-technology communication via emulation," in *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*, 2017, pp. 2–14.

[12] Z. Yin, W. Jiang, S. M. Kim, and T. He, "C-morse: Cross-technology communication with transparent morse coding," in *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*. IEEE, 2017, pp. 1–9.

[13] H. Shah-Mansouri and V. W. S. Wong, "Hierarchical fog-cloud computing for iot systems: A computation offloading game," *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 3246–3257, 2018.

[14] X. Zhang and J. Wang, "Joint heterogeneous statistical-qos/qoe provisionings for edge-computing based wifi offloading over 5g mobile wireless networks," in *2018 52nd Annual Conference on Information Sciences and Systems (CISS)*. IEEE, 2018, pp. 1–6.

[15] X. He, K. Wang, H. Huang, T. Miyazaki, Y. Wang, and S. Guo, "Green resource allocation based on deep reinforcement learning in content-centric iot," *IEEE Transactions on Emerging Topics in Computing*, vol. 8, no. 3, pp. 781–796, 2020.

[16] Y. Xiao and M. Krunz, "Qoe and power efficiency tradeoff for fog computing networks with fog node cooperation," in *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*. IEEE, 2017, pp. 1–9.

[17] W. Lu, P. Si, G. Huang, H. Han, L. Qian, N. Zhao, and Y. Gong, "Swipt cooperative spectrum sharing for 6g-enabled cognitive iot network," *IEEE Internet of Things Journal*, vol. 8, no. 20, pp. 15 070–15 080, 2021.

[18] N. Yang, H. Zhang, K. Long, C. Jiang, and Y. Yang, "Spectrum management scheme in fog iot networks," *IEEE Communications Magazine*, vol. 56, no. 10, pp. 101–107, 2018.

[19] W. U. Khan, J. Liu, F. Jameel, V. Sharma, R. Jäntti, and Z. Han, "Spectral efficiency optimization for next generation noma-enabled iot networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 15 284–15 297, 2020.

[20] H.-H. Chang, H. Song, Y. Yi, J. Zhang, H. He, and L. Liu, "Distributive dynamic spectrum access through deep reinforcement learning: A reservoir computing-based approach," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1938–1948, 2019.

[21] O. Naparstek and K. Cohen, "Deep multi-user reinforcement learning for distributed dynamic spectrum access," *IEEE Transactions on Wireless Communications*, vol. 18, no. 1, pp. 310–323, 2019.

[22] Z. Shi, X. Xie, M. Kadoch, and M. Cheriet, "A spectrum resource sharing algorithm for iot networks based on reinforcement learning," in *2020 International Wireless Communications and Mobile Computing (IWCMC)*. IEEE, 2020, pp. 1019–1024.

[23] Q. Tang, R. Xie, F. R. Yu, T. Huang, and Y. Liu, "Decentralized computation offloading in iot fog computing system with energy harvesting: A dec-pomdp approach," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4898–4911, 2020.

[24] K. Wu, J. Xiao, Y. Yi, D. Chen, X. Luo, and L. M. Ni, "Csi-based indoor localization," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 7, pp. 1300–1309, 2013.

[25] Z. Liu, Y. Han, J. Fan, L. Zhang, and Y. Lin, "Joint optimization of spectrum and energy efficiency considering the c-v2x security: A deep reinforcement learning approach," in *2020 IEEE 18th International Conference on Industrial Informatics (INDIN)*, vol. 1. IEEE, 2020, pp. 315–320.

[26] Y. Lin, M. Wang, X. Zhou, G. Ding, and S. Mao, "Dynamic spectrum interaction of uav flight formation communication with priority: A deep reinforcement learning approach," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 3, pp. 892–903, 2020.

[27] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA; London, England: The MIT Press, 1998.

[28] M. Hausknecht and P. Stone, "Deep recurrent q-learning for partially observable mdps," 2017. [Online]. Available: https://arxiv.org/abs/1507.06527

[29] M. Tang and V. W. Wong, "Deep reinforcement learning for task offloading in mobile edge computing systems," *IEEE Transactions on Mobile Computing, early access*, 2020.

[30] Q. Chen, Z. Kuang, and L. Zhao, "Multi-user computation offloading and resource allocation for cloud-edge heterogeneous network," *IEEE Internet of Things Journal, early access*, 2021.

[31] X. Chen, H. Zhang, C. Wu, S. Mao, Y. Ji, and M. Bennis, "Optimized computation offloading performance in virtual edge computing systems via deep reinforcement learning," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4005–4018, 2019.