Towards Deep Q-Network Based Resource Allocation in Industrial Internet of Things

Fan Liang*, Wei Yu*, Xing Liu*, David Griffith†, and Nada Golmie†
*Towson University, USA
Emails: {fliang1,xliu10}@students.towson.edu, wyu@towson.edu
†National Institute of Standards and Technology (NIST), USA
Emails:{david.griffith, nada.golmie}@nist.gov

***Abstract*—With the increasing adoption of Industrial Internet of Things (IIoT) devices, infrastructures, and supporting applications, it is critical to design schemes to effectively allocate resources (e.g., networking, computing, and energy) in IIoT systems, generally formalized as optimization problems. Nonetheless, because the system is highly complex, operation environments are time-varying, and required information may not be available, it is difficult to leverage traditional optimization techniques to solve the optimal resource allocation problem. To this end, in this paper we propose a Deep Q-Network (DQN) based scheme to address both bandwidth utilization and energy efficiency in an IIoT system. In detail, we design a DQN model that consists of two deep neural networks (DNN) and a Q-learning model. The DNN network abstracts the features from the highly dimensional inputs and obtains the approximate Q-function for the Q-learning model. Based on the Q-function, the Q-learning model can generate the Q-table and reward function. After the training process, the DQN model can select appropriate actions for the agents (i.e., robots in a smart warehouse in this study) to improve bandwidth utilization and energy efficiency. To evaluate our proposed scheme, we design a simulation environment to investigate a typical IIoT scenario: the actuation of robotics in a smart warehouse. We then implement the DQN model and conduct extensive experiments to validate the efficacy of our scheme. Our experimental results confirm that our scheme can improve both bandwidth utilization and energy efficiency, as compared to other representative schemes.**

***Keywords*—Industrial IoT, Deep Q-Network, Non-Orthogonal Multiple Access, Resource Management**

I. Introduction

Continuing advancements in the Internet of Things (IoT) and its related technologies have instigated wide adoption across a variety of industrial fields to assist in the monitoring, control, and automation of industrial manufacturing and production, also known as the Industrial Internet of Things (IIoT) [1]. Generally speaking, an IIoT system interconnects a massive number of physical computing devices via communication networking infrastructures. From a Cyber-Physical System (CPS) point of view, the IIoT system consists of both cyber and physical subsystems [2]. The cyber subsystem plays a key role in data collection, communication, and analysis, providing information communication and computing services for the physical devices (sensors, actuators, computing nodes, etc.). As the number of physical devices in IIoT systems increases, more and more IoT data must be transmitted, collected, and stored. Transmitting massive amounts of IoT data in turn consumes large network bandwidth resources [3]. Moreover, the massive number of physical devices that transmit data through the cyber subsystem also generate significant network congestion and further increase data transmission latency. Thus, limited network resources cannot satisfactorily support services for IIoT applications without the deployment of effective resource allocation schemes [4]. Further, the physical subsystem has resource limitations. As a number of devices are deployed with a portable energy supply, discontinuous power cycling can greatly affect the operation of IIoT systems. Thus, improving energy efficiency for IIoT systems is critical.

How to effectively allocate resources in an IIoT system can be formalized as a resource optimization problem. Nonetheless, conducting efficient resource allocation in an IIoT system has many challenges. First, connecting and integrating a large number of IoT devices means that IIoT environments are highly distributed and complex. Second, because IIoT environments are highly dynamic, IIoT systems can present different system states over time. Third, as it is not feasible to collect global information on the entire system, a number of existing resource allocation schemes that rely on global information are rendered ineffective [5], [6], [7]. With the increased number of IoT devices, finding the optimal solution is an NP-hard problem [8]. To address the resource allocation problem in IIoT systems, we leverage reinforcement learning as a distributed optimization solution. In IIoT systems, an IoT device (a robot, a sensor, an actuator, etc.) can be regarded as a learning agent in the optimization process, and thus an IIoT system is considered a multi-agent system. The complex conditions of the system and the large number of agents increase computing complexity immensely. However, through the learning process, an approximation of the optimal solution can be ultimately realized [9].

Generally speaking, Q-learning is a typical reinforcement learning technique that updates a Q-table by an appropriate policy to determine the next action of the agent [10], [11]. One limitation of Q-learning is the scalability, as the Q-table that represents the state-action space can be very large when the system is complicated. To deal with a system with a large state-action space, Deep Q-Networks (DQNs) adopt deep neural networks to infer the Q-value of new states by training on the previously explored states. Thus, applying DQNs to IIoT systems to assist resource allocation is a viable solution. Nonetheless, existing DQN algorithms generally focus on

finding the optimal solution for the single agent. For example, Lample *et al.* [12] leveraged deep reinforcement learning to play complex video games. Although the state-action space is large in the video game scenario, the DQN model only operates one agent (i.e., the player) that interacts with the dynamic environment. Note that in our case, all the IoT devices interact with various dynamic environments, forming a system with multiple agents. Therefore, how to leverage DQNs to obtain the optimal solution for IIoT systems is a challenging problem.

To address the aforementioned issue, in this paper we propose a DQN based scheme that carries out distributed resource allocation in a complex IIoT system. In detail, we collect the possible states and actions for all agents as the training dataset. We utilize the dataset to train the Deep Neural Network (DNN) model and obtain the approximate Q-function. The Q-learning model is leveraged with the Q-function to obtain the rewards for an agent to take different actions. By comparing all the rewards, the DQN scheme can select the next action that leads to the maximal reward for one agent. Then, we design the reward function for the multi-agent environment. Based on the Q-function and reward function, we can determine the optimal actions for all agents, to achieve the maximal reward for the entire system.

To summarize, we make the following contributions in our study.

- **DQN Based Scheme:** We propose a DQN based scheme to solve the distributed resource allocation problem in the IIoT system. The proposed DQN based scheme can control the agents in each step during travel to achieve the maximal bandwidth utilization and energy efficiency for the system. To be specific, we design a DNN network to handle the large state-action space and obtain the approximation of the Q-function. Then, we utilize the Q-learning algorithm to obtain the maximal reward.
- **Implementation and Extensive Experiments:** We consider a scenario to represent a typical IIoT environment and system model. Based on the defined system model, we implement the DQN based scheme by Python. In addition, we design a Python-based simulator to interact with the DQN model to evaluate our scheme. Furthermore, by leveraging the simulator and DQN model, we define the evaluation metrics and evaluate the efficacy of our proposed DQN based scheme comprehensively, comparing it with the standard Q-learning and the shortest path based schemes.

The remainder of this paper is organized as follows: In Section II, we provide background on key concepts and techniques, such as IIoT systems, Non-Orthogonal Multiple Access (NOMA), and DQN. In Section III, we conduct a brief literature review of the related studies in IIoT and deep learning. In Section IV, we present our design rationale and define the scenario and system model. In Section V, we first formalize the problem and then introduce our scheme in detail. In Section VI, we introduce the implementation details of the simulator and DQN based scheme, as well as parameter settings and experimental design. In Section VII, we present the evaluation results. Finally, we summarize the paper in Section VIII.

II. Preliminaries

In this section, we brief the background of IIoT systems, NOMA technology, and DQN, respectively.

A. IIoT Systems

Otherwise known as Industry 4.0, IIoT are systems that connect a number of IoT devices in industrial settings, such as manufacturing and production, in order to conduct the monitoring and control of those systems. The IoT devices with sensors collect data from the system environment, and a variety of data analysis techniques can be leveraged to extract understanding from the collected data. Based on the data analysis results, the system can take actions to automatically control and operate itself through actuation devices, with the goal of improving overall performance and reducing cost and waste. From a CPS perspective, cyber and the physical subsystems govern the function of IIoT. The two subsystems interact to achieve system automation. The cyber subsystem consists of networking infrastructures to support data transmission in IIoT. The physical subsystem consists of IoT devices that collect the data and control the system [13]. Nonetheless, as the number of IoT devices and the volume of generated data increase, network resources cannot satisfy the requirements of IIoT applications, especially for low latency applications. In the IIoT system, the physical subsystem consists of a massive number of IoT devices such as sensors, controllers, and actuators. Some IoT devices (robots, etc.) in the IIoT system highly depend on energy resources. As those devices may be battery powered or use renewable energy generation, they may be offline when energy is depleted. Thus, it is critical to design energy resource allocation schemes to consider resource limited IoT devices.

B. Non-Orthogonal Multiple Access (NOMA)

To fulfill the emerging requirements of IIoT systems, new generation wireless mobile communication technologies will be leveraged in IIoT to improve the performance of the cyber subsystem. As a viable technique, NOMA has great potential to support high spectral efficiency, low latency, massive device connectivity, high achievable data rate, high reliability, user fairness, high throughput, diverse quality of services (QoS), and energy efficiency [14].

Generally speaking, NOMA leverages two multiplexing mechanisms: power-domain multiplexing and code-domain multiplexing. On one hand, in power-domain multiplexing, instead of using different frequencies, users are assigned different power coefficients according to channel. On the other hand, code-domain multiplexing utilizes the successive interference cancellation (SIC) mechanism to decode signals. Although NOMA techniques can achieve better spectrum multiplexing performance in heavy communication scenarios, it does not have the ability to address several emerging challenges, such as the high number of users and resulting interference that are

present in IIoT scenarios, among others. Specifically, in IIoT, the large number of IoT devices communicating with each other leads to a massive number of connections and poses incredible signal interference [15]. To avoid interference and increase bandwidth utilization, NOMA leverages subchannels to allocate multiple users [16]. In our study, we leverage NOMA technology with subchannels to support IIoT systems.

C. Deep Q-Network (DQN)

Reinforcement learning algorithms can be divided into three categories: value-based, policy-based, and actor-critic [17]. The most common is the value-based algorithms represented by DQN, which have only one value function network and no policy network. The actor-critic algorithms, such as Deep Deterministic Policy Gradient (DDPG) and Trust Region Policy Optimization (TRPO) [18], [19], have both a value function network and policy network. Deep reinforcement learning combines deep neural networks with a reinforcement learning architecture that enables high dimensional inputs to find the optimal reward for agents to learn the best actions for long-term rewards. The reinforcement learning system consists of a dynamic environment and agents. The agents act in the environment in finite discrete time steps. As the DQN model cannot traverse all the situations for the environment, it cannot obtain the exact Q-function. Based on Bellman's equation, the training process is to find the approximate Q-function and obtain the policy [20]. Note that in our study, we leverage the DQN model to solve the distributed resource allocation problem in the investigated IIoT system.

III. Related Works

We now review some existing studies concerning IIoT and resource allocation that are relevant to our study.

The conception and progressing development of IIoT has seen it touted as the next wave of innovation in industrial fields [21]. Based on IoT devices (e.g., sensors, actuators, and controllers) that are deployed in industrial environments, industrial systems are able to collect large amounts of data. With advanced data analysis techniques such as deep learning, IIoT data can be further processed to manage and control IIoT systems. However, data transmission and analysis are resource-intensive, while IIoT systems are largely resource constrained. Thus, a variety of research efforts have focused on resource allocation in IIoT systems.

In the area of network resource allocation, a number of scheduling schemes have been proposed to efficiently use network resources. For instance, Li *et al.* [22] proposed a three-layer QoS scheduling framework, which can adjust the priority of applications based on QoS requirements. Their scheme provides services to application, network, and sensing layers. As demonstrated in their evaluation, the three-layer QoS scheduling framework for service-oriented IoT architecture can improve the performance in IoT networks. In addition, Turjman *et al.* [23] proposed a fully informed particle swarm (FIPS) optimization scheme based on the robust canonical particle swarm optimization (CPSO). In their study, network traffic are grouped into different categories, and different data traffic are optimized based on the category. Their evaluation results showed that the proposed schemes could improve both computing and network performance. Likewise, Basu *et al.* [24] proposed a hybrid algorithm, which combines the genetic algorithm and the ant colony optimization to balance computing payloads for computing resources on edge devices. Furthermore, as an intelligent model, the proposed algorithm can improve itself based on historical data, leading to a better computing payload balancing solution.

Since deep learning technologies provide excellent abilities in data mining and analysis, there is a number of research efforts devoted to leveraging deep learning to assist in resource allocation [25], [26], [27], [28], [29]. For example, Ye *et al.* [25] proposed a deep reinforcement learning based resource allocation scheme for vehicle-to-vehicle (V2V) communications. Their scheme can control an autonomous "agent" (V2V link or a vehicle) to find optimal sub-band and power level for transmitting data. The proposed scheme can minimize interference in vehicle-to-infrastructure communications. Likewise, Liang *et al.* [27] leveraged deep reinforcement learning to solve the wireless resource allocation problem in cognitive radio networks. They first discussed deep learning-assisted optimization for resource allocation, including supervised learning paradigms, objective-oriented unsupervised learning paradigms, and learning accelerated optimization paradigms. Then, they leveraged deep reinforcement learning techniques to deal with the resource allocation problem in wireless networks. Furthermore, Liu *et al.* [28] proposed a deep recurrent neural network based scheme to solve the resource allocation problem in the IoT system. Their evaluation results showed that the proposed scheme could optimally and rapidly allocate resources for IoT devices. The limitation of the scheme is that it only considers the resource allocation for each agent. Likewise, Sun *et al.* [29] proposed a deep learning based long-term power allocation (DL-PA) scheme, which obtains the optimal power level for the NOMA based wireless network. The evaluation results showed the proposed scheme can reduce the power usage and increase the average data rate.

In this paper, we focus on resource allocation problem in IIoT systems, and utilize a generic example to demonstrate our proposed algorithmic improvement. To be specific, we attempt to improve both bandwidth utilization and energy efficiency of the NOMA wireless network environment in our investigated smart warehouse IIoT system. In our study, we formalize the resource allocation problem as an MDP problem and propose a DQN based scheme to find the optimal solution for allocating bandwidth and energy resources of the investigated IIoT system. Further, we implement a simulator and our DQN based scheme, and conduct extensive experiments to evaluate the effectiveness of our scheme in comparison with two representative baseline schemes.

IV. System Model

In this section, we first present our design rationale and introduce the proposed IIoT scenario. Based on the scenario, we design the system model. Table I lists key notations in the paper.

TABLE I
NOTATIONS

Symbols	Descriptions
U_i	The i^{th} UE in the scenario
n	Number of UEs in the scenario
L	Length and width of the warehouse
l	Size of each packet
d_i	Distance from UE_i to the BS
B	Total bandwidth of network
$B^{'}$	Bandwidth of subchannel
N_m	Number of UEs in subchannel m
K	Number of subchannels
θ	Signal to Interference plus Noise Ratio (SINR)
D_i	Data rate for UE_i
a_i	Power coefficient
P	Transmitting power
$\|h\|$	Power fading gain of the link
σ^2	Average power spectral density of white Gaussian noise
μ	Transmission power level
α	Path loss exponent
q	Maximal number of packet collisions
R_t	Total reward of the system
R_d	Distributed reward of the system
r_b	Total reward for switching subchannels
r_{sub}	Reward for switching one subchannels
r_e	Reward for energy resource allocation
r	Reward of each action for one UE
γ	Expect discounted factor
ρ	Distance coefficient
Q_c	Distributed Q-function for the system

A. Design Rationale

Fig. 1 illustrates the problem space of IIoT, which consists of two dimensions (i.e., QoS requirements and the type of resources). In this paper, we focus on the resource allocation issues of both network bandwidth and energy. The shadow blocks indicate the problem that we focus on in this study. Specifically, we define a typical IIoT scenario and utilize NOMA techniques to support the communication of IoT devices. In our scenario, we consider that the network resources are limited and the IoT devices have limited battery power. In this case, optimizing bandwidth utilization and energy efficiency can improve the overall performance of IIoT systems. Thus, we consider both bandwidth utilization and energy efficiency in the IIoT environment and formalize the resource allocation problem as an optimization problem in IIoT environments. Recall that the challenge is that IIoT is a complex and dynamic system with a large number of IoT devices. Solving the distributed resource allocation problem in IIoT systems requires finding the optimal solution from the entire system perspective. In addition, finding the optimal solution for a large number of agents (nodes) is difficult, as the computation complexity increases rapidly. To this end, we propose a DQN based scheme to solve the resource allocation problem in a distributed manner for the investigated IIoT scenario.

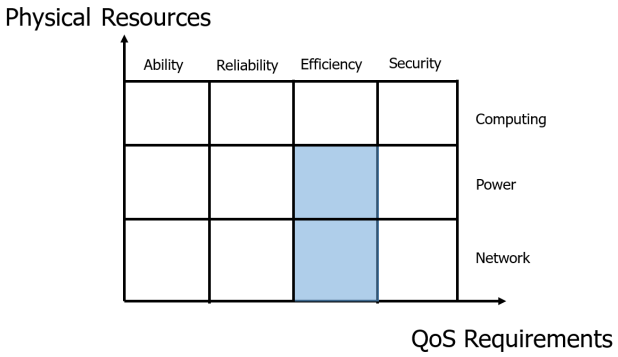


Fig. 1. Problem space of IIoT

We now introduce our design rationale that focuses on improving network utilization and energy efficiency. Recall the constraints in the IIoT system: the dynamic environments and a large number of IoT devices are the primary challenges to finding the optimal solution. There are some existing research efforts focused on solving the optimization problem for a single agent [28], [29]. Nonetheless, as we discussed, our goal is to find the optimal solution for all agents in the entire system. Based on a large number of devices and complexity of the environments in IIoT systems, finding the optimal solution in such a complex system is an NP-hard problem [30].

In our study, based on the features of NOMA, we present a co-design of the bandwidth and energy resource allocation scheme in NOMA based wireless network environments. In detail, we deploy the wireless network that adopts NOMA technology in the investigated IIoT scenario. Based on optimizing the path of UEs (i.e., robots in a smart warehouse), we can optimize the energy consumption for individual UEs. Further, we divide the wireless network resources into several subchannels to optimize the bandwidth utilization efficiency. Furthermore, reinforcement learning algorithms are viable for mapping actions and rewards based on accumulated experience in order to achieve maximum rewards. In our scenario, the IoT devices affect each other and interact with the dynamic environment. Thus, the state-action space of agents is very large. Because of the large size of the state-action space, it is impossible to traverse the entire space in practice, and the precise Q-function to represent the state-action space cannot be found. In this study, we design a simulator to generate training datasets. In detail, the simulator generates actions and calculates corresponding rewards. Then, we train the proposed DNN model with the dataset to establish the approximation of Q-function. Finally, based on the system model and the approximation of the Q-function, we design the reward function and leverage the DQN model to find the optimal solution for the multi-agent system.

B. Motivated Scenario

In the following, we introduce the investigated IIoT smart warehouse scenario in detail. We consider a typical smart warehouse, such as the various Amazon warehouses [31], in which unmanned vehicles carry packages around the ware-

house based on various requirements. Unmanned vehicles communicate with computing nodes via wireless network. In addition, the computing center distributes different tasks to each individual unmanned vehicle, and the unmanned vehicles carry packages to target destinations. During the package delivery process, unmanned vehicles constantly send location information and environmental data to the computing and operation center in order to optimize the delivery path.

Under this premise, we leverage the NOMA technology to provide wireless communication between the unmanned vehicles and base station (BS). We deploy a BS with a single antenna in the center of the warehouse. Here, the unmanned vehicle can be regarded as the UE in the investigated IIoT system. As we mentioned in Section II-B, to improve the bandwidth utilization, the bandwidth is divided into several subchannels according to different data transmission power levels. In addition, the energy consumption for data transmission depends on the distance between the UE and BS. Thus, different routes that a UE travels will affect bandwidth utilization and energy efficiency. To this end, finding optimal paths for UEs from their origins to destinations can improve the bandwidth utilization and energy efficiency of UEs when carrying the packages across the warehouse.

C. System Model

Based on the scenario is defined in Section IV-B, we now present the system model. We denote the length of the warehouse as L, the width of the warehouse as L, and the location for the BS as $(\frac{L}{2}, \frac{L}{2})$. In our case, the wireless network is based on NOMA technology and the total bandwidth is B. Note that serious interference and collisions will arise when all devices compete for the entire bandwidth resource. Thus, we divide the bandwidth into several subchannels and allocate subchannels as concentric rings according to the distance from the BS. Fig. 2 shows the system model. We denote the fixed packet length as l, the data transmission time slot as t_s, and the signal to interference plus noise ratio (SINR) as θ. We also denote $B^{'}$ as the bandwidth of a subchannel. Thus, the maximal number of subchannels K can be represented by $K = \left\lfloor \frac{B}{B^{'}} \right\rfloor$.

We denote a UE as $U_i = \langle \vec{v}, (x, y), n \rangle$, where $\vec{v}$ is the speed of the UE, (x, y) represents the location of the UE, and n is the series number of the UE. Here, a set of UEs, $\{U_1, U_2, \ldots, U_n\}$ compete for the resources. In addition, the UEs are randomly deployed in this area and carry packages. The data rate for a UE i can be represented by

$$D_i = B \cdot \log_2 \left(1 + \frac{a_i \cdot P_i |h_{i,s}|^2}{\sum_{j=i+1}^{n} a_j P_j |h_{j,s}|^2 + \sigma^2}\right). \quad (1)$$

Here, a_j is the power coefficient for U_i and $a_j = \frac{d_j}{\sum_{j=1}^{n} d_i}$, where d_i is the distance from U_i to the BS. In addition, $h_{i,s}$ is the power fading gain of the link from U_i to the BS and σ^2 is the average power spectral density of the white Gaussian noise.

The set of the different transmission power levels of subchannels is $\{\mu_1, \mu_2, \mu_3, \ldots \mu_K\}$ and each subchannel can tolerate at most q packet collisions to alleviate the decoding failure. Also, we denote θ_i as the SINR at U_i. Thus, the transmission power level can be represented by

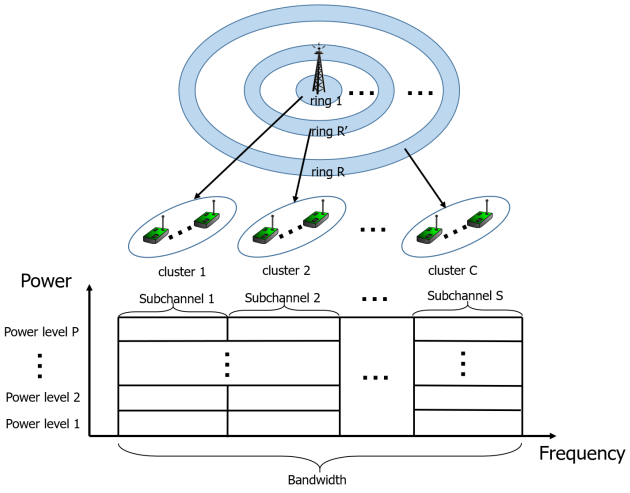


Fig. 2. System structure

$$\mu_i = \theta_i \left(q\theta_i + 1\right)^{i-1} \sigma^2 B^{'}. \quad (2)$$

The total power usage at time t can be derived by

$$P_i\left(t\right) = \frac{\mu_i}{h_{i,s}(t) d_i^{-\alpha}}, \qquad P_i \in \mu_i. \quad (3)$$

where, α is the path loss exponent [32].

Without any interference, the transmission power of each UE depends on the distance between itself and the BS. To reduce the power consumption on data transmission, the UE needs to travel through a route that brings it close to the BS. Nonetheless, UEs competing for bandwidth creates interference to the wireless network. The interference from different moving UEs becomes a key factor affecting the energy consumption through transmission power. In addition, the subchannels are defined by concentric rings centered on the base station. As shown in Fig. 2, when a UE changes subchannels by leaving one ring and entering another ring, the possibility of packet collision in the UE's new subchannel increases. If the number of packet collisions is larger than q, the system cannot decode the packet. Thus, determining whether a UE should switch subchannels or not is the key factor that affects bandwidth utilization.

V. Our Scheme

In this section, we introduce our proposed DQN based scheme in detail. First, we describe the problem and provide the reasoning for adopting reinforcement learning to solve the resource allocation problem in the IIoT system. Then, we present resource allocation algorithms for both bandwidth and energy resources. Finally, we transform the resource allocation problem into an optimal path discovery problem and detail our proposed DQN based scheme to solve the problem.

A. Problem Formalization

In our investigated IIoT system, because we are leveraging NOMA technology and subchannels, the relative location of each UE affects the subchannel selection and data transmission power. As the UEs are moving, the next action of each UE affects the bandwidth utilization and energy efficiency. To this end, the UEs and communication environments at any time t can be regarded as a pair of dynamic agents and environments. At any time step t, the agent enters the state S and selects an action a from all possible actions. This decision is only related to the current state, meaning that it has Markov property and can be regarded as the Markov Decision Process (MDP). Taking action a at time t causes the current state S to transition to the next state $S^{'} = T(S, a)$ at time $t+1$ and an immediate reward $R = R_{function}(S, a)$ is provided (if R is a negative number, we mean that penalty will be enforced). Here, T refers to transition function and $R_{function}$ refers to reward function.

When the agent selects the decision sequence from the initial state S_1 to final state S_n, $\mathbb{A} = \{a_1, a_2, a_3, \ldots, a_n\}$, the total reward obtained is the sum of individual rewards: $R_{total} = R(S_1, a_1) + R(S_2, a_2) + \ldots + R(S_n, a_n)$. The goal of the MDP is to find an optimal policy function π^*, which can make the dynamic agents obtain the maximum reward. The policy function determines the optimal action sequence $\mathbb{A}$. In our investigated IIoT system, the optimal policy function π^* can determine the action sequence $\mathbb{A}$ for the UE to guide the moving path from the source location to the destination.

A variety of techniques exist (game theory, etc.) that focus on solving resource allocation problems [33], [34], [35]. Those techniques consider the predicted and actual behaviors of individuals in the dynamic process, and study their optimization strategies. For instance, Watkins *et al.* [36] proposed a machine learning scheme to find π^* as the optimal policy for the agent, leading to the maximum long-term reward. Nonetheless, recall that because the IIoT system is complex, operation environments are time-varying, and sufficient information may not be available globally. Thus, it is difficult to leverage traditional optimization techniques to solve the problem, especially, in a system with a large state-action space and a number of agents.

Recall in our scenario that, to improve the bandwidth utilization and energy efficiency, the UEs try to find an action sequence $\mathbb{A}$ such that the optimal path can be found and the maximal reward R_{total} can be obtained. Due to the massive number of UEs and complex and dynamic environments in the investigated IIoT system, sufficient information may not be available globally. In addition, the large number of IoT devices (i.e., agents) leads to a huge state-action space, which makes it challenging to traverse the entire possible state-action space. Thus, it is difficult to use Q-learning to find the optimal π^* for the system. To this end, we leverage DQN to solve the problem, which will be detailed in the following subsections.

B. Resource Allocation Scheme

We now present our designed resource allocation scheme for both bandwidth and energy optimization. Because the entire bandwidth is divided into several subchannels to avoid interference and each subchannel has limited capacity, when a UE moves from the current subchannel to the target subchannel, the number of data packets increases in the target subchannel. That is, the packet collisions q increase. In this case, if $q^{'}$ is larger than the threshold in the target subchannel, the data cannot be decoded by the system. Thus, it is necessary to identify the conditions of current and target subchannels when the UE is crossing the subchannels. To this end, we define a reward r_{sub} to evaluate two actions (i.e., cross and stay), which can be represented by

$$\begin{cases} 0 < B_m^{'} - N_m \cdot b < B_{m+1}^{'} - N_{m+1} \cdot b & (r_{sub} = 5) \\ B_m^{'} - N_m \cdot b \leq 0 < B_{m+1}^{'} - N_{m+1} \cdot b & (r_{sub} = 10) \\ B_{m+1}^{'} - N_{m+1} \cdot b = B_m^{'} - N_m \cdot b & (r_{sub} = 1) \\ B_m^{'} - N_m \cdot b > B_{m+1}^{'} - N_{m+1} \cdot b > 0 & (r_{sub} = 0.1) \\ B_{m+1}^{'} - N_{m+1} \cdot b = 0 < B_m^{'} - N_m \cdot b & (r_{sub} = 0.01) \end{cases} \tag{4}$$

Here, b represents the bandwidth occupation for each UE, and $b = \frac{l}{t_s}$ (Here, I use t_s instead of t). Denote B_m as the bandwidth of current subchannel and B_{m+1} as the bandwidth of target subchannel. Thus, $B_m^{'} - N_m \cdot b$ is the amount of bandwidth available in the current subchannel and $B_{m+1}^{'} - N_{m+1} \cdot b$ is the amount of bandwidth available in the target subchannel.

Equation (4) defines the reward value r_{sub} that can be acquired when a specific UE switches subchannels. Generally speaking, if the target subchannel is busy, the UE that stays in the current subchannel could obtain a larger reward; otherwise, switching to the target subchannel leads to a larger reward. In particular, consider the first case (i.e., $r_{sub} = 5$), in which the next subchannel has more bandwidth resources than the current subchannel. Here, the UE switching subchannels obtains a high reward. In the last case (i.e., $r_{sub} = 0.01$), no bandwidth resources are available in the next subchannel, and the current subchannel has bandwidth resources remaining. Here, the UE switching subchannels results in very low reward. Based on Equation (4), we thus design the bandwidth resource selection algorithm, which is provided in Algorithm 1. Basically, the algorithm detects the related parameters for both current and target subchannels, and returns the reward values of both actions. Based on the analysis of the algorithm, the time complexity is $O(4n) \approx O(N)$, where n is the number of UEs in the subchannel and N is the scale of the problem.

Furthermore, we must consider energy resource allocation. Recall from the definition of the NOMA technology in Section II and the system model in Subsection IV-C that the transmission power of each UE depends on the distance d_i between itself and the BS. Considering the interference between the UE and the surrounding environment, we transform the energy resource allocation problem into an optimal path discovery problem. The system computes the transmission power P_{m+1} for the next possible positions of the UE. Then, the UE is guided to move to the position with the lowest transmission power. Similar to bandwidth resource allocation, Algorithm 2 illustrates the energy resource allocation procedure. The algorithm also returns the reward based on the action that the UE selects. The computation complexity of Algorithm 2 is $O(N)$, where N is the scale of the problem.

Algorithm 1: Bandwidth Resource Allocation

Data: $U_i(x_i, y_i)$: location of U_i, K_i: current subchannel, $B_m^{'}$: bandwidth of current subchannel, $U_{n,m}$: number of UEs in current subchannel, $B_{m+1}^{'}$: bandwidth of target subchannel, $U_{n,m+1}$: number of UEs in target subchannel.
Result: r_b: Total reward for the switch subchannel action from network perspective.

initialization
while $U_i; (i = 1, 2, 3 \cdots, n)$ *cross the subchannel* **do**
  update the bandwidth remaining $B_{m,r}^{'}$ and $B_{m+1,r}^{'}$ for current and target subchannels
  Update the q_m and q_{m+1} for current and target subchannels
  if $B_m^{'} - N_m \cdot b < B_{m+1}^{'} - N_{m+1} \cdot b$ **then**
    if $B_m^{'} - N_m \cdot b > 0$ **then**
      cross the subchannel and return $r_{sub} = 5$
    else
      cross the subchannel and return $r_{sub} = 10$
  if $B_{m+1}^{'} - N_{m+1} \cdot b = B_m^{'} - N_m \cdot b$ **then**
    cross the subchannel and return $r_{sub} = 1$
  if $B_m^{'} - N_m \cdot b < B_{m+1}^{'} - N_{m+1} \cdot b$ **then**
    if $B_{m+1}^{'} - N_{m+1} \cdot b > 0$ **then**
      stay in the current subchannel and return $r_{sub} = -0.1$
    else
      stay in the current subchannel and return $r_{sub} = 0.01$
  $r_b = \sum_{i=1}^{n} r_{sub,i}$
return total reward r_b

Algorithm 2: Energy Resource Allocation

Data: $U_i(x_i, y_i)$: location of U_i, K_i: current subchannel, $U_{n,m}$: number of UEs in current subchannel, $\mathbb{A}$: action set
Result: r_e: Total reward for the switch subchannel action from energy perspective.

initialization
while U_i *selects actions in action set* $\mathbb{A}$ **do**
  update the sending power level $\mu_{i,t+1}$
  if $\mu_{i,t+1} > \mu_i$ **then**
    check other actions in action set $\mathbb{A}$ and return maximum $r_{e,i}$
  else
    move to the lower power level position and return $r_{e,i}$
  $r_e = \sum_{i=1}^{k} r_{e,i}$
return total reward r_e

C. Map Resource Allocation Scheme to DQN Model

Recall in Subsection V-A that the resource allocation problem can be formalized as an MDP problem. We now leverage the DQN based scheme to solve the MDP problem. In the proposed scenario, we assume that a UE travels from source location A to destination B. During travel, the UE may cross multiple rings associated with subchannels. Further, the distance between the UE and BS constantly changes during travel while the relative positions between UE and surrounding UEs change as well. Thus, depending on the paths and locations of UEs, the network environment and data transmission power level constantly change. To this end, as the moving path directly affects bandwidth utilization and the energy efficiency of UEs, finding suitable paths for all the UEs is the key to improving the bandwidth utilization and energy efficiency of the system.

Because of the uncertainties of the dynamic distributed system, some necessary information may not be obtained before actions must be taken, such as interference, channel status, and others. Thus, it is challenging to compute the optimal path before the UE travels. Furthermore, because of the large number of devices in the system, the state-action space is large. Recall that it is difficult to use the standard Q-learning algorithm with a huge state-action space. Thus, we design a DQN based scheme to solve the problem. We first leverage the proposed simulation to generate training data and using the data to train the DQN model. After training, the well-trained DQN model can guide UEs to select better actions in order to obtain a higher total reward. In our case, the DQN based scheme dynamically guides each UE to find its next position, instead of planning the path for the UE before it moves. By doing this, the DQN based scheme can guide paths for all UEs, en-route, during system runtime.

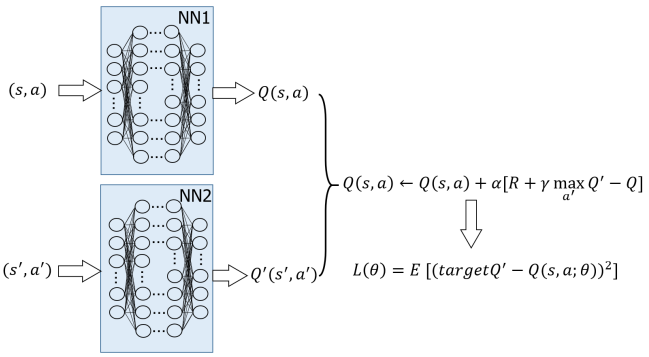


Fig. 3. Structure of DQN model

D. DQN Based Scheme for Resource Allocation

We now present the design of the DQN based scheme. Fig. 3 shows the structure of the DQN model. The goal of our DQN based scheme is to find the maximal reward of actions for the UE. Based on Bellman's Equation, the expression of our model is represented by

$$Q(s, a) = R(s, a) + \gamma \max Q^{'}\left(s^{'}, a^{'}\right), \tag{5}$$

$$Q(s, a) \leftarrow Q(s, a) + \alpha\left[R + \gamma \max_{a^{'}} Q^{'}\left(s^{'}, a^{'}\right) - Q(s, a)\right], \tag{6}$$

Equations (5) and (6) represent the Q-function of the proposed DQN model, where $Q(s, a)$ is first derived from Equation (5) and is then used to update its value in the next round shown in Equation (6). Before the learning process begins, Q is initialized as a fixed value. According to action a that is selected by the agent, the agent goes to a new state $s^{'}$ and receives a reward R. Here, the maximal $Q^{'}$ can be determined by the next state $s^{'}$ and possible action $a^{'}$ multiplied by the expected discount factor γ. According to the Q-function, we define the loss function as

$$L(\Theta) = E\left[\left(target\, Q - Q(s, a; \Theta)\right)^2\right]. \tag{7}$$

Equation (7) represents the loss function. The loss function minimizes the error estimation by optimizing the weights Θ.

Here, $target\,Q$ is the approximate Q-value that is calculated by the Q-function and $L(\Theta)$ represents the distance between the predicted Q-value of the next action and the Q-value of the current action. Also, $target\,Q$ can be represented by

$$target\,Q = R + \gamma max_{a'} Q'\left(s', a'; \Theta\right). \tag{8}$$

Recall that in Fig. 3, to avoid overfitting, we design two DNN networks, each with two hidden layers. In our scenario, the actions and rewards are independent, meaning that the reward for action a does not affect the reward for action a'. Thus, only using one DNN network may lead to an overfitting problem. To overcome this issue, we leverage two DNN networks to estimate the two neighboring Q-values for t and $t+1$ to avoid overfitting. The input is (s, a), which represents a pair of corresponding state and action. In order to obtain the optimal solution, the input a is a vector that includes all the UE's actions at time t. We compute the reward R for each action vector a and use Eq. (7) as loss function to estimate the two neighboring Q-values at time t and $t+1$.

Having defined the Q-function and the loss function $L(\Theta)$, we need to define the reward function. For the DNN, the reward r is identified by each pair of state s and action a. In our scheme, we define the total reward R_t by

$$R_t(s, a) = r_b(s, a) + r_e(s, a). \tag{9}$$

Here, R consists of two terms: r_b represents the reward to bandwidth utilization for taking a specific action, and r_e represents the reward to energy efficiency for taking a specific action. Also, r_b and r_e are determined by

$$\begin{aligned} &r_b(s, a) \\ &= \log_2 D_i + \log_2 r_{sub} = log_2 D_i \cdot r_{sub} \\ &= \log_2\left(B \cdot \log_2\left(1 + \frac{a_i \cdot P_i |h_{i,s}|^2}{\sum_{j=i+1}^{n} a_j P_j |h_{j,s}|^2 + \sigma^2}\right) \cdot r_{sub}\right). \end{aligned} \tag{10}$$

$$r_e = \frac{1}{log_2 P_i} = \frac{1}{log_2\left(\frac{\theta_i (q\theta_i + 1)^{i-1} \sigma^2 B'}{h_{i,s} d_i^{-\alpha}}\right)}. \tag{11}$$

Due to the large state-action space, we cannot traverse all the cases. Instead, finding an approximate $R_d' \approx R_d$ is necessary. Here, R_d is the distributed reward of the system, and can be represented by

$$R_{d,t}(\mathbb{S}, \mathbb{A}) = \frac{R_t^{U_{1,t}} + R_t^{U_{2,t}} + \cdots + R_t^{U_{n,t}}}{n}, \tag{12}$$

and R_d' can be represented by.

$$\begin{aligned} R'_{d,t}(\mathbb{S}, \mathbb{A}) &= \frac{\rho_1 R_t^{U_{1,t}} + \rho_2 R_t^{U_{2,t}} + \cdots + \rho_n R_t^{U_{n,t}}}{n} \\ &= \frac{\sum_{i=1}^{n} \rho_i R_t^{U_{i,t}}}{n}, \end{aligned} \tag{13}$$

where each ρ is a distance coefficient to represent the distance relationship between UE U_i and surrounding UEs. We define ρ_i by

$$\rho_i = \frac{log_2\left(d_{U_i, U_{i+1}} + 1\right)}{log_2\left(d_{U_i, S} + 1\right)}, \tag{14}$$

where $d_{U_i, U_{i+1}}$ denotes the distance between UE and surrounding UEs, and $d_{U_i, S}$ denotes the distance between UE and BS.

Then, we leverage reward function R_d' to train the DQN model, in order to find the Q-function and Q-value. Algorithm 3 illustrates the detailed procedure of the DQN model training process. Based on the system model and DQN based resource allocation algorithm, we show the implementations in the following section.

Algorithm 3: Deep Q-Learning

Data: $(\mathbb{S}, \mathbb{A})$: input dataset, R_d': labels
Initialize replay memory *episode* and random α,γ
Initialize the loss function $L(\theta)$
Result: $\mathbb{A}$: action set.
initialization
while *episode* $= 1$, *Memory DNN* **do**
  Initialize sequence $s_1 = \{x_1\}$ and preprocessed sequence $\phi_1 = \phi(s_1)$
  while $t = 1$, *Target DNN* **do**
    Select random action $a_{i,t}$ from each UE
    Otherwise, define $a_{i,t} = argmax_a Q(\phi(s_t), a, \theta)$
    Do the action $a_{i,t}$ and calculate reward Rt
    Set $s_{t+1} = s_t, a_{i,t}, x_{t+1}$ and $\phi_{t+1} = \phi(s_{t+1})$
    Store (s, a, r, s') in *episode* memory
    while *in episode memory* **do**
      random select sample (s, a, r, s')
      if *episode terminates at step* $j+1$ **then**
        set $y_j = r_j$
      else
        set $y_j = r_j + \gamma max_{a'} \hat{Q}(\phi_{j+1}, a'; \theta)$
    Calculate the loss $L(\theta)$ by loss function
    Reset $\hat{Q} = Q$

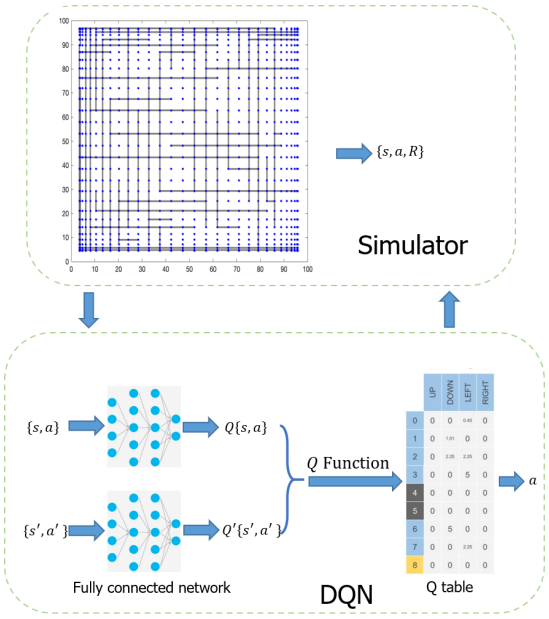


Fig. 4. System architecture

VI. Implementation

In this section, we introduce our implementation to validate the efficacy of our scheme. We first present our simulator program and propose the DQN model in detail. Then, we identify the related parameters. Finally, we present the experimental design.

A. Implementation

To simulate the proposed IIoT scenario and generate related data for the DQN model, we first design a simulator. We leverage Python 3.6 to design the simulator that can generate time-series IIoT data. The simulator can interact with the DQN model in order to train the model. We implement the NOMA scheme and integrate it into the simulator. The simulator can define the number of UEs and randomly generate their locations. In addition, the simulator can operate each UE to move up, down, right, or left (in 2D space). The simulator operates UEs and records environment information to create the action and reward datasets. We utilize the data as the training dataset, which is formed as $\{S, a, R\}$, to train the DNN model. Here, S represents the current stage, a represents the action set, and R represents the reward.

With the simulator designed and data generated, we implement the proposed DQN based scheme. To be specific, we first leverage the Tensorflow library[1] to design a typical DNN model that includes four dense layers. We select Rectified Linear Unit (ReLU) as the activation function. Then, we use the Keras library to create the DQN. We set four actions for each UE, which refer to up (U), down (D), left (L), and right (R). In addition, we implement the reward function that we defined in Section V to compute the distributed reward. Finally, we connect the DNN model and Q-learning model. Fig. 4 shows the system architecture of our implementation.

B. Parameter Settings

To generate the dataset, we configure parameters to initialize the system. In particular, we define the area of the warehouse to be 100*100 m^2. A base station (or access point) is located in the middle of this area at the coordinates $(50, 50)$. The simulator generates four evaluation cases, which include different numbers of UE ($n = 5, 25, 50, 100$). The UEs are randomly generated in the area with a constant speed v =1 m/s. The maximal capacity of a subchannel is defined as $C = \lfloor 80\%n \cdot b \rfloor$. The number of subchannels is 10. We assume the total transmission power P is 23 dBm. In addition, we define $\alpha = 3.5$, $\sigma^2 = -153\ dBm/Hz$, and θ =0 dB as the path loss exponent, noise power spectral density, and SINR threshold at BS, respectively. The maximal downlink bandwidth is 100 MHz. Since there are 10 subchannels, the downlink bandwidth for each subchannel is 10 MHz. In addition, we set the packet size as 1000 bits and the time slot as 0.1 s.

We now define the parameters for the DQN model. For the DNN, we set epochs to 1000, the learning rate to 0.001, the input size as N, and the output as 1. For the Q-learning model, we define the action set as $\{U, D, L, R\}$, the exploration factor as 0.2, and the backtracking as 10 iterations. In addition, to avoid redundant path discovery, we set ε as a reward threshold and $\varepsilon = -180$. If the reward is less than ε, then the UE has traveled too long and still cannot arrive at the destination. In this case, the Q-learning model has learned the experience, and the Q-learning model marks this iteration as "Lose". Otherwise, the models is marked as "Win". Table II lists the settings for all key parameters [37], [38].

[1]Certain commercial equipment, instruments, or materials are identified in this paper in order to specify the experimental procedure adequately. Such identification is not intended to imply recommendation or endorsement by the National Institute of Standards and Technology, nor is it intended to imply that the materials or equipment identified are necessarily the best available for the purpose.

TABLE II
PARAMETERS AND VALUES

Parameters	Values
Area of the warehouse	100*100 m^2
Number of UE (n)	5, 25, 50, 100
Speed of UE (v)	1 m/s
Number of subchannels (K)	10
Total power (P)	23 dBm
Path loss exponent (α)	3.5
Noise power spectral density (σ^2)	-153 dBm/Hz
SINR (θ)	0 dB
Downlink bandwidth (B)	100 MHz
Packet size (l)	1000 bits
Time slot (t)	0.1 s
Epoch	1000
Learning rate	0.001
Exploration factor	0.2

C. Experiment Design

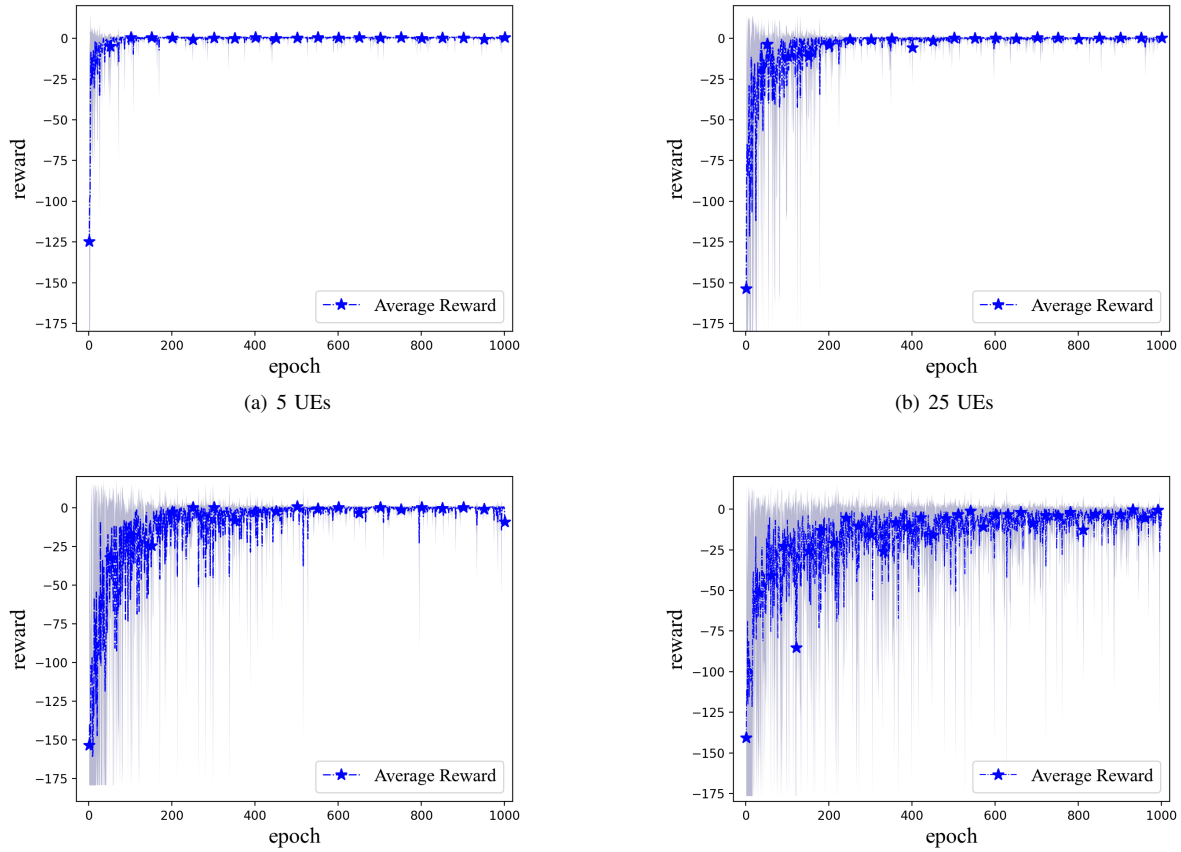
Based on the aforementioned implementation and parameter settings, we design experiments to evaluate the efficacy of our scheme. To comprehensively evaluate the proposed DQN scheme, we generate four evaluation groups with different numbers of UEs (i.e., 5 UEs, 25 UEs, 50 UEs, and 100 UEs). The simulator generates tasks (the source locations and destination locations) for each group within a given time period and calculates the relevant data, including average waiting time, average energy efficiency, reward, and actions, among others. Then, we use the data that is collected to train the DQN model and leverage the well-trained model to control the simulator to evaluate the performance of the DQN model. We execute experiments multiple times for each evaluation group to obtain the mean value. Also, as one baseline scheme, we implement the standard Q-learning scheme to solve the resource allocation problem in our IIoT system and compare its performance with our scheme. As the other baseline scheme, we utilize the simulator to generate the shortest path for each UE and calculate bandwidth utilization and energy efficiency.

VII. PERFORMANCE EVALUATION

We now detail the evaluation results of the experiments outlined in Section VI. In the following, we first present the evaluation methodology, and then detail the performance evaluation of our proposed DQN model compared to the baseline schemes.

A. Methodology

1) Compared schemes: We leverage our designed simulator to generate related data and utilize the collected dataset to train



(a) 5 UEs

(b) 25 UEs

(c) 50 UEs

(d) 100 UEs

Fig. 5. Average reward for the system with different number of UEs

the DQN model. After training, we connect the simulator and DQN model together as an interactive system, which allows the well-trained DQN model to determine the actions for each UE in the simulator. Further, we record the results to evaluate our scheme. In detail, we leverage the simulator to create random tasks and different numbers of UEs. Based on the NOMA scheme and transmission power level, the simulator generates the shortest paths for individual UEs depending on the tasks. Meanwhile, the simulator computes the waiting time and energy consumption for the entire system until all the UEs complete all the tasks. We record all the actions and rewards sequentially. We then use this dataset as the training data to feed the DNN model. After training the DNN model, we obtain the approximate Q-function for the Q-learning model. Based on the reward function that we have introduced in Section V-D and the Q-function, the DQN model can identify the optimal action that can obtain the maximal reward for UEs.

Because the key to our scheme is to discover the optimal path for each UE, we compare the overall average waiting time and energy efficiency of utilizing the proposed DQN model with the two baseline schemes: the standard Q-learning and the shortest path based schemes. Specifically, as the first baseline scheme, we leverage the standard Q-learning to find the optimal solution for each UE. Because of the dynamic environment and a large number of UEs, the state-action space is large. In order to leverage the standard Q-learning to solve the problem, we need to reduce the state-action space. To do so, we only leverage Q-learning to find the optimal solution for each UE, which reduces the state-action space. Then, we leverage the mean value of waiting time and energy consumption for all the UEs to calculate the overall average waiting time and average energy efficiency, which will be formally defined later.

Similarly, as the second baseline scheme, the shortest path based scheme only considers the energy efficiency for one UE. Moving through the shortest path can obtain the best energy efficiency for each individual UE. Nonetheless, the best energy efficiency for each UE is the purely local optimal solution. Only leveraging the shortest path algorithm cannot obtain the optimal solution for the entire system. In this case, we compare the average waiting time and energy efficiency of the proposed DQN based scheme, compared to the standard Q-learning and the shortest path based schemes.

In our study, we first evaluate the performance of the proposed DQN based scheme. Our DQN based scheme combines the DNN network and Q-learning models. In order to evaluate our scheme, we first leverage the simulator to generate 5000 data samples as the training dataset and 500 data samples as the testing dataset. Then, we train the DNN model to achieve 93.7 % accuracy. After training the DNN network, we define

4 evaluation cases, which consist of 5 UEs, 25 UEs, 50 UEs, and 100 UEs, respectively. Based on the dataset from each group, we train the DQN and evaluate the average reward of the actions by using DQN model. In addition, we evaluate the average travel time for all UEs from the source location to the destination location. We use the well-trained DQN model to control the UEs and record the average travel time. Since the speed of a UE is constant, a shorter travel time indicates the shorter path of travel.

To evaluate the various models, we consider the overall average waiting time and average energy efficiency. We first evaluate the performance of the proposed DQN model alone. Then, we leverage the model to interact with the simulator so that the performance of the IIoT system can be determined. Note that as the data is randomly generated, and thus we execute procedure 20 times for each case and consider the mean value.

2) Evaluation Metrics: Based on the outlined scope and evaluation methodology, we define the following metrics to evaluate the performance of our proposed scheme.

Average Reward: The DQN utilizes rewards to measure good and bad transitions from the current state to the next. As detailed in Equations (8), (9), (10), and (12), we have defined the reward function for the proposed DQN model. Here, we use the average reward as a metric to evaluate the convergence speed of our DQN model. We define the average reward as the total reward divided by the number of UEs. A higher average reward indicates that the system obtains higher profits.

Average Travel Time: We define the average travel time as total travel time divided by the number of UEs. In our case, we utilize the average travel time as another metric to evaluate the effectiveness of the DQN based scheme. The average travel time indicates the efficacy of the path discovery process. Recall from Section VI-B that we set a reward threshold $\varepsilon = -180$. We define path discovery failure as the condition where the total rewards for the path discovery are less than ε. Some UEs cannot find the destination at the initial stage, since the model has not obtained sufficient learning experience. Through the training process, the speed of path discovery should become faster over time until convergence. Thus, evaluating the average travel time can measure the performance of the DQN based scheme.

Average Waiting Time: In our case, we leverage the average waiting time to evaluate the effectiveness of the DQN model. The waiting time refers to the time interval from when one process is submitted to the ready queue to the time when the process is executed by the CPU. In our experiment, we define the average waiting time as the total waiting time divided by the number of tasks, represented by $AWT = \frac{\sum(T_{execute} - T_{submit})}{Tasks}$, where $T_{execute}$ is the time of a task that is executed by the computing node and T_{submit} is the time the task was submitted to the computing node.

Average Energy Efficiency: Energy efficiency is a representative metric in wireless communications, which is defined by the ratio of the achievable sum-rate of the users to the total power consumption [39]. In our case, we use the average energy efficiency to evaluate the efficacy of our scheme. In the evaluation, average energy efficiency represents the mean value of total data transmission rate per Joule for UEs. The higher average energy efficiecy indicates higher performance in energy efficiency. The average energy efficiency can be represented by $AEE \triangleq \frac{R}{(P_t + P_c)\cdot N}$, where $P_t \triangleq \sum_{n=1}^{N} a_n$ and a_n is the n^{th} user's power allocation coefficient.

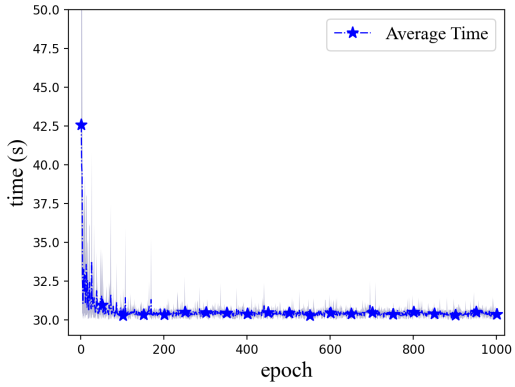
B. Evaluation Results

In this subsection, we present our evaluation results in detail.

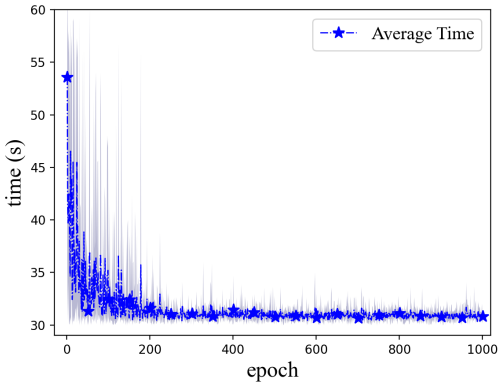
1) Average Reward and Travel Time: Figs. 5 and 6 show the average reward and average travel time for the four UE group size cases. Each experiment is run 20 times for each case, and the data displayed is the average of the 20 runs. The blue line with the star mark shows the mean value of all the experiments. The light blue area shows the maximal and minimal values of the experiment.

Fig. 5 illustrates the average reward for the four cases (varying numbers of UEs). These figures show the total rewards in each path discovery process. Note that we defined a positive reward value of reaching the destination and a negative reward for other situations (e.g., cannot reach to the destination, using larger power, longer waiting time). Thus, with the training of the DQN model, the total reward value becomes larger and approaches the maximal reward. Fig. 5(a) shows the results of the case with 5 UEs. This clearly has the fastest convergence speed, achieving the maximal reward in the shortest time. Specifically, it reaches the maximal reward at 150 epochs and maintains the reward until the experiment is complete, with only small fluctuations. Fig. 5(d) shows the results of the 100 UEs case. Compared to the 5 UEs case, the convergence speed of the 100 UEs case is slower. Nonetheless, the tendency of the rewards is monotonically increasing, achieving a relatively high reward after 700 epochs. Furthermore, the performance of the 25 UEs and 50 UEs cases are between the 5 UEs and 100 UEs cases, as expected. The evaluation shows the convergence of the proposed DQN based scheme.

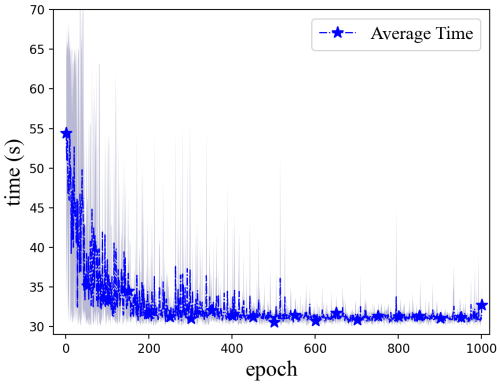
Similarly, Fig. 6 shows the average travel time for four cases (i.e., 5, 25, 50, and 100 UEs). Since the DQN model discovers the path from the start location to the destination for each UE, the performance of the DQN based scheme directly affects the time taken to reach the destination. Thus, at the beginning of the experiment, the performance of the DQN based scheme is not good, indicating that it takes UEs longer to reach their destinations. When more experience is accumulated by the DQN model, the travel time for each UE is significantly reduced. Particularly, in Fig. 6(a), which shows the results of the 5 UEs case, the average travel time falls off quite rapidly, indicating that the DQN based scheme can quickly obtain positive rewards and enable the UEs to reach their destinations. Furthermore, Fig. 6(d) shows the results of the case with 100 UEs. Here, the average travel time is also reduced quite rapidly, but still requires more training and has relatively large fluctuations in comparison to the scenarios with fewer UEs. There are two main reasons for this. First, more UEs require more bandwidth resources and are more likely to encounter packet collisions when they switch subchannels. To avoid network collisions, the UE may travel a longer distance to avoid switching between subchannels. Second, similar to the



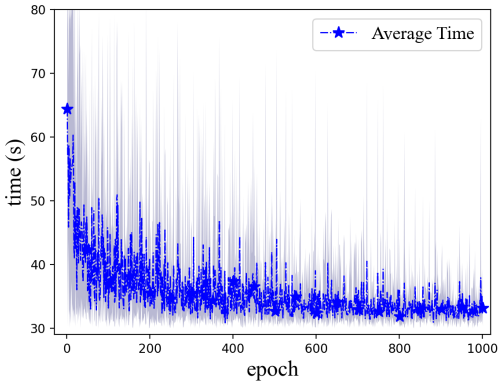
(a) 5 UEs



(b) 25 UEs



(c) 50 UEs



(d) 100 UEs

Fig. 6. The average travel time for all UEs

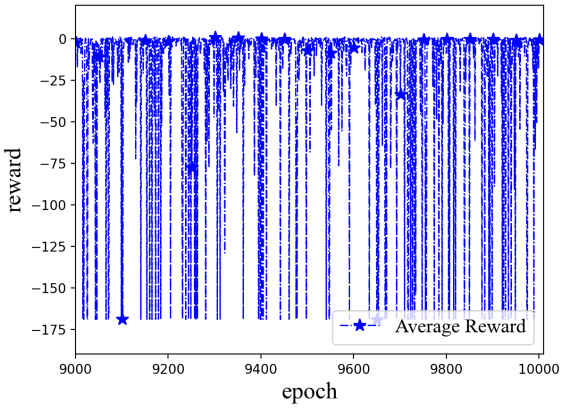


Fig. 7. Rewards for leveraging Q-learning

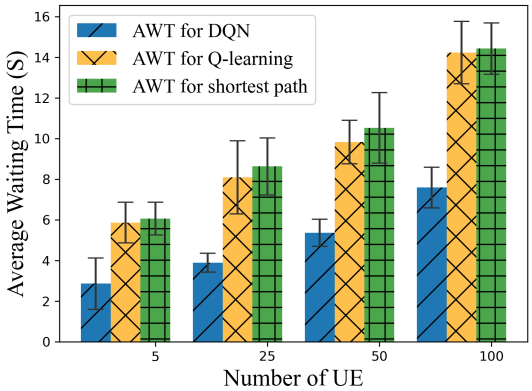


Fig. 8. Average Waiting Time

relationship between the 5 UEs and 100 UEs cases for average reward, the convergence speed of the DQN model is slower when there are more UEs, leading to a longer time for the path discovery. In addition, as the tasks are randomly generated, the DQN model may not traverse all possibilities for tasks, which leads to fluctuation in the experiment. Nonetheless, even in the 100 UEs case, the DQN model is convergent, indicating that our proposed DQN scheme can find the optimal solution for allocating the network resources.

2) Performance Comparison: We now compare the performance between our DQN based scheme and the standard Q-learning based scheme. We implement standard Q-learning to solve the resource allocation problem in the proposed scenario. Because the state-action space is quite large, we evaluate whether the Q-learning model can converge at all, given that it is necessary for its operation. Fig. 7 shows the rewards of

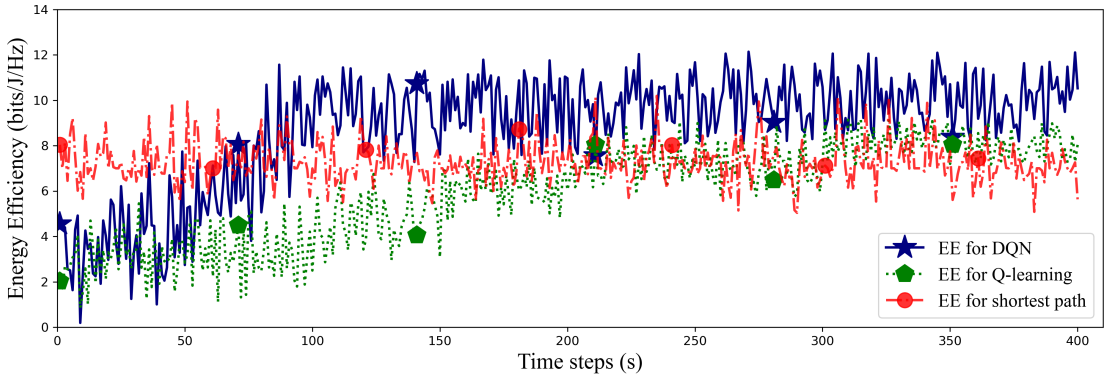


Fig. 9. Average Energy Efficiency

using Q-learning in 100 UEs case. Compared to the proposed DQN based scheme, the standard Q-learning based scheme has no obvious convergence in 10,000 epochs. Here, we select the reward value from 9000 epochs to 10,000 epochs. The results clearly show that the standard Q-learning based does not converge to find a global solution. Thus, to compare the performance with our scheme, we leverage DQN to find optimal solutions for each UE, which does not consider the interference between UEs.

In addition to our evaluation of the DQN model alone, we leverage both the standard Q-learning and the shortest path based schemes as baselines for comparison, considering the average waiting time and average energy efficiency as metrics. Fig. 8 illustrates the comparison results of the average waiting time between the DQN based scheme and the baseline schemes. Specifically, utilizing the well-trained DQN model, the Q-learning model, and the shortest path algorithm control UEs to discover and reach the destination in all the different cases (i.e., 5, 25, 50, and 100 UEs). Meanwhile, we record the average waiting time data and compare the performance. In Fig. 8, the blue bars with single slashes indicate the performance of our DQN based scheme, the yellow bars with crossed slashes indicate the performance of the standard Q-learning based scheme, and the green bars with the grid pattern indicate the performance of the shortest path based scheme. We execute the experiments 20 times for each case, taking 95 % as confidence interval to provide the error bars on the chart. The evaluation results show that our DQN based scheme achieves the smallest average waiting time in all schemes. In addition, compared to the standard Q-learning and the shortest path schemes, the average waiting time of our DQN based scheme increases at a slower rate with the number of UEs, indicating that the DQN based scheme significantly improves the network resource allocation performance of the system. As the standard Q-learning only finds the solution for each UE based on its local information without considering interactions with other UEs, the performance of the average waiting time is similar to the shortest path algorithm.

Finally, we compare the average energy efficiency of our DQN based scheme, the standard Q-learning based scheme, and the shortest path based scheme. Fig. 9 shows the comparison results of average energy efficiency for the three schemes over a single run. Here, we select the case with 100 UEs and execute over 400 s. The blue line with stars represents the average energy efficiency performance of our DQN based scheme, the green line with pentagon marks represent the average energy efficiency performance of the standard Q-learning scheme, and the red line with dots represents the shortest path based scheme. In the beginning, the average energy efficiency performance of our DQN based scheme and the standard Q-learning based scheme are worse than the shortest path based scheme, since our DQN model and standard Q-learning have not yet obtained sufficient learning experience from the training process. In this period, the DQN model obtains positive rewards faster than the standard Q-learning scenario. In other words, the DQN based scheme has a faster learning speed than the standard Q-learning scheme. In addition, the performance of our DQN model reaches and then outperforms the shortest path based scheme at around 100 s time step, while the Q-learning based scheme takes more than twice as long to outperform the shortest path scheme at around 220 s. Also, the standard Q-learning scheme can only obtain purely local optimal solutions for individual UEs, and its overall performance is worse than that of the DQN based scheme, which considers the interactions among UEs. The average energy efficiency performance of the standard Q-learning is similar to the shortest path based scheme, only marginally better at 300 s.

Overall, based on our evaluation results, our proposed DQN scheme can improve both the efficiency of network resources and energy use in the investigated IIoT system. For the standard Q-learning scheme, as it cannot handle a large state-action space, the derived solution for each UE only considers one UE at a time without considering the interactions among UEs. To this end, the performance of the standard Q-learning scheme is similar to that of the shortest path based scheme.

VIII. Final Remarks

In this paper, we addressed the resource allocation problem in the IIoT system. Particularly, we first introduced the problem space of resource allocation problems in IIoT systems. We then defined a representative smart warehouse IIoT

scenario, in which a number of robots carry packages from different sources to destinations, which communicate through a centrally located base station in the warehouse. We proposed a DQN based scheme to carry out efficient bandwidth and energy resource allocation. We also designed a simulator and implemented the DQN based scheme in Python. Our extensive experimental results indicate that our proposed DQN based scheme can improve both bandwidth utilization and energy efficiency in the IIoT environment, in comparison with two representative baseline schemes. As an extension of this study, we plan to apply the proposed DQN scheme to other IIoT systems in the future and address other resource management issues.

REFERENCES

[1] H. Xu, W. Yu, D. Griffith, and N. Golmie, "A survey on industrial Internet of things: A cyber-physical systems perspective," *IEEE Access*, vol. 6, pp. 78 238–78 259, 2018.

[2] W. Yu, F. Liang, X. He, W. G. Hatcher, C. Lu, J. Lin, and X. Yang, "A survey on the edge computing for the Internet of things," *IEEE Access*, vol. 6, pp. 6900–6919, 2018.

[3] F. Liang, W. Yu, X. Liu, D. Griffith, and N. Golmie, "Toward edge-based deep learning in industrial Internet of things," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4329–4341, 2020.

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

[5] Y. Gu, Z. Chang, M. Pan, L. Song, and Z. Han, "Joint radio and computational resource allocation in IoT fog computing," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 8, pp. 7475–7484, 2018.

[6] A. Sutagundar and S. B. Shahapur, "Development of fog based dynamic resource allocation and pricing model in IoT," in *2018 Second International Conference on Green Computing and Internet of Things (ICGCIoT)*, 2018, pp. 349–354.

[7] S. F. Abedin, M. G. R. Alam, S. M. A. Kazmi, N. H. Tran, D. Niyato, and C. S. Hong, "Resource allocation for ultra-reliable and enhanced mobile broadband IoT applications in fog network," *IEEE Transactions on Communications*, vol. 67, no. 1, pp. 489–502, 2019.

[8] J. Chen, Z. Gao, and Y. Xu, "Opportunistic spectrum access with limited feedback in unknown dynamic environment: a multi-agent learning approach," in *The 2014 5th International Conference on Game Theory for Networks*, 2014, pp. 1–6.

[9] E. Semsar-Kazerooni and K. Khorasani, "Multi-agent team cooperation: A game theory approach," *Automatica*, vol. 45, no. 10, pp. 2205–2213, 2009.

[10] D. Pandey and P. Pandey, "Approximate Q-learning: An introduction," in *2010 Second International Conference on Machine Learning and Computing*, 2010, pp. 317–320.

[11] C. Qiu, X. Wang, H. Yao, J. Du, F. R. Yu, and S. Guo, "Networking integrated cloud-edge-end in IoT: A blockchain-assisted collective Q-learning approach," *IEEE Internet of Things Journal*, pp. 1–1, 2020.

[12] G. Lample and D. S. Chaplot, "Playing FPS games with deep reinforcement learning," *arXiv preprint arXiv:1609.05521*, 2016.

[13] M. Ghobakhloo, "The future of manufacturing industry: A strategic roadmap toward Industry 4.0," *Journal of Manufacturing Technology Management*, vol. 29, no. 6, pp. 910–936, 2018.

[14] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. Soong, and J. C. Zhang, "What will 5G be?" *IEEE Journal on selected areas in communications*, vol. 32, no. 6, pp. 1065–1082, 2014.

[15] H. Kim, Y.-G. Lim, C.-B. Chae, and D. Hong, "Multiple access for 5G new radio: Categorization, evaluation, and challenges," *arXiv preprint arXiv:1703.09042*, 2017.

[16] J. Zhao, Y. Liu, K. K. Chai, Y. Chen, and M. Elkashlan, "Joint subchannel and power allocation for NOMA enhanced D2D communications," *IEEE Transactions on Communications*, vol. 65, no. 11, pp. 5081–5094, 2017.

[17] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[18] C. Qiu, Y. Hu, Y. Chen, and B. Zeng, "Deep deterministic policy gradient (DDPG)-based energy harvesting wireless communications," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8577–8588, 2019.

[19] L. Engstrom, A. Ilyas, S. Santurkar, D. Tsipras, F. Janoos, L. Rudolph, and A. Madry, "Implementation matters in Deep RL: a case study on PPO and TRPO," in *International Conference on Learning Representations*, 2019.

[20] A. Choudhary, "A hands-on introduction to deep q-learning using openai gym in python," *Retrived from https://www. analyticsvidhya. com/blog/2019/04/introduction-deep-q-learningpython*, 2019.

[21] L. Zhuhadar, E. Thrasher, S. Marklin, and P. O. de Pablos, "The next wave of innovationreview of smart cities intelligent operation systems," *Computers in Human Behavior*, vol. 66, pp. 273–281, 2017.

[22] L. Li, S. Li, and S. Zhao, "Qos-aware scheduling of services-oriented internet of things," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1497–1505, 2014.

[23] F. Al-Turjman, M. Z. Hasan, and H. Al-Rizzo, "Task scheduling in cloud-based survivability applications using swarm optimization in iot," *Transactions on Emerging Telecommunications Technologies*, p. e3539, 2018.

[24] S. Basu, M. Karuppiah, K. Selvakumar, K.-C. Li, S. H. Islam, M. M. Hassan, and M. Z. A. Bhuiyan, "An intelligent/cognitive model of task scheduling for iot applications in cloud computing environment," *Future Generation Computer Systems*, vol. 88, pp. 254–261, 2018.

[25] H. Ye, G. Y. Li, and B.-H. F. Juang, "Deep reinforcement learning based resource allocation for v2v communications," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3163–3173, 2019.

[26] H. Xu, X. Liu, W. Yu, D. Griffith, and N. Golmie, "Reinforcement learning-based control and networking co-design for industrial Internet of things," *IEEE Journal on Selected Areas in Communications*, pp. 1–1, 2020.

[27] L. Liang, H. Ye, G. Yu, and G. Y. Li, "Deep-learning-based wireless resource allocation with application to vehicular networks," *Proceedings of the IEEE*, vol. 108, no. 2, pp. 341–356, 2019.

[28] M. Liu, T. Song, and G. Gui, "Deep cognitive perspective: Resource allocation for noma-based heterogeneous iot with imperfect sic," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2885–2894, 2018.

[29] Y. Sun, Y. Wang, J. Jiao, S. Wu, and Q. Zhang, "Deep learning-based long-term power allocation scheme for noma downlink system in S-IoT," *IEEE Access*, vol. 7, pp. 86 288–86 296, 2019.

[30] C. A. Floudas and P. M. Pardalos, *State of the art in global optimization: computational methods and applications*. Springer Science & Business Media, 2013, vol. 7.

[31] X. Liu, J. Cao, Y. Yang, and S. Jiang, "CPS-based smart warehouse for industry 4.0: a survey of the underlying technologies," *Computers*, vol. 7, no. 1, p. 13, 2018.

[32] F. Al Rabee, K. Davaslioglu, and R. Gitlin, "The optimum received power levels of uplink non-orthogonal multiple access (noma) signals," in *2017 IEEE 18th Wireless and Microwave Technology Conference (WAMICON)*. IEEE, 2017, pp. 1–4.

[33] S. Yan, M. Peng, and X. Cao, "A game theory approach for joint access selection and resource allocation in UAV assisted IoT communication networks," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1663–1674, 2018.

[34] M. Ficco, C. Esposito, F. Palmieri, and A. Castiglione, "A coral-reefs and game theory-based approach for optimizing elastic cloud resource allocation," *Future Generation Computer Systems*, vol. 78, pp. 343–352, 2018.

[35] A. S. Bedi and K. Rajawat, "Asynchronous incremental stochastic dual descent algorithm for network resource allocation," *IEEE Transactions on Signal Processing*, vol. 66, no. 9, pp. 2229–2244, 2018.

[36] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[37] W. Hao, M. Zeng, Z. Chu, and S. Yang, "Energy-efficient power allocation in millimeter wave massive mimo with non-orthogonal multiple access," *IEEE Wireless Communications Letters*, vol. 6, no. 6, pp. 782–785, 2017.

[38] J. Zeng, T. Lv, R. P. Liu, X. Su, M. Peng, C. Wang, and J. Mei, "Investigation on evolving single-carrier NOMA into multi-carrier NOMA in 5G," *IEEE Access*, vol. 6, pp. 48 268–48 288, 2018.

[39] F. Fang, H. Zhang, J. Cheng, and V. C. Leung, "Energy-efficient resource allocation for downlink non-orthogonal multiple access network," *IEEE Transactions on Communications*, vol. 64, no. 9, pp. 3722–3732, 2016.