IEEE ComSoc IEEE Communications Society | IEEE COMPUTER SOCIETY | IEEE Signal Processing Society | IEEE VTS Connecting the Mobile World | IEEE Transactions on Machine Learning in Communications and Networking

Received 3 October 2025; revised 16 January 2026; accepted 24 February 2026. Date of publication 3 March 2026; date of current version 27 March 2026.

The associate editor coordinating the review of this article and approving it for publication was E. V. Belmega.

Digital Object Identifier 10.1109/TMLCN.2026.3669837

Digital Twin-Driven Continual Deep Reinforcement Learning for Coexistence of Multiple Radio Access Technology IoT Links With Nonlinear Receivers

NAHED BELHADJ MOHAMED[1], GEORGES KADDOUM[1] (Senior Member, IEEE), AND MD. ZOHEB HASSAN[2] (Member, IEEE)

[1]École de Technologie Supérieure (ÉTS), Université du Québec, Montréal, QC H3C 1K3, Canada
[2]Université Laval, Quebec City, QC G1V 0A6, Canada

CORRESPONDING AUTHOR: M. Z. HASSAN (md-zoheb.hassan@gel.ulaval.ca)

This work was supported in part by Canada Research Chair Program Tier-II entitled "Toward a Novel and Intelligent Framework for the Next Generations of IoT Networks." The work of Md. Zoheb Hassan was supported in part by the Discovery Grant from the National Science and Engineering Research Council (NSERC) of Canada and in part by Fonds de recherche du Québec (FRQ).

ABSTRACT This article investigates the coexistence of downlink Internet-of-Things (IoT) links enabled by multiple radio access technologies (RATs), including long-term evolution (LTE) and 5G new radio (NR). The coexistence of multiple RAT IoT links is significantly challenged by adjacent channel interference (ACI) and hardware impairments (HWI) that arise from practical low-complexity radio-frequency front ends. To mitigate these challenges, we propose a radio resource optimization scheme that dynamically adjusts link adaptation parameters (transmit power, modulation, and coding rate) to maximize overall throughput while explicitly accounting for ACI and HWI. However, the proposed optimization is an NP-hard mixed-integer non-linear programming problem that requires global channel state information and centralized optimization, making it impractical for large-scale, dynamic multi-RAT IoT networks. To enable distributed optimization under ACI and HWI, we reformulate the problem as a Markov game and develop a multi-agent deep reinforcement learning (MADRL) framework that derives equilibrium link adaptation policies from local observations. Direct deep reinforcement learning (DRL) training in real networks, however, incurs high communication overhead and can create adverse effects due to the random explorations. To overcome these limitations, we introduce a context-aware digital twin network (DTN) that provides a safe and efficient virtual environment for training. In particular, we propose a novel DTN-empowered MADRL scheme that employs a replay memory-based continual model updating strategy, enabling policies to be learned from DT-generated experiences and periodically refined with real network data. This approach alleviates the need for frequent physical network interactions and significantly reduces communication overhead. Extensive simulations demonstrate that the proposed framework is scalable, computationally efficient, and robust in dynamic IoT environments, while outperforming 3GPP-standardized link adaptation in the presence of non-negligible ACI and HWI.

INDEX TERMS Adjacent channel interference, continual deep reinforcement learning, digital twin, hardware impairment, resource allocation.

I. INTRODUCTION

THE Internet of Things (IoT) has witnessed rapid growth in recent years, with an ever-expanding number of interconnected devices and objects. With the growth in the number of IoT devices, the demand for higher data rates, bandwidth, capacity, and throughput has increased exponentially [1]. This rapid expansion underscores the critical need for advanced multi-radio access technology

(RAT) systems—such as long-term evolution (LTE) and 5G new radio (NR)—to provide seamless, efficient, and reliable connectivity across diverse IoT applications [2].

In large-scale IoT networks, where multiple RATs operate in close proximity, strong adjacent channel interference (ACI) may occur when links operate on adjacent channels. This interference is caused by the non-linear response of the radio frequency (RF) front end, which produces distortion and intermodulation, degrading signal quality and receiver sensitivity. Notably, when IoT links employ spectrum-agile devices with wideband pre-selection filters to operate across multiple RATs and frequency bands, they may receive multiple unwanted signals from adjacent channels [3]. These interfering signals can originate from the same or different RATs and, due to the receiver's inherent non-linearity, the resulting interference can be significant—particularly in dense IoT networks, where it is non-trivial to maintain a large distance between adjacent channel transmitters and victim receivers. Besides the ACI, IoT devices frequently use low-complexity RF front ends that induce signal distortions due to hardware impairments (HWIs). These HWIs originate from the impact of the phase noise, quantization error, amplifier non-linearity, and so on [4].

Both ACI and HWI-induced distortions can considerably reduce the achievable throughput. In such scenarios, conventional model-based optimization methods experience difficulties for several reasons. First, these impairments make the optimal resource allocation in large-scale IoT networks computationally intractable (i.e., NP-hard) [5]. Second, conventional iterative optimization methods (e.g., successive convex approximation) usually require significant computation overhead, long converge time, and high energy consumption, particularly in scenarios with large numbers of IoT devices. Third, these methods frequently rely on analytical models to capture ACI- and HWI-induced distortions. However, existing models may not accurately reflect impairments in dynamic IoT systems. For example, classical Rapp [6] and Saleh [7] models are derived under idealized assumptions, including memoryless behavior, device-agnosticity, and static operating conditions [8]. In contrast, HWI and ACI in practical IoT networks are vendor- and manufacturer-specific, and may vary with device conditions such as aging, operating temperature, and hardware features. Consequently, in real-world IoT networks, optimization approaches that rely on predefined analytical impairment models while overlooking these aspects may experience severe performance degradation.

In this context, deep reinforcement learning (DRL) emerges as a promising paradigm to tackle complex and non-convex optimization in IoT networks by learning efficient resource allocation policies from environmental observations. However, pre-trained DRL models are susceptible to catastrophic forgetting when adapting to new scenarios [9]. To overcome this concern, a viable solution is continual DRL that trains the DRL model using both past and new experiences (stored in the experience replay memory), ensuring knowledge retention and adaptability. However, training such algorithms directly in real-world environments not only requires significant communication overhead, but also poses significant risks due to the potential for erroneous decisions during exploration [10]. To mitigate these challenges, a transformative solution that has recently emerged is the concept of a digital twin (DT). A digital twin network (DTN) provides a virtual replica of the physical network, enabling realistic simulations of multi-RAT IoT scenarios. DTNs can serve as controllable and trustworthy virtual environments to train DRL algorithms while eliminating the high costs of frequent interactions with the physical environment and the risk of disrupting network operations during exploration. This approach enhances practicality and performance of continual learning-based resource allocation in dynamic IoT networks.

A. RELATED WORKS

Resource optimization plays a fundamental role in mitigating interference in large-scale networks. Numerous previous studies presented various optimization approaches, including heuristic and iterative methods, aiming to tackle the resource allocation issue in various wireless communication scenarios [11]. However, due to the problem's non-convex nature, a common limitation of traditional optimization techniques is that they require numerous iterations to reach convergence. As a result, these methods are ill-suited for optimal resource allocation in large-scale IoT networks [12]. Recently, reinforcement learning (RL) has emerged as a promising technique to solve complex optimization problems [13]. In RL, the agent learns the optimal policy while interacting with the environment. Owing to its ability to interact with an unknown environment through exploitation and exploration, RL was reported to be an efficient technique to solve radio resource optimization. In [14], [15], the authors proposed an RL-based modulation and coding scheme (MCS) to maximize the spectral efficiency of cellular networks. In another relevant study [16], the authors proposed a multi-agent RL (MARL) approach for joint channel assignment and power allocation in platoon-based cellular vehicle-to-everything (C-V2X) systems.

However, since the practical wireless environment is characterized by vast and continuous state and action spaces, it remains challenging to find a suitable RL policy capable of learning the optimal mapping between environmental states and actions. DRL, which combines deep learning (DL) and RL, has gained a lot of scholarly attention in solving complicated control problems with highly-dimensional data [17]. In [18], the authors proposed an intelligent DRL-based MCS selection algorithm in cognitive heterogeneous networks. In [19], the authors suggested using a novel centralized DRL-based downlink power allocation scheme to maximize the total network throughput in a multi-cell system. Likewise, a DRL-based power allocation was proposed to enhance the sum rate in multi-user cellular networks [20]. Furthermore, in [21], a distributed DRL-based spectrum

access approach capable of maximizing the network utility was proposed. In addition, a DRL-based energy-efficient link adaptation algorithm was proposed to jointly determine the transmitted power level and MCS [22]. Likewise, a multi-agent DRL (MADRL) framework was introduced for joint power allocation and MCS selection to maximize throughput in a downlink cellular network [23].

However, while the studies briefly reviewed above showcase the potential of DRL in addressing resource allocation challenges, there remain significant barriers in deploying DRL-based solutions in real-world systems. Specifically, during the training phase, DRL algorithms explore the environment to optimize their decision making processes. However, incorrect decisions during this exploration can lead to harmful consequences when training directly in the physical environment. To overcome this limitation, in the present study, we propose using the DT technology to create a virtual environment that accurately replicates the physical system. This virtual pre-validation environment allows DRL algorithms to train and explore in a safe and cost-effective manner, ensuring robust performance before deployment in real-world networks [24]. Recent studies have investigated various roles of DT in wireless networks, including learning [25], proactive interference management [26], [27], and RF channel twin modeling [28]. Building on these advancements, this study leverages DTNs to optimize radio resource allocation in multi-RAT IoT networks.

B. CONTRIBUTIONS AND PAPER ORGANIZATION

Emerging downlink IoT applications, such as real-time video surveillance, industrial automation, and augmented reality, require high downlink data rates and stringent reliability. Meeting these requirements with legacy systems like narrowband (NB)-IoT or a single RAT is challenging, especially for dense IoT environments. Accordingly, it becomes imperative to integrate multi-RATs, as doing so enables flexible spectrum usage, higher aggregate throughput, and robust connectivity across heterogeneous IoT scenarios.

In this paper, we investigate the coexistence of multi-RAT-enabled IoT systems, a key enabler of next-generation Internet-of-Everything (IoE) networks that must support diverse quality-of-service (QoS) demands and an exponentially growing number of IoT links [29]. Recent studies demonstrated the importance ensuring coexistence between multiple RATs in the emerging cellular bands [30], [31]. Nevertheless, managing the large-scale coexistence of multi-RAT IoT links is confronted by the following challenges: (**C1**) insufficient physical distance between links operating on adjacent channels, particularly when unlicensed services are involved; (**C2**) the deleterious impact of ACI and HWI resulting from non-linear hardware commonly used in cost-constrained IoT devices; and (**C3**) the absence of centralized mechanisms and global network information required to optimize the transmission parameters of distributed multi-RAT IoT links at scale.

To address these challenges, in this work, we propose a DTN-enhanced and MADRL-empowered decentralized radio resource optimization framework to facilitate the harmonious coexistence of multi-RAT IoT links. In the proposed framework, each IoT link determines its suitable link adaptation parameters—namely, transmit power, modulation, and coding rate—based on its local observation. To the best of our knowledge, our study is the first to consider both ACI and HWI-induced distortions in the radio resource optimization of multi-RAT IoT systems. The specific contributions of this work are summarized below.

- **Design of DTN of multi-RAT IoT networks:** A DTN for a multi-RAT IoT system is designed to enhance resource allocation in dynamic and heterogeneous environments. Its key features include: (1) network topology modeling; (2) awareness of time-varying fading channels in multi-RAT IoT networks; and (3) incorporation of non-linear device characteristics to model ACI- and HWI-induced distortions.
- **Radio resource management (RRM) problem formulation:** An optimization problem is formulated to maximize the long-term sum throughput in downlink multi-RAT IoT networks by optimally selecting link adaptation parameters (power level, modulation order, and coding rate) for the coexisting IoT links. Solving this RRM problem is challenged by: (1) computational intractability due to its NP-hardness and mixed-integer nonlinear programming (MINLP) structure, and (2) the requirement of global channel state information (CSI),[1] which is impractical to acquire in real time, particularly in large-scale networks [32].
- **Distributed resource allocation policy:** To address these challenges and enable a distributed solution, the RRM problem is reformulated as a Markov game, and a MADRL approach is employed to learn the equilibrium policy. In this framework, each IoT link uses a trained DRL model to select appropriate link adaptation parameters based solely on its local state,[2] without requiring global network information. This design makes the solution well-suited for large-scale multi-RAT IoT networks with partial observability.
- **DTN-assisted continual DRL framework:** Since training a DRL model directly in live networks is costly and impractical due to large exploration overhead, we exploit DTN as a reliable virtual environment for training and online updating of the DRL model. We propose a DTN-empowered, replay memory–based continual learning framework that first learns an initial policy from DT-generated experiences and then periodically refines the policy using experiences collected from the physical network. This approach offers the following key advantages: (i) fast convergence; (ii) reduced communication overhead; and (iii) continual adaptation to evolving network conditions.

[1]Global CSI comprises two components: (i) the individual CSI of each IoT link and (ii) the CSI of adjacent interfering links.

[2]The proposed DRL-based link-adaptation framework is generic and can be applied to any wireless system where standard-form of channel quality feedback (such as CSI and signal-to-interference-plus-noise ratio (SINR)) is available.

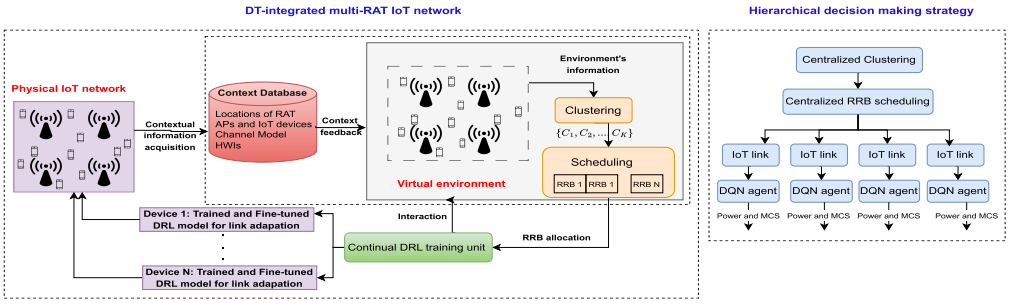


Fig. 1. Proposed DTN-empowered resource allocation framework for multiple RAT APs IoT networks.

- **Extensive performance evaluation:** Extensive simulations are conducted to evaluate the performance of the proposed solution under ACI and HWI-induced distortions. Simulation results demonstrate (a) the robustness and scalability of the proposed DTN-empowered continual-DRL framework in dynamic networks with ACI and HWI uncertainties, and (b) that it achieves up to 114.16% higher sum throughput than the 3GPP-standardized channel quality indicator (CQI)-based link adaptation scheme[3] in dense IoT networks under high HWI levels.

The remainder of this paper is organized as follows. Section II presents an overview of the system model and the proposed framework. Section III provides the description of DTN. Section IV formulates the RRM problem. Section V details the proposed framework, while Section VI discusses the simulation results. Finally, Section VII concludes the paper.

II. SYSTEM MODEL

A. SYSTEM OVERVIEW

As shown in Fig. 1, we consider a downlink IoT network comprising multiple RAT access points (APs) and uniformly distributed IoT devices. Let $\mathcal{K} = \{1, 2, \cdots, K\}$ be the set of all RAT APs, $\mathcal{N} = \{1, 2, \cdots, N\}$ be the set of IoT devices, and $\mathcal{R} = \{1, 2, \cdots, R\}$ be the set of radio resource blocks (RRBs), where each RRB provides the minimum granularity of bandwidth allocated to an IoT link. In the proposed system, IoT devices are clustered with their nearest RAT APs based on proximity. Each RAT AP simultaneously transmits data to its associated IoT devices using the orthogonal frequency-division multiple access (OFDMA) technique. Meanwhile, a centralized network controller assigns an orthogonal RRB to each device[4] by proportional fairness (PF) RRB scheduling algorithm, thus ensuring efficient resource distribution and maintaining fairness across the network. Owing to these considerations, co-channel interference among devices—both within a cluster and across clusters—remains negligible.

[3]In this work, 3GPP-specified SINR–MCS mapping is used solely as a representative example of a standardized link-adaptation scheme. Our proposed learning framework itself is not restricted to 3GPP systems and remains applicable to any deployment where suitable CQIs can be obtained.

[4]Without the loss of generality, throughout this paper, we use the terms device" and link" interchangeably.

Importantly, the devices associated with different RAT APs can be scheduled to adjacent RRBs. However, due to the low-cost receivers that are typically employed in IoT devices, the system is susceptible to HWI and ACI. Specifically, non-ideal filters cause undesired signals to leak into adjacent frequency bands, resulting in ACI for devices allocated to neighboring RRBs. While ACI and HWI explicitly depend on the selected transmit power level, because of the fundamental power–modulation–coding trade-off, they also implicitly depend on the chosen MCS schemes. On one hand, increasing transmit power and signal-to-noise ratio enables the use of higher-order MCS at the cost of amplifying HWI-induced distortion and ACI toward neighboring IoT links. On the other hand, decreasing transmit power minimizes the impact of non-linear impairments, but reduces throughput, as only lower-order MCS can be supported to maintain transmission reliability. In essence, there exists an inherent relationship between link adaptation parameters and the impairments caused by ACI and HWI. In the present study, we use this relationship to optimize link adaptation and enable harmonious coexistence among multi-RAT IoT links.

B. PROPOSED FRAMEWORK

This section provides an overview of the proposed RRM framework, which comprises the following three key components: (a) a context database; (b) a digital twin network; and (c) a continual DRL training unit. Fig. 1 illustrates these components and the hierarchical resource allocation decision making strategy.

1) CONTEXT DATABASE

The context database gathers data from the real-world environment and stores this data to support the creation of a virtual representation of the physical network. To maintain the relevance and accuracy of the virtual environment, the contextual data are categorized into the following two types:

- Static contextual data: Environmental parameters such as the radio propagation model, RAT AP information (e.g., number, location, height, maximum/minimum transmit power), frequency band allocation, and a theoretical HWI model.

• Dynamic contextual data: Network parameters such as RRB allocation, IoT device locations, and individual channel measurements of IoT links.[5]

The static contextual data remain fixed over extended periods of time and are collected only once during the DT model construction phase. In contrast, dynamic contextual data is collected periodically through various key performance measurement (KPM) reports, widely available in wireless standards. Importantly, network parameters are periodically collected at RAT APs over standard control feedback channels and provided as dynamic contexts to the DTN once in every continual learning period (or DRL refinement period) spanning over several time slots (TSs).

2) DIGITAL TWIN NETWORK

The DTN serves as a virtual emulation environment designed to train DRL algorithms in a controlled and reliable setting. The DTN developed in this study maintains a virtual network topology with a digital abstraction of multi-RAT APs, IoT devices, and radio resources. It also incorporates a baseline representation of time-varying fading channels and approximate models of HWI and ACI (given in Section III-B.1). These initial models, while simplified, provide a starting point for generating realistic datasets and enabling the training of DRL-based RRM policies. Importantly, the DTN leverages continual learning to refine the DRL agent's policy with observations periodically collected from the real deployment, rather than relying solely on theoretical models. Such an adaptive learning mechanism ensures that the trained policies remain resilient even when the actual device nonlinearities and interference patterns deviate from the initial assumptions. The DTN integrates virtual radio resource control functionalities such as clustering, scheduling, and data transmission with adaptive modulation (AM) and coding selection. In creating the DTN, we relied on the following assumptions. **A1:** Each device is aware of its local CSI. **A2:** We consider a time-slotted fading channel where the small-scale gain remains constant in each TS and changes from one TS to the next. Besides, we assume that large-scale fading remains the same over T TSs. **A3:** Each device is associated with only one RAT AP. **A4:** For the direct collection of channel-specific datasets at the DTN from the physical network, as well as for transferring the trained DRL model from the DTN to the physical network, following the DT literature [26], [27], and [33], we assume the existence of trustworthy and high-bandwidth physical-to-digital (P2D) and digital-to-physical (D2P) feedback links between the multi-RAT APs in the physical network and the DT. These P2D/D2P links are also used to disseminate updated deep Q-network (DQN) parameters during continual refinement, while no inter-link communication or synchronization is required during online operation. These assumptions provide a foundation for accurate modeling and simulating the physical IoT network within the DTN. Further detail on the design of this DTN is provided in Section III.

[5]We assume that each IoT node estimates its own CSI and reports it to the RAT AP over an uplink control feedback channel, consistently with off-the-shelf wireless standards.

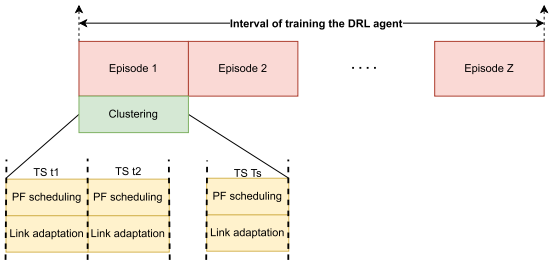


Fig. 2. Timing diagram.

3) CONTINUAL DRL TRAINING UNIT

This unit is responsible for delivering a trained and fine-tuned DRL agent for deployment in the coexisting multi-RAT IoT links. Each link operates as an independent agent and makes resource allocation decisions based on local observations. Given the dynamic nature of IoT networks with uncertain channels, ACI, and HWI, the DRL agent is continuously trained through interactions with both the DTN and the physical environment. The continual learning strategy leverages replay memory–based periodic fine-tuning that (a) enhances model stability by storing and reusing past experiences and (b) updates the model with new data to adapt to evolving network dynamics. The continual DRL algorithm is discussed in detail in Section V.

To support adaptive decision making, we divide the mission duration into several episodes, each consisting of T_s TSs. Device clustering around multi-RAT APs is performed once at the beginning of each episode based on device proximity to the APs, whereas PF-based RRB scheduling and DRL-based link adaptation are executed at every TS. Fig. 2 illustrates a representative timing diagram showing this periodic coordination across clustering, scheduling, and link adaptation.

III. DIGITAL TWIN NETWORK

This section provides a comprehensive overview of designing the DTN for multi-RAT IoT systems, including the overall network environment and the modeling parameters.

A. ENVIRONMENT OF THE DIGITAL TWIN

1) RAT APS

Each RAT AP has a coverage radius of 300 m, with a distance of 600 m between neighboring RAT APs. Following 3GPP urban micro (UMi) specifications, the RAT APs' heights are set to 10 m. Each RAT AP operates in the 6.5 GHz [34] frequency band. In addition, each RAT AP contains nonlinear components contributing to HWI.

2) IOT DEVICES

IoT devices, which are uniformly distributed within the coverage area, experience varying path losses based on their proximity to RAT APs and the network's dynamic conditions. Following 3GPP UMi specifications, their heights are set to 1.5 m. Similarly to the RAT APs, IoT devices also contain nonlinear components that result in HWI.

3) TIME VARYING CHANNEL

A fully synchronized, time-slotted system with a slot duration of T_s is considered. The channel gain accounts for both small-scale and large-scale fading. Small-scale fading remains constant within each TS and varies between TSs, modeled using the Jakes fading model. The TS interval, T_s, is set to 20 ms, representing the channel coherence time to capture fading effects. In its turn, large-scale fading incorporates path loss and shadowing to reflect the overall propagation environment, following the 3GPP UMi path loss model.

4) DEVICE CLUSTERING

Devices are clustered to the nearest RAT AP based on their proximity. The number of devices associated with each RAT AP can vary across RAT APs and from deployment to deployment. Let $\{\mathcal{C}_1, \mathcal{C}_2, \cdots, \mathcal{C}_K\}$ represent the set of clusters formed.

5) RRB-DEVICE SCHEDULING

Following [35], the PF algorithm is used to schedule the device to the appropriate available RRB. The PF scheduling algorithm is discussed in Section III-C.3.

B. DIGITAL TWIN MODELING PARAMETERS

1) CHANNEL MODEL

We denote the location of the n-th devices in TS t by (x_n, y_n, H_n), $\forall n \in \mathcal{N}$, and the location of the RAT AP by (x_k, y_k, H_k). In TS t, the 2D and 3D distances between the RAT AP and the n-th device are expressed, respectively, as shown in Eqs. (1)-(2).

$$d_{2D} = \sqrt{(x_n(t) - x_k)^2 + (y_n(t) - y_k)^2}, \tag{1}$$

and

$$d_{3D} = \sqrt{(x_n(t) - x_k)^2 + (y_n(t) - y_k)^2 + (H_n - H_k)^2}. \tag{2}$$

The downlink channel gain from the k-th RAT AP to its associated device is modeled following [32], where each channel is subjected to large-scale fading $\beta_{n,k}$ and small-scale block Rayleigh fading $h_{n,k}$. The corresponding channel gain is $g^t_{n,k} = |h_{n,k}(t)|^2 \beta_{n,k}$. According to the Jakes fading model [36], $h_{n,k}$ can be expressed as a first-order complex Gauss-Markov process (see Eq. (3)).

$$h_{n,k} = \rho h_{n,k}(t-1) + \sigma, \tag{3}$$

where ρ is the correlation coefficient between two TSs, σ is a random variable with a distribution $\sigma \sim \mathcal{CN}(0, 1-\rho^2)$, and $h(0)$ is a random variable with a normal distribution $h(0) \sim \mathcal{CN}(0,1)$. The large-scale fading component $\beta_{n,k} = 10^{\frac{-(PL+\sigma_s)}{10}}$ depends on both path loss PL and shadowing σ_s. In the present study, we consider the path loss model proposed by 3GPP [37], specifically focusing on the UMi scenario. The average path loss (in dB) between the RAT AP and the n-th device is given by Eq. (4).

$$PL = Pr_{LoS}\, PL_{LoS} + Pr_{NLoS}\, PL_{NLoS}, \tag{4}$$

where Pr_{LoS} and Pr_{NLoS} are the probabilities of having a line-of-sight (LoS) and non-line-of-sight (NLoS) between the RAT AP and the n-th device, respectively. PL_{LoS} and PL_{NLoS} represent the path loss between the RAT AP and the n-th device for the LoS and NLoS links (in dB), respectively. These values were computed based on [37, Table 7.4.1-1]. The PL_{LoS} is expressed as shown in Eq. (5).

$$PL_{LoS} = \begin{cases} PL_1 & 10\text{m} \le d_{2D} \le d_{BP} \\ PL_2 & d_{BP} \le d_{2D} \le 5\text{km} \end{cases} \tag{5}$$

where PL_1 and PL_2 are defined as follows (see Eqs. (6)-(7)).

$$PL_1 = 32.4 + 21\ log_{10}(d_{3D}) + 20\ log_{10}(f_c), \tag{6}$$

and

$$PL_2 = 32.4 + 40\ log_{10}(d_{3D}) + 20\ log_{10}(f_c) + \eta_{LoS}. \tag{7}$$

The PL_{NLoS} is given by Eq. (8).

$$PL_{NLoS} = \max(22.4 + 35.3\ log_{10}(d_{3D}) + 21.3\ log_{10}(f_c) + \eta_{NLoS}, PL_{LoS}), \tag{8}$$

where f_c denotes the carrier frequency, η_{LOS} and η_{NLoS} represent additional attenuation factors due to the LoS and NLoS connections, respectively. Parameter d_{BP} represents the breakpoint distance, expressed as shown in Eq. (9).

$$d_{BP} = \frac{4}{c}(H-1)(H_n-1)f_c, \tag{9}$$

where c is the speed of light. From [37, Table 7.4.2-1], we obtain $Pr_{LoS} = \frac{18}{d} + e^{\left(\frac{-d_{2D}}{36}\right)}\left(1 - \frac{18}{d_{2D}}\right)$ and $Pr_{NLoS} = 1 - Pr_{LoS}$. Of note, ensuring analysis is also valid for other channel fading and path loss models.

2) ACI AND HWI MODEL IN DT

We consider the effects of both ACI and HWI-induced distortions on the system performance.

- **ACI:** After scheduling each device to its appropriate RRB, we consider the ACI as previously proposed in [38]. Specifically, the ACI from the first and second adjacent channels dominate as compared to the remaining adjacent channels. For instance, a device allocated to RRB_r will experience ACI from devices assigned to RRB_{r-2}, RRB_{r-1}, RRB_{r+1}, and RRB_{r+2}. The ACI power is expressed as shown in Eq. (10).

$$P_{ACI} = \sum_{l=1}^{L} \frac{g^t_l P_l}{A_l}, \tag{10}$$

where L is the total number of adjacent channels, P_l is the power received from the l-th adjacent channel, g_l^t denotes the channel gain for the l-th adjacent channel at time t, and A_l is the ACI ratio that quantifies the interference attenuation for the l-th adjacent channel [38, Table 10].

- **Linear distortion:** In the present study, we analyze the impact of linear distortion arising from imperfections in the power amplifier. Following [39] and [40], the distortion noise, denoted as z, is modeled as $z \sim \mathcal{CN}\left(0, (\sigma_t^2 + \sigma_r^2)P_{total,k}\right)$, where $P_{total,k}$ denotes the total transmit power of the k-th RAT AP to its associated IoT devices [5]. Here, $\sigma_t^2 \geq 0$ and $\sigma_r^2 \geq 0$ denote the degree of HWI at the transmitter and receiver, respectively. Specifically, they represent error vector magnitude (EVM) of RF transceivers, which quantifies the ratio of the average distortion magnitude to the average reference signal magnitude [41].

3) ACI AND HWI MODEL IN PHYSICAL NETWORK

Due to factors such as hardware nonlinearity, temperature drift, aging, and manufacturing tolerances, the HWI and ACI observed in practice diverge from those modeled in the DTN. Since these effects cannot be precisely captured by a static theoretical model, a mismatch between the impairments in the DTN and those in the physical environment is both inevitable and unknown a priori. Without loss of generality, we model these deviations using linear functions and introduce a scalar parameter ϕ to represent them. Specifically, ϕ quantifies the relative differences between the digital and physical representations of ACI power and HWI-induced distortion with respect to the theoretically modeled values. The ACI power and HWI-induced distortion noise in the physical environment are defined as shown in Eqs. (11)-(12)

$$P_{ACI}^{\text{PHY}} = \sum_{l=1}^{L} \frac{g_l^t P_l}{A_k}(1+\phi), \tag{11}$$

and

$$z^{\text{PHY}} \sim \mathcal{CN}\left(0, (\sigma_t^2 + \sigma_r^2)P_{total,k}(1+\phi)\right), \tag{12}$$

respectively. Evidently, small and large values of this scalar parameter represent scenarios where the ACI and HWI in the physical network are more and less severe, respectively. Importantly, our continual DRL framework is agnostic to the choice of deviation model, as the link-adaptation DRL agent is periodically refined using experiences collected from the physical network. Consequently, our framework is also applicable to other impairment deviation models.

C. VIRTUAL RRM FUNCTIONALITY

1) DATA TRANSMISSION MODEL AND SIGNAL-TO-INTERFERENCE-PLUS-NOISE RATIO

In what follows, we derive the SINR while taking into account the deleterious impact of both ACI and HWI-induced distortions. Without the loss of generality, we consider that the n-th IoT device is associated with the k-th RAT AP. The received baseband signal from the k-th RAT AP to the n-th IoT device is expressed by Eq. (13).

$$\mathrm{r_n} = \sqrt{g_{n,k}^t}\left(\sqrt{P_{n,k}^t}x_n + z\right) + \sum_{l=1}^{L}\sqrt{\frac{g_l^t P_l}{A_l}}x_l + n_x, \tag{13}$$

where x_n is baseband modulated symbol with $\mathbb{E}[|x_n|^2] = 1$, $P_{n,k}^t$ denotes the transmit power at t TS, z denotes the signal distortion caused by HWIs [42], [43], x_l refers to baseband symbol of the l-th interfering link with $\mathbb{E}[|x_l|^2] = 1$, and $n_x \sim \mathcal{N}(0, \sigma^2)$ is the additive white Gaussian noise (AWGN) with variance σ^2. Using both the ACI and HWI-induced distortions models, the downlink SINR expression for the n-th device is expressed as shown in Eq. (14).

$$\mathrm{sinr_{n,k}^t} = \frac{P_{n,k}^t\ g_{n,k}^t}{g_{n,k}^t(\sigma_t^2 + \sigma_r^2)P_{total,k} + P_{ACI} + \sigma^2}. \tag{14}$$

Of note, when the SINR for IoT links is computed in the real system, rather than in the DT (i.e., based on measurements from the physical environment), the expression in Eq. (14) incorporates real-world parameter such as $P_{\text{ACI}}^{\text{PHY}}$ and z^{PHY}.

2) MODULATION AND CODING RATE

In this study, we consider AM with error-correcting codes, where $\mathcal{M}$ modulation schemes, either Gray-coded phase-shift keying (PSK) or quadrature amplitude modulation (QAM), and $\mathcal{C}$ coding schemes are available. For the n-th IoT device, the selected MCS at TS t are denoted as $m_{n,k}^t$ and $c_{n,k}^t$, respectively. Let $r_{m,c}$ (in bits per symbol) represent the transmission efficiency, assuming that N_p denotes the number of symbols per packet or frame. The achievable data rate (measured in bits per frame) for the n-th IoT device is then given by Eq. (15) [44].

$$\mathrm{R}_n^t = r_{m,c}\left(1 - \rho_{m,c}(\mathrm{sinr}_{n,k}^t)\right)N_p, \tag{15}$$

where $\rho_{m,c}$ denotes the packet error rate (PER) and is defined as shown in Eq. (16) [44].

$$\rho_{m,c}(\mathrm{sinr}_{n,k}^t) = 1 - \left(1 - f_{m,c}(\mathrm{sinr}_{n,k}^t)\right)^{N_p}. \tag{16}$$

In the above expression, $f_{m,c}(\cdot)$ represents the symbol error rate (SER), which is given by Eq. (17).

$$f_{m,c}(\mathrm{sinr}_{n,k}^t) = 2\left(1 - \frac{1}{\sqrt{m_{n,k}^t}}\right) \times Q\left(\sqrt{\frac{3\ G\log_2(m_{n,k}^t)\ \mathrm{sinr}_{n,k}^t}{m_{n,k}^t - 1}}\right), \tag{17}$$

where $Q(.)$ is the tail distribution function of the standard Gaussian distribution. Coding gain G is defined as $G = c_{n,k}^t d_{free,H}$, where $d_{free,H}$ is the Hamming free distance.

3) PF SCHEDULING ALGORITHM

We consider the PF scheduling algorithm to schedule RRBs among the coexisting devices while striking a suitable balance between fairness and throughput efficiency [35]. The PF ratio for device n at TS t is defined as[6] shown in Eq. (18).

$$PF_n = w_n(t)\ log_2\left(1 + \frac{P_c\ g^t_{n,k}}{\sigma^2}\right), \tag{18}$$

where $w_n(t) = \frac{1}{\hat{\mathrm{R}}^t_n}$ is the weight assigned to device n at each TS, and P_c is a fixed value of transmit power for all IoT links. Without loss of generality, we consider $P_c = P_{min}$ for all IoT links. The long-term average rate of device n at t TS denoted as $\hat{\mathrm{R}}^t_n$ is updated using an exponential moving average to smooth variations over time (see Eq. (19)).

$$\hat{\mathrm{R}}^t_n = (1-\alpha_r)\hat{\mathrm{R}}^{t-1}_n + \alpha_r \mathrm{R}^{t-1}_n, \tag{19}$$

where $\alpha_r \in [0,1]$ determines the smoothing factor or the window size for the exponential moving average. R^{t-1}_n is the achieved rate of device n at the previous time step, which depends on the modulation m and coding c schemes selected at $t-1$. At each TS, the PF ratios of all devices are sorted, and available RRBs are allocated to devices accordingly. All steps of the RRB scheduling algorithm are set out in Algorithm 1.

Algorithm 1 PF Scheduling Algorithm

1: **Input:** Total scheduling duration T, number of devices N, R available RRBs.
2: **Initialization:** Initialize with random scheduling.
3: **for** $t = 1 : T$ **do**
4: Compute the PF ratio for all devices.
5: Sort devices in descending order based on their PF ratio.
6: Select the top R devices from the sorted list.
7: Assign a single orthogonal RRB to each selected device.
8: **end for**
9: **Output:** RRB scheduling for devices at each TS.

Remark 1: In the present study, assuming that the multi-RAT APs always have data to transmit to IoT devices, we adopt PF scheduling and a full-buffer traffic model. This setup emulates persistent downlink demand and enables performance evaluation under worst-case load conditions. While typical IoT traffic is often bursty or periodic, emerging IoE applications are increasingly characterized by continuous or semi-continuous data flows [45]. However, to accommodate delay-sensitive or freshness-critical traffic, the proposed DRL-based link adaptation framework is scheduler-agnostic and can be readily extended to alternative scheduling policies.

[6]PF scheduling not only prioritizes the devices with the best channel gain, but also with its long-term behavior.

IV. RADIO RESOURCE MANAGEMENT PROBLEM

A. PROBLEM FORMULATION

In the present study, we aim to maximize the long-term sum-throughput of the network by optimizing the IoT links' transmit power, modulation orders, and coding rates at each TS. The RRM problem is formulated as shown in Eq. (20).

$$\text{P0:} \max_{\boldsymbol{P}^t_{n,k}, \boldsymbol{m}^t_{n,k}, \boldsymbol{c}^t_{n,k}} \lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T}\sum_{n=1}^{N} R^t_n$$
$$\text{s.t.} \begin{cases} \text{C1: } P_{min} \le P^t_{n,k} \le P_{max}, \forall n \in \mathcal{N} \\ \text{C2: } m^t_{n,k} \in \mathcal{M}, \forall n \\ \text{C3: } c^t_{n,k} \in \mathcal{C}, \forall n \end{cases} \tag{20}$$

Constraint (C1) implies that the transmit power between the k-th AP and the n-th IoT device is bounded by P_{min} and P_{max} that represent the minimum and maximum transmit power limits of a RAT AP, respectively. Constraints (C2) and (C3) imply that the optimal MCS are selected from $\mathcal{M}$ and $\mathcal{C}$, respectively. P0 is a MINLP problem, as it involves both continuous and discrete optimization variables, and its throughput function is non-convex.

Lemma 1: P0 is an NP-hard optimization problem.

Proof: The proof is provided in Appendix. □

B. MOTIVATION FOR USING DRL TO SOLVE P0

Problem P0 is NP-hard, and obtaining its optimal solution requires an exhaustive search over the joint space of transmit power, modulation, and coding, which is infeasible in large-scale systems. Beyond this combinatorial complexity, conventional optimization techniques face several fundamental limitations. First, the objective function of P0 is implicit due to the lack of precise knowledge of HWI and ACI parameters, which vary across IoT devices because of dynamic and nonlinear hardware behaviors. This renders classical gradient-based methods inapplicable [46]. Second, optimal centralized optimization requires global CSI, including both direct- and adjacent-link channel gains, the acquisition of which incurs prohibitive signaling overhead in dense deployments [32]. Third, even assuming perfect CSI and exact HWI/ACI models, solving P0 via a centralized MINLP solver entails excessive computational complexity, making per-device link adaptation at each TS impractical.

Given these challenges, we cast P0 as a learning problem rather than a traditional optimization problem, with the objective of learning a policy that maps system states to transmission parameters under uncertainty and dynamics. Although supervised DL could, in principle, approximate this mapping, it requires labeled optimal solutions over a vast state space, whose generation is computationally intractable due to the NP-hardness of P0 and the continuous network dynamics. To address this, we reformulate P0 as a decentralized Markov game and adopt an MADRL framework. In this model-free setting, agents learn their policies directly through interaction with the environment. The DRL-based approach enables: (i) implicit handling of

interference and hardware uncertainties; (ii) real-time, low-complexity decision making across a large number of agents; and (iii) scalable and distributed link adaptation without requiring full CSI.

Why DTN? Training DRL models directly in a live network is impractical due to safety risks, high operational costs, excessive overhead, and suboptimal exploration. DTNs provide a high-fidelity virtual environment for training and continuously updating DRL models by accurately replicating key network dynamics. Nevertheless, DTNs inevitably deviate from real-world conditions (e.g., due to channel variations, HWI, and ACI), leading to a *sim-to-real gap*. To mitigate this, we adopt a replay-memory-based continual learning strategy [47]. Specifically, the DRL model trained in the DT is deployed at RAT APs, where a subset of real-world experiences is periodically collected and stored in a replay memory. These samples are then used to fine-tune the DTN and continuously adapt the DRL model using both simulated and real experiences.

V. DIGITAL TWIN-ENABLED CONTINUAL MADRL FRAMEWORK

We first reformulate P0 as a multi-player stochastic game (also known as a Markov game). This approach is motivated by the need for a distributed solution to P0 while taking interdependence of devices' decisions within the system. To manage the computational complexity, we discretize the decision space. Specifically, we represent transmit power levels using a discrete set,[7] as defined in Eq. (25). Due to the presence of ACI, each IoT link's achievable downlink rate is affected not only by the transmit power of its serving RAT AP, but also by the power levels allocated to other coexisting IoT links, particularly those operating on adjacent channels. Hence, a game-theoretic approach is well-suited to solve P0 in a decentralized manner. Furthermore, the IoT network exhibits stochastic dynamics driven by both the environment and the actions of individual links, which naturally calls for a Markov game formulation. In this setting, each IoT link is modeled as a player that iteratively explores actions (i.e., transmit power and MCS) according to a well-defined strategy and periodically refines it, enabling autonomous learning of optimal link adaptation strategy from local state information (as in Eq. (21)) without requiring knowledge of other IoT links' strategies.

A. PROPOSED MULTI-PLAYER STOCHASTIC GAME

1) GAME FORMULATION

P0 is reformulated as a Markov game, formally defined as tuple $\mathcal{G} = (\mathcal{N}, \mathcal{S}, \mathcal{A}, \mathcal{R}, \pi)$ with the following components:

- $\mathcal{N}$ is the set of all agents. The stochastic game is conducted in a virtual space and a virtual agent is created for each IoT link.
- $\mathcal{S}$ is the state space of the game that comprises representative features extracted from each agent's interaction with the environment. To accurately characterize the environment, the state space includes the following key elements:
  (i) The local CSI between the k-th RAT AP and the n-th IoT device at TS t, $g^t_{n,k}$.
  (ii) Historical data including SINR of the IoT devices at TS $t-1$, denoted as $\text{sinr}^{t-1}_{n,k}$, and average transmission rate at TS $t-1$, represented as R^{t-1}_n.
  (iii) Actions taken by the agent in the previous TS, including $P^{t-1}_{n,k}$, $m^{t-1}_{n,k}$, and $c^{t-1}_{n,k}$.

  Therefore, for each agent, the state space is formally defined as shown is Eq. (21).

$$\mathcal{S}_n = \{g^t_{n,k}, \text{sinr}^{t-1}_{n,k}, R^{t-1}_n, P^{t-1}_{n,k}, m^{t-1}_{n,k}, c^{t-1}_{n,k}\}. \tag{21}$$

  The overall state space for all agents is expressed as shown in Eq. (22).

$$\mathcal{S} = \bigcup_{n=1}^{N} \mathcal{S}_n. \tag{22}$$

  Incorporating historical data (e.g., past SINR, throughput, and actions) into the state space enables the agent to (i) track the effectiveness of past actions and (ii) capture inherent temporal correlations across consecutive actions arising from correlated channel realizations. This approach enhances decision making by leveraging past experiences to improve future performance.
- $\mathcal{A}$ is the overall action space of the game denoted as $\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2 \times \ldots \times \mathcal{A}_N$. In consistent with P0 formulation, the goal of each player in the Markov game is to find the optimal combination of the transmit power, modulation, and coding rate at the beginning of each game round (i.e., each TS). To describe each of these schemes, we use a set of discrete values. Specifically, for the transmit power, we consider discrete power levels ranging from P_{min} to P_{max}. The modulation and coding action space encompasses all available MCS levels. Consequently, the action space of each agent is designed as shown in Eq. (23).

$$\mathcal{A}_n = \{m^t_{n,k} \in \mathcal{M}, P^t_{n,k} \in \mathcal{P}, c^t_{n,k} \in \mathcal{C}\}, \tag{23}$$

where

$$\mathcal{M} = \{M_1, M_2, \ldots, M_{|\mathcal{M}|}\}, \tag{24}$$

$$\mathcal{P} = \left\{P_{min}, \frac{P_{max}}{|\mathcal{P}|-1}, \frac{2P_{max}}{|\mathcal{P}|-1}, \ldots, P_{max}\right\}, \tag{25}$$

$$\mathcal{C} = \{C_1, C_2, \ldots, C_{|\mathcal{C}|}\}, \tag{26}$$

$\mathcal{M}$, $\mathcal{P}$, and $\mathcal{C}$ represent the modulation, power, and coding rate action spaces, respectively. Furthermore, we use $|.|$ to denote the cardinality of each action space. For implementation in the proposed DRL-based framework, each joint action $(m^t_{n,k}, P^t_{n,k}, c^t_{n,k}) \in \mathcal{M} \times \mathcal{P} \times \mathcal{C}$ is mapped to a single discrete action index corresponding to one unique combination of transmit power, modulation, and coding rate. Consequently, the total number of discrete actions for each agent is $|\mathcal{A}_n| = |\mathcal{M}| \times |\mathcal{P}| \times |\mathcal{C}|$.

[7] Although theoretical optimization models assume continuous power allocation, hardware constraints and implementation ease require that power allocation be discrete and quantized. Consequently, devices can only choose from predefined power levels (e.g., level 1, level 2, level 3, etc.).

• $\mathcal{R}$ is the agents' common reward given by the sum throughput, i.e., $\mathcal{R} = \sum_{n=1}^{N} R_n^t$.

• $\Pi = \{\pi_1, \pi_2, \ldots, \pi_N\}$ is the overall strategy space of the game. In particular, $\pi_n : \mathcal{S} \times \mathcal{A} \rightarrow [0.1]$ provides the resource allocation policy of the n-th agent, $\forall n \in \mathcal{N}$. The resource allocation policy is essentially a probability mass function (PMF) over possible actions in a given state. More specifically, $\pi_n(s_{n,t}, a_{n,t}) = [\pi_n(s_{n,t}, a_{n,t})]_{a_{n,t} \in \mathcal{A}_n}$, where $\pi_n(s_{n,t}, a_{n,t}) \in [0, 1]$ denotes the n-th agent's probability of selecting action $a_{n,t}$ from action space $\mathcal{A}_n$ in state $s_{n,t}$, $\forall n \in \mathcal{N}$ and $\forall s_{n,t} \in \mathcal{S}_n$. In this case, $\sum_{a_{n,t} \in \mathcal{A}_n} \pi_n(s_{n,t}, a_{n,t}) = 1$ holds. To describe the solution to this Markov game, we first introduce the notion of state value function as follows [48].

Definition 1: For an infinite-horizon Markov game, state value function $V_n(s_{n,t}, \pi_n, \pi_{-n})$ represents the n-th agent's expected utility over the state transitions starting from state $s_{n,t}$ when the agents employ the resource allocation policy $\{\pi_n, \pi_{-n}\}$ and is defined as shown in Eq. (27).

$$V_n(s_{n,t}, \pi_n, \pi_{-n}) = \mathbb{E}\left[\sum_{i=0}^{\infty} \gamma^i R_{i|s_0 = s_{n,t}, \pi_n, \pi_{-n}}\right]$$
$$= \bar{\mathrm{R}} + \gamma \sum_{s_{n,t+1} \in \mathbf{S}_n} \mathcal{P}_{s_{n,t}, s_{n,t+1}}(\pi_n, \pi_{-n}) V_n(s_{n,t+1}, \pi_n, \pi_{-n}), \quad (27)$$

where $\bar{\mathrm{R}} = \mathbb{E}_\pi[R_0(s_{n,t}, \pi_n, \pi_{-n})]$ indicates the reward to be expected in state $s_{n,t}$ according to resource allocation policy $\{\pi_n, \pi_{-n}\}$, $\mathcal{P}_{s_{n,t}, s_{n,t+1}}(\pi_n, \pi_{-n})$ presents the probability of transitioning from state $s_{n,t}$ to state $s_{n,t+1}$ with resource allocation policy $\{\pi_n, \pi_{-n}\}$, $\gamma \in (0, 1)$ represents the discount factor that captures how important recent experiences are in the state value function, and, finally, π_{-n} denotes the resource allocation policies of all agents except the n-th agent.

2) SOLUTION TO GAME $\mathcal{G}$

In a Markov game, each agent's optimal policy corresponds to its best response to the policies of the other agents, and it is obtained by maximizing the agent's state-value function, expressed as follows.

$$\pi_n^* = \arg\max_{\pi \in \Pi} V_n(s_{n,t}, \pi_n, \pi_{-n}), \forall s_{n,t} \in \mathbf{S}_n, n \in \mathcal{N}. \quad (28)$$

Definition 2: The optimal solution to the Markov game is expressed in terms of the Nash equilibrium (NE) resource allocation policies $\Pi^* = \{\pi_1^*, \pi_2^*, \ldots, \pi_N^*\}$, where each agent's policy is its best response (i.e., solution to (28)) with respect to the other agents' policy. Thus, in NE, the condition in Eq. (29) is satisfied for the n-th agent, $\forall n \in \mathcal{N}$.

$$V_n(s_{n,t}, \pi_n^*, \pi_{-n}^*) \geq V_n(s_{n,t}, \pi_n, \pi_{-n}^*), \forall s_{n,t} \in \mathcal{S}_n, \forall \pi_n, \quad (29)$$

where π_{-n}^* denotes the optimal resource allocation policies of all agents except the n-th agent.

Since the state value function does not have any closed-form expression and it depends on the neighbor agents' actions, it is non-trivial to derive optimal policies by (directly) solving (28). MARL enables agents to iteratively learn a set of equilibrium policies by iteratively interacting with the environment without any specific information about the agents' state transition probabilities [48]. Accordingly, we develop a MARL framework to obtain the agents' NE resource allocation policies.

3) LEARNING-BASED OPTIMAL SOLUTION TO GAME $\mathcal{G}$

To solve (28) via learning, we use the multi-agent Q-learning framework to determine the optimal state value function from the optimal Q-function. The Q-function denoted by $Q_m(s_{n,t}, a_{n,t}, \pi)$ represents the n-th agent's expected utility over the state transitions starting from state $s_{n,t}$ and action $a_{n,t}$ using resource allocation policy π, $\forall n \in \mathcal{N}$, and it is expressed by

$$Q_n(s_{n,t}, a_{n,t}, \pi)$$
$$= \mathbb{E}\left[\sum_{i=0}^{\infty} \gamma^i R_{i|s_0 = s_{n,t}, a_0 = a_{n,t}, \pi}\right]$$
$$= \bar{\mathrm{R}} + \gamma \sum_{s_{n,t+1} \in \mathbf{S}_n} \mathcal{P}_{s_{n,t}, s_{n,t+1}}(\pi_n, \pi_{-n})$$
$$\sum_{a_{n,t+1} \in \mathbf{A}_n} \pi(s_{n,t+1}, a_{n,t+1}) Q_n(s_{n,t+1}, a_{n,t+1}, \pi). \quad (30)$$

If we compare Eq. (27) and Eq. (30), with $\forall n \in \mathcal{N}$ and $s_n^t \in \mathbf{S}_n$, we obtain Eq. (31).

$$V_n(s_n, \pi_n, \pi_{-n}) = \sum_{a_n \in \mathbf{A}_n} \pi(s_n, a_n) Q_n(s_n, a_n, \pi). \quad (31)$$

The optimal state value function [49, Eq. (30)] can be expressed as shown in Eq. (32).

$$V_n(s_{n,t}, \pi_n^*, \pi_{-n}^*) = \max_{\pi_n} V_n(s_n, \pi_n, \pi_{-n}^*)$$
$$= \max_{\pi_n} \sum_{a_{n,t} \in \mathbf{A}_n} \pi(s_{n,t}, a_{n,t}) Q_n(s_{n,t}, a_{n,t}, \pi)$$
$$= \sum_{a_{n,t} \in \mathbf{A}_n} \pi(s_{n,t}, a_{n,t}) Q_n^*(s_{n,t}, a_{n,t}) \leq \max_{a_{n,t} \in \mathbf{A}_n} Q_n^*(s_{n,t}, a_{n,t}), \quad (32)$$

where $Q_n^*(s_{n,t}, a_{n,t}, \pi) = \max_{\pi_n} Q_n(s_{n,t}, a_{n,t}, \pi)$ is the optimal Q-function. The maximum value of the state value function is obtained from the optimal Q-function of the most profitable action. Accordingly, the optimal solution to (28) for the n-th agent in in state $s_{n,t}$, $\forall n \in \mathcal{N}$ and $s_{n,t} \in \mathcal{S}_n$, is obtained as shown in Eq. (33).

$$\pi(s_n^t, a_n^t) = \begin{cases} 1, & \text{if } a_{n,t} = a_{n,t}^* \\ 0, & \text{if } a_{n,t} \neq a_{n,t}^* \end{cases} \quad (33)$$

where $a_{n,t}^* = \arg\max_{a_n \in \mathbf{A}_n} Q_n^*(s_n, a_n)$. Therefore, the task of determining the agents' optimal resource allocation policies boils down to learning optimal Q-functions. However, since the state space is continuous (see Eq. (21)), conventional Q-learning suffers from the curse of dimensionality; therefore, we employ a multi-agent DQN approach. In what follows, we provide an overview of the DQN and the proposed DT-enabled continual MADRL learning framework.

B. OVERVIEW OF DEEP Q-NETWORK

The DQN is a popular DRL algorithm dealing with complex and high-dimensional optimization problems. In general, the DQN comprises the following four main elements: s^t, a^t, r^{t+1}, and s^{t+1}, where s represents the state of the environment, a is the action to be taken by the agent, and r is the reward received from the environment. At TS t, the agent observes the state s^t of its environment and then takes action a^t according to a certain policy $\pi(s/a)$. The policy serves as a map associating a specific action to be executed by the agent based on the given state. Once the agent takes an action, it receives a reward r^{t+1}, and the environment moves from one state to the next s^{t+1}. The Q-learning's main goal is to find an optimal policy $\pi^*(s/a)$ that would maximize the discounted accumulated reward, which is defined as $R_t = \sum_{k=1}^{\infty} \gamma^k \ r^{t+k}$, where $\gamma \in [0,1]$ is the discount factor and r^{t+k} is the value of the reward at TS $t+k$. Basically, the Q-learning is based on the action value function, which is the expected return for selecting action a in state s based on policy π (see Eq. (34)).

$$Q_\pi(s,a) = \ \mathbb{E}(R_t | s^t = s, a^t = a). \tag{34}$$

The optimal action value function, denoted as $Q^*(s,a)$, captures the maximum achievable action value by adhering to any policy [19]. This optimal value can be obtained using the Bellman equation (see Eq. (35)).

$$Q^*(s,a) = \ \mathbb{E}(r^{t+1} + \gamma \max_{a'} Q^*(s^{t+1}, a') | s^t = s, a^t = a). \tag{35}$$

In the DQN, we use a deep neural network (DNN) to compute the Q-value function, $q(s,a;\theta)$, where θ denotes the weight of the DNN. Here, the action is determined using the ϵ-greedy algorithm. Specifically, for a given state s, the agent selects the action $a = \arg\max_a \ q(s,a;\theta)$ with probability $1-\epsilon$, to then randomly select an action from the action space with probability ϵ. Parameter ϵ represents a trade-off factor between exploitation and exploration. After the agent completes a new experience as a result of the chosen action, the Q-value is updated according to Eq. (36).

$$\begin{aligned} q(s^t, a^t; \theta) \leftarrow (1-\alpha) q(s^t, a^t; \theta) + \alpha [r^{t+1} \\ + \gamma \max_{a'} \ q(s^{t+1}, a'; \theta)], \end{aligned} \tag{36}$$

where $\alpha \in (0,1]$ is the learning rate [50]. To accelerate convergence and ensure stability, DQN employs the following key techniques:

- **Target network:** Mirroring the architecture of the main Q-network, the target network maintains separate weights θ', which are periodically updated (typically every T_{step}). This approach provides consistent ground truth values for updates and prevents instability that would otherwise arise from using the same network for both prediction and target calculation.
- **Experience replay memory:** The experience replay memory D stores past experiences, allowing the agent to sample mini-batches for training. This sampling reduces the bias from correlated recent data and enables the agent to learn from a diverse set of experiences, which results in more robust and unbiased learning.

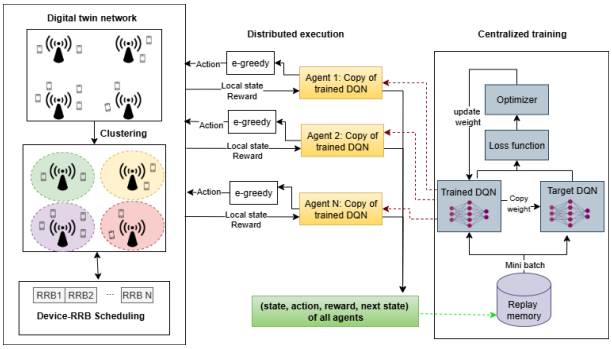


Fig. 3. Overview of the proposed DTN-empowered and replay memory-based continual DRL framework.

Together, these techniques enhance both efficiency and stability of the learning process. The loss function for updating the Q-network using a randomly sampled mini-batch D_b from the experience replay memory is expressed as shown in Eq. (37).

$$\mathcal{L}(\theta) = \frac{1}{2} \sum_{(s,a,r,s') \in D} (r' - q(s,a;\theta))^2, \tag{37}$$

where $r' = r + \gamma \max_{a'} \ q(s', a'; \theta')$ is the target value. Finally, we use the gradient descent algorithm to minimize the loss function in Eq. (37) and train the weights of the DQN over the mini-batch D_b. With this technique, the weights are updated as follows: $\theta \leftarrow \theta - [r' - q(s,a;\theta)] \nabla q(s,a;\theta)$. The target Q-network is then periodically updated by copying parameters from the trained DQN.

C. DTN-ENABLED CONTINUAL DRL FRAMEWORK

1) DESCRIPTION OF THE PROPOSED FRAMEWORK

In this study, we propose a continual DQN algorithm where the IoT link acts as the RL agent, while the DTN serves as the environment. Fig. 3 provides an overview of the architecture of the proposed solution. The DTN environment comprises K RAT APs and N uniformly distributed IoT devices. The framework begins by clustering IoT devices to the nearest RAT AP, forming clusters $\{\mathcal{C}_1, \mathcal{C}_2, \cdots, \mathcal{C}_K\}$. After clustering, the devices are scheduled to RRBs based on the PF approach (Algorithm 1). In the proposed framework, each IoT link is modeled as an independent RL agent. At each step of an RL episode, an agent receives its state from the DTN environment and subsequently performs the corresponding action. Importantly, each IoT link operates independently, resulting in a total of N agents. To address the scalability issue caused by simultaneous training a total of N agents, we adopt a centralized training with a decentralized execution (CTDE) approach [32]. In this approach, all links share an identical DQN that is centrally trained by a centralized controller using the experiences collected from all agents. This design ensures scalability and efficiency while maintaining decentralized decision making

during execution. Furthermore, training the DQN based on collective experiences of all agents enhances stability and promotes collaborative learning. During execution, each agent operates independently, relying on the shared DQN to select actions based on its own state. While the agents use the same DQN, their actions differ, as each of them encounters unique states, leading to varied action selections. Importantly, during training, the DTN has access to complete system information, allowing it to accurately compute global rewards and state transitions for each IoT link and to store the corresponding experience tuples in the replay memory. This memory is then used to train the DRL model. All policy updates are performed centrally, and the refined DQN model is redistributed and applied synchronously across all IoT links over reliable control channels, thus ensuring that all agents operate with a consistent policy version at any given time.

Algorithm 2 provides the overall steps for training our proposed framework using the CTDE approach. The input of our algorithm is the maximum number of training episodes Z and the maximum number of steps per episode T. The algorithm begins by constructing the DTN environment, followed by initializing the replay memory, the trained DQN, and the target DQN. It then proceeds as follows: within each training episode, the IoT devices are clustered to the nearest RAT AP using a proximity-based clustering mechanism (Line 7). For each step within an episode, the PF scheduling algorithm is executed to determine the RRB allocation for each device (Line 9). For each IoT link, the state of the environment is observed, and the DQN agent selects an action based on the ϵ-greedy policy (Lines 10-17). This policy allows the agent to balance exploration and exploitation by selecting the action with the highest estimated Q-value with probability $1-\epsilon$ or choosing a random action with probability ϵ. On performing the selected action, the algorithm interacts with either the physical environment or the DT environment. Of note, the algorithm interacts with the physical environment every Z_k RL training episodes, where Z_k is judiciously chosen to strike a balance between overhead and accuracy. During these interactions, the physical twins of the agents (i,e., IoT links) deploy the link adaptation parameters selected by the most recently trained DRL model in the real-world channel and observe the resulting rewards. The IoT links then forward such channel-specific dataset (comprising states, actions, and rewards) to the context database of the DTN via RAT APs.[8] These datasets are subsequently stored in the replay memory for fine-tuning the DRL model within the DT domain (Lines 18-23). For the sake of simplicity, we assume a uniform TS structure and a fixed interaction frequency across all devices during the training phase.

Algorithm 2 Algorithm for Training the DQN-Based Resource Allocation

1: **DT Construction Phase:** Create a virtual DT environment to emulate the real system behavior, by (a) placing multi-RAT APs and IoT devices, (b) creating propagation channel and non-linear impairments (ACI and HWI), (c) enabling RRM functionalities (device clustering, RRB scheduling, and link adaptation), and (d) integrating the capability to compute each IoT link's SINR and throughput based on the RRM decisions, as described in Section III.
2: **Input:** Maximum number of episodes Z, maximum number of steps per episode T, and frequency of interaction with physical environment Z_k.
3: Initialize replay memory D to zero (**Start of Training Phase**).
4: Create trained DQN with random weights θ.
5: Create target DQN with $\theta' = \theta$.
6: **for** $i = 1 : Z$ **do**
7: Cluster IoT devices to the nearest RAT AP.
8: **for** $t = 1 : T$ **do**
9: Perform the PF scheduling algorithm.
10: **for** $n = 1 : N$ **do**
11: Observe state of the environment $s_{n,t}$.
12: Generate random number $\eta \in [0, 1]$.
13: **if** $\eta > \epsilon$ **then**
14: Select $a_{n,t} = \arg\max_{a_{n,t} \in \mathcal{A}_n} q(s_{n,t}, a_{n,t}; \theta)$, where q is estimated by the trained network.
15: **else**
16: Randomly select action $a_{n,t}$.
17: **end if**
18: **if** $i \mod Z_k = 0$ **then**
19: Observe real reward $r_{n,t+1}$ and real new state $s_{n,t+1}$.
20: **else**
21: Observe DT reward $r_{n,t+1}$ and DT new state $s_{n,t+1}$.
22: **end if**
23: Save new experience $(s_{n,t}, a_{n,t}, r_{n,t+1}, s_{n,t+1})$ into experience replay memory D.
24: **end for**
25: Sample random mini-batch D_b experiences from D.
26: Use the backpropagation method to update the weights of the trained DQN.
27: Update weights of the target network θ' by weights of the trained network θ every T_{step}.
28: **end for**
29: **end for**
30: **Output:** Trained resource allocation agent $q^*(s, a; \theta)$.

After collecting experiences (from either DT or physical environment), a mini-batch is randomly sampled from the replay memory (Line 25). The trained DQN updates its

[8]Of note, the channel-specific data collection from the physical network to DTN is required only during the DRL training phase. Once the DRL model is fully trained, inference and decision making are carried out in a fully distributed manner by the individual IoT links, without requiring any real-time CSI exchanges between the IoT links and DTN.

weights using the backpropagation method by minimizing the temporal difference loss Eq. (37) (Line 26). The weights of the target DQN are periodically updated to match the trained DQN, thus ensuring stability in learning (Line 27). Finally, the algorithm outputs the trained DQN policy, where actions are chosen based on the maximum Q-value for the observed state. This policy is subsequently deployed for resource allocation in the multi-RAT IoT system (Line 30). Importantly, while our study focuses on the downlink, the proposed architecture and learning model can also be readily extended to handle uplink scenarios.

Remark 2: The proposed framework follows a continual learning paradigm, where a DT-trained DRL agent is incrementally fine-tuned using a sequential stream of experiences generated by multiple IoT links operating under time-varying environmental conditions. To preserve knowledge acquired from prior interactions, an experience replay mechanism is employed, whereby the agent is trained using a combination of newly collected experiences D_t (i.e., experiences collected at TS t) and a subset of previously stored experiences $(D_0, D_1, \ldots, D_{t-1})$ drawn from a shared replay buffer. Since replay memory aggregates experiences across multiple IoT links and diverse environment regimes, the resulting policy learns representations that generalize both spatially (across IoT links) and temporally (across evolving channel conditions). As a result, the proposed framework fundamentally differs from conventional periodic retraining approaches that retrain models from scratch for each new dataset or experience batch, and thus lack knowledge retention capability and require large overhead.

2) SIGNALING OVERHEAD

In this section, we compare the signaling overhead of training our algorithm solely in a physical network versus our proposed approach—predominantly training in the DT domain while periodically collecting physical network data.

- **Training solely in the physical environment:** Here, we consider that Algorithm 2 trains our proposed framework exclusively in the physical environment. At each episode, Algorithm 2 computes the distance between N existing devices and K RAT APs, clustering each device to its nearest RAT AP. Without the loss of generality, we consider T_D as the amount of information required from each device to accomplish a task. Accordingly, each new clustering step involves a total of NKT_D information exchanges between RAT APs and devices. Next, the algorithm calculates the instantaneous CSI for each device with its associated RAT AP, resulting in an additional NT_D information exchanges. Subsequently, devices are scheduled to available RRBs, and the PF ratio in Eq. (18) is computed, leading to NT_D information exchanges between the centralized scheduler and the devices. For each device, the algorithm observes the current state, performs an action, receives a reward, and observes the next state, resulting in a total of $4NT_D$ information exchanges across all devices. Experience tuple $(s_{n,t},\ a_{n,t},\ r_{n,t+1}\, s_{n,t+1})$ is then stored in the replay memory, adding another N information exchange. Updating the DQN model's weights requires exchanging updated parameters, which contributes D_b exchanges. Similarly, updating the target network weights requires additional D_b exchanges. Therefore, at each TS of Algorithm 2, the total information exchange amounts to $(N(6T_D + KT_D + 1) + 2D_b)$. Over the entire execution of the algorithm, the total signaling overhead is given by the following: $ZT(N(6T_D+KT_D+1)+2D_b)$.
- **Our proposed RL training:** As outlined in Algorithm 2, in our proposed scheme, the DRL model is trained in the DT environment and periodically streamlines using the experience data collected from the physical network. Since the training in DT is purely software-based, it does not introduce any communication overhead T_D for device-specific information acquisition. However, every Z_k episodes, the algorithm transfers the actions selected by the most recent DRL model to the RAT APs and collects real-world experiences to ensure alignment with actual network conditions. Therefore, the total signaling overhead can be expressed as shown in Eq. (38).

$$\frac{Z}{Z_K}T(N(6T_D + KT_D + 1) + 2D_b) + \left(Z - \frac{Z}{Z_K}\right)T(NK + 7N + 2D_b). \tag{38}$$

In Eq. (38), the first term represents the over-the-air information exchanges resulting from interactions with the physical environment, whereas the second term corresponds to the total information exchanges within the DT environment. The proposed approach explicitly reduces over-the-air signaling overhead during DRL training.

D. OVERALL ALGORITHM

1) DESCRIPTION OF THE OVERALL ALGORITHM

After training the DQN-based resource allocation model using Algorithm 2, the trained model is saved and deployed in the physical network (i.e., IoT devices). Algorithm 3 outlines the steps involved in executing the resource allocation process during testing. The algorithm begins by capturing the positions of IoT devices (Line 3) and clustering them to their nearest RAT APs (Line 4). For each TS, each RAT AP collects the instantaneous channel gains for its associated devices (Line 6). Next, PF scheduling is applied using Algorithm 1 to assign available RRBs to devices based on fairness (Line 7). Finally, each IoT link observes the current state of the environment (Line 9) and performs the optimal resource allocation action (Line 10). This process is repeated until the total number of TSs T_s is reached. The final output of the algorithm includes device clusters, device-to-RRB scheduling, and an optimal resource allocation solution for problem P0 at each TS. Algorithm 3 adopts a semi-centralized architecture that combines centralized device clustering and RRB scheduling with distributed link adaptation. Specifically, the centralized controller performs PF-based RRB scheduling across heterogeneous multi-RAT IoT links, thus achieving a balanced tradeoff between

Algorithm 3 DRL-Enabled ACI and HWI Aware Distributed Link Adaptation Algorithm

1: **Input:** Number of RAT APs K; number of IoT devices N and RRBs R; trained DRL model $q^*(s, a; \theta)$; total number of TSs T_s.
2: **Initialize:** TS index $t = 1$.
3: Determine the positions of IoT devices.
4: Cluster IoT devices to the nearest RAT AP.
5: **repeat**
6: Collect instantaneous channel gains $\{g^t_{n,k}\}$, for the existing IoT devices.
7: Perform the PF scheduling using Algorithm 1.
8: **for** $n = 1 : N$ **do**
9: Observe state of the environment $s_{n,t}$.
10: Determine the optimal set of transmission parameters (power level and MCS): $a^*_{n,t} = \arg\max_{a_{n,t} \in \mathbf{A}_n} q^*(s_{n,t}, a_{n,t}, \theta)$.
11: Inform the associated RAT-AP of the selected transmission parameters over a reliable control channel for downlink data transmission.
12: **end for**
13: **until** $t > T_s$
14: **Output:** Device clusters, devices-RRBs scheduling, and resource allocation solution to P0 at each TS.

throughput and fairness. Meanwhile, each IoT link independently performs link adaptation using its local state and the trained DQN model to select the appropriate transmit power level and MCS index. By decoupling global RRB coordination from local link adaptation, Algorithm 3 remains scalable and well-suited for large-scale IoT networks.

2) COMPUTATIONAL COMPLEXITY

Computational complexity of Algorithm 3 is determined by analyzing its key steps. Capturing the positions of IoT devices (Line 3) requires $\mathcal{O}(N)$ operations, while clustering each device to the nearest RAT AP (Line 4) involves checking all K APs per device, leading to a worst-case complexity of $\mathcal{O}(NK)$. Next, collecting the instantaneous channel gains for all devices at their associated RAT APs (Line 6) requires $\mathcal{O}(N)$ computations, while performing PF scheduling (Line 7) typically incurs a complexity of $\mathcal{O}(N \log N)$. Resource allocation decisions for all devices (Lines 9-10) require selecting an action from the available action space, resulting in a complexity of $\mathcal{O}(N|\mathcal{A}_n|)$. Summing these steps, the overall complexity of Algorithm 3 is then $\mathcal{O}(2NK + N \log N + N|\mathcal{A}_n|)$.

3) CONTROL OVERHEAD OF THE PROPOSED ALGORITHM

The centralized device clustering step requires collecting IoT device positions, incurring N uplink control signaling exchanges. Note that, to limit signaling overhead, such position information is collected infrequently—namely, once in several TSs. The centralized RRB scheduling step requires per-link CSI, i.e., the channel gain between each device and its serving AP, at every TS. This CSI is estimated at the associated RAT AP using standard-compliant procedures and then is forwarded to the centralized controller via fronthaul/backhaul links, resulting in N uplink feedback exchanges per TS. The distributed link adaptation step at each IoT link requires only local CSI for the current TS and the received SINR from the previous TS, also requiring N uplink feedback exchanges per TS. Overall, the feedback required by Algorithm 3 is standard-compliant, as it does not require global or cross-cell CSI acquisition (e.g., CSI from non-serving RAT APs operating on adjacent channels) and any inter-agent coordination signaling during online operation.

VI. SIMULATION RESULTS

A. BENCHMARK SCHEMES

• **Random algorithm (RA)**: Power, modulation, and coding rate are randomly selected from sets $\mathcal{P}$, $\mathcal{M}$, and $\mathcal{C}$.

• **Single-agent DRL (SADRL):** This approach involves a centralized agent that observes the global state of the environment and performs joint action selection for all devices.

• **Exhaustive search algorithm (ESA)**: In this scheme, the power level, modulation order, and coding rate are jointly selected via an exhaustive search over the entire feasible set of configurations. The combination that maximizes the sum-throughput is then selected. Perfect knowledge of ACI and HWI is considered in order to achieve an upper bound of sum-throughput.

• **ESA and fractional programming (FP)**: This benchmark serves as an upper bound for performance evaluation. Specifically, we perform an exhaustive search over all possible combinations of modulation schemes and coding rates selected from the discrete sets $\mathcal{M}$ and $\mathcal{C}$. For each (m, c) pair, we then optimize the transmission power using FP, since power is treated as a continuous variable. Within the nested loop structure, the FP algorithm computes the optimal power level p for the given (m, c), and the resulting sum throughput is calculated and stored. After evaluating all combinations, the configuration that yields the highest overall sum throughput is selected. Similarly to the ESA method, perfect knowledge of ACI and HWI is assumed.

• **Maximum power 3GPP-CQI based MCS selection (MaxP-3GPP-MCS):** In this benchmark scheme, all RAT APs transmit at a fixed maximum power level ($P_{\max}$). The MCS for each IoT link is selected based on the SINR, following the standard 3GPP CQI-based AMC method. Specifically, for each IoT link, the SINR is first calculated by applying the maximum transmit power $P_{\max}$ to (14). Then, the corresponding modulation order and coding rate are selected by comparing the computed SINR with predefined SINR thresholds from the standard CQI-MCS lookup table (as defined in [51, Table 2.1]). This ensures that the selected MCS is supported under the current channel conditions.

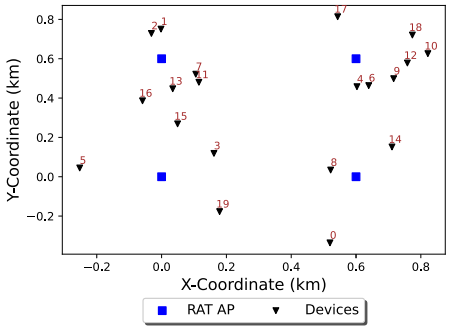


Fig. 4. **Network configuration for 4 RAT APs and 20 devices.**

TABLE 1. **Default simulation parameters.**

Parameter	Value(s)
Network's parameters	
Maximum power (P_{max})	30 dBm
Minimum power (P_{min})	5 dBm
RAT AP coverage (R)	300 m
Small region radius (r)	30 m
AWGN power spectral density	-174 dBm/Hz
Time slot duration (T_s)	20 ms
Number of transmitted symbol (N_p)	1000 packet/frame
Smoothing factor (α_r)	0.1
Scalar parameter (ϕ)	0.4
Level of HWI (σ_{rt})	0.05
Path loss parameters	
Height of RAT AP (H_k)	10 m
Height of IIoT devices (H_n)	1.5 m
Speed of light (c)	3 10^8 m/s
Carrier frequency (f_c)	6.5 GHz

TABLE 2. **DQN's hyper-parameters.**

Parameter	Value
Episode number (Z)	2000
Learning rate (α)	$5e^{-3}$
Replay memory buffer size (D)	5000
Mini-batch size (D_b)	64
Time step (T_{step})	500

B. SIMULATION SETTINGS

Our simulation model comprises 4 RAT APs and 20 uniformly distributed IoT devices (Fig. 4). The RAT AP's coverage is set to $R = 0.3$ km [32]. In addition, to ensure that the position of the device will never be confused with that of the RAT AP, we define a small region of radius $r = 0.03$ km with no active devices. For the large-scale fading, the path loss parameters are selected following [32] and are presented in Table 1. To simulate the HWIs, we consider that $\sqrt{\sigma_t^2 + \sigma_r^2} = \sigma_{rt} = 0.05$ as previously indicated in [40]. Furthermore, we consider $f_c = 6.5$ GHz in accordance with [34].

The hyper-parameters used for our proposed algorithm's architecture are shown in Table 2. The DQN model comprises one input layer, two hidden layers, and one output layer. The hidden layers consist of $N_1 = 64$, $N_2 = 128$ neurons, respectively. The input size is equal to the number of elements in the state vector, which is 6. For the output layer, we consider that $|\mathcal{M}_m| = 4$, $|\mathcal{P}| = 4$, and $|\mathcal{C}| = 3$; accordingly, the DQN has a total of 48 outputs. In the present study, we consider an error-control system where the devices support multiple modulation schemes, such as BPSK, QPSK, 16QAM, and 64QAM, as illustrated in Table 3. Table 3 provides the SER expressions for the modulation schemes considered. For each scheme, the coding rate is selected from the set $\{\frac{1}{2}, \frac{3}{4}, \frac{5}{6}\}$.

TABLE 3. **Considered modulation schemes and corresponding SERs.**

MCS	SER
BPSK	$f_{1,c}(\text{sinr}_{n,k}^t) = Q(\sqrt{2\ G\ \text{sinr}_{n,k}^t})$
QPSK	$f_{2,c}(\text{sinr}_{n,k}^t) = 2\ (1 - \frac{1}{\sqrt{4}})\ Q(\sqrt{\frac{3\ G \log_2(4)\ \text{sinr}_{n,k}^t}{4-1}})$
16QAM	$f_{3,c}(\text{sinr}_{n,k}^t) = 2\ (1 - \frac{1}{\sqrt{16}})\ Q(\sqrt{\frac{3\ G \log_2(16)\ \text{sinr}_{n,k}^t}{16-1}})$
64QAM	$f_{4,c}(\text{sinr}_{n,k}^t) = 2\ (1 - \frac{1}{\sqrt{64}})\ Q(\sqrt{\frac{3\ G \log_2(64)\ \text{sinr}_{n,k}^t}{64-1}})$

For our DQN algorithm, we employ the hyperbolic tangent (tanh) function as the activation function. Moreover, we use the adaptive ϵ-greedy algorithm to perform actions, where $\epsilon(t) = \max\{\epsilon_{min}, (1 - \lambda_\epsilon)\epsilon(t-1)\}$. Here, $\epsilon(0)$ is set to 0.7, ϵ_{min} to 10^{-2}, and λ_ϵ to 10^{-3}. To update the parameter vector θ, we use the RMSprop optimizer with adaptive learning rate $\alpha(t) = (1 - \lambda)\alpha(t-1)$, where $\alpha(0) = 5\ e^{-3}$ is the initial learning rate and $\lambda = 1\ e^{-3}$ is the learning rate decay. The implementation was carried out in Python 3 and executed on a 64-bit Windows 10 platform equipped with an Intel Core i7-6700 CPU (3.40 GHz) and 8 GB of RAM.

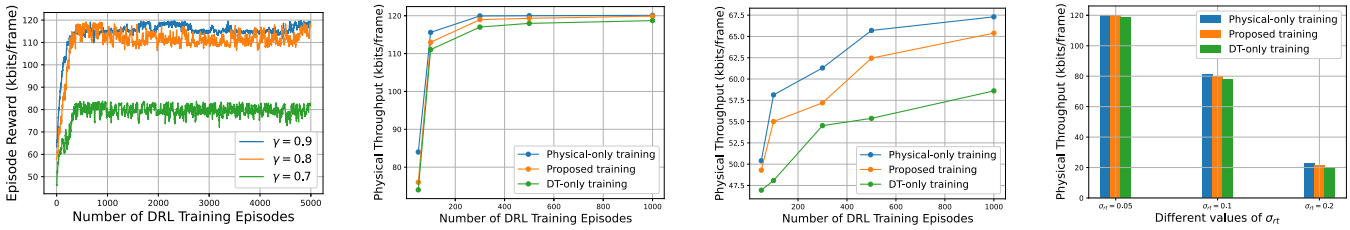
C. IMPACT OF DISCOUNT FACTOR

Fig. 5a evaluates the episodic reward of the DQN agent with respect to the discount factor that determines the relative importance of future rewards compared to the current reward. In other words, the discount factor governs the agent's decision making process by influencing how much weight is assigned to the rewards expected in the future. Therefore, selecting an appropriate value for the discount factor is essential to achieving optimal performance. In this evaluation, we vary the discount factor of the DQN-based resource allocation while keeping other parameters fixed. Fig. 5a shows the DQN's episodic reward (evaluated in the physical twin environment) vs. number of training episodes for different values of the discount factor, such as $\gamma = 0.9$, $\gamma = 0.8$, and $\gamma = 0.7$. As shown in Fig. 5a, the highest reward is achieved when $\gamma = 0.9$. Therefore, we choose $\gamma = 0.9$ as the discount factor in the ensuing performance evaluations.

D. ADVANTAGE OF THE PROPOSED RL TRAINING APPROACH

1) COMPARISON AMONG DIFFERENT RL TRAINING APPROACHES

Here, we train our algorithm for a varying number of DRL training episodes under the following three distinct scenarios:



(a) Impact of discount factors. (b) $\phi = 0.4$. (c) $\phi = 0.9$. (d) Different values of σ_{rt}.

Fig. 5. Comparison among different RL training approaches.

• Physical-only training (Scenario 1): The DRL model is trained solely in the physical environment.

• Proposed training (Scenario 2): The model is primarily trained in the DT environment, with interactions and experience collection from the physical environment occurring every 100 episodes.

• DT-only training (Scenario 3): The model is trained exclusively in the DT environment.

To compare performance across these scenarios, we test the trained models' performance in terms of the physical throughput.[9] Fig. 5b and 5c show the physical throughput for each scenario when $\phi = 0.4$ and $\phi = 0.9$, respectively. The results indicate that our proposed training approach achieves throughput levels comparable to those obtained through physical-only training. For instance, as shown in Fig. 5b, after 300 episodes of DRL training, scenarios 1, 2, and 3 achieve throughputs of 119.94 kbits/frame, 118.96 kbits/frame, and 117 kbits/frame, respectively. Moreover, increasing the deviation of ACI/HWI (ϕ) from the theoretical model widens the performance gap between scenarios 1 and 3, since the DT environment in scenario 3 is unaware of such uncertainties, rendering the trained DRL model sub-optimal in the physical environment. For example, as shown in Fig. 5c, after 500 episodes of DRL training, scenarios 1, 2, and 3 achieve throughputs of 65.7 kbits/frame, 62.45 kbits/frame, and 55.38 kbits/frame, respectively. These findings confirm that our DT-empowered continual-learning DRL approach (Algorithm 2) achieves high accuracy and adaptability under ACI and HWI uncertainties, while significantly reducing signaling overhead.

2) COMPARISON OF RL TRAINING APPROACHES UNDER VARYING IMPAIRMENT LEVELS

Fig. 5d compares the performance of our proposed and other training approaches under different levels of HWIs. The results shown in Fig. 5d demonstrate that our approach achieves comparable performance across different impairment levels as when compared to those afforded by training exclusively in the physical environment. For instance, when $\sigma_{rt} = 0.1$, the proposed framework achieves a throughput of 80.98 kbits/frames, 79.86 kbits/frames, and 77.58 kbits/frames for scenario 1, scenario 2, and scenario 3, respectively. Fig. 5d also intuitively shows that with an increase in the impairment levels, the achievable throughput decreases.

[9]In our simulations, physical throughput refers to the throughput obtained by applying the selected transmit power, modulation, and coding schemes to the physical IoT network.

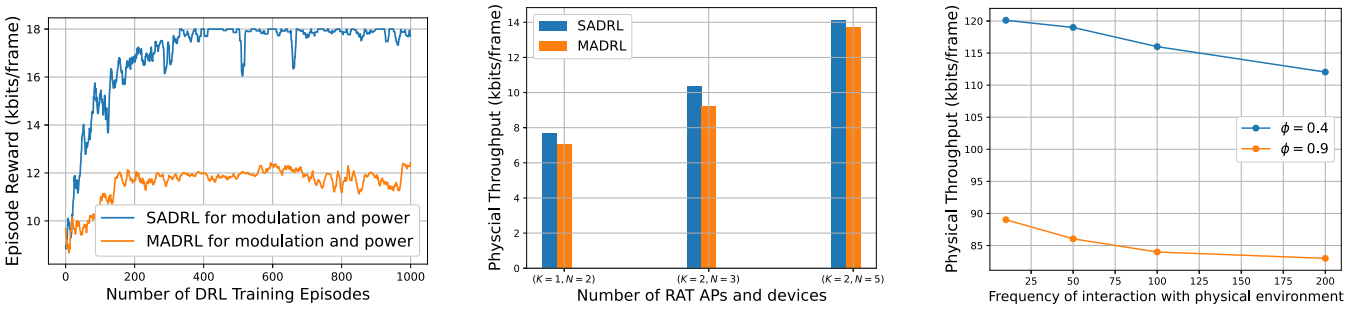
E. ADVANTAGE OF MADRL FRAMEWORK COMPARED TO SADRL FRAMEWORK

1) CONVERGENCE OF SADRL AND MADRL FOR DIFFERENT NUMBER OF EPISODES

In this section, we evaluate the performance of a MADRL framework against that of a SADRL framework. To this end, the coding rate for all devices is fixed at $\frac{1}{2}$. The SADRL framework is designed such that a centralized DRL agent receives state information from all devices and performs joint action selection. Given the high computational complexity of SADRL, we consider a scenario with 2 RAT APs and 3 IoT devices. Fig. 6a plots the throughput obtained in the DT network, where the interaction with the physical environment occurs every 100 episodes. As shown in Fig. 6a, the SADRL framework converges to a higher throughput than the proposed MADRL framework in this specific scenario. The throughput gain of SADRL is intuitive, as SADRL considers global state information and jointly determines the actions of all devices in the network. However, the action space for the SADRL framework increases exponentially with the number of devices. For instance, with 2 RAT APs and 3 devices, the action space size is $(|\mathcal{M}_m| \times |\mathcal{P}|)^3 = 4096$. With the growth in the number of devices, this exponential increase in action space size imposes severe challenges, including excessive computational complexity and memory requirements to explore and optimize the extensive action space, making the SADRL impractical for large-scale multi-RAT IoT networks.

2) PERFORMANCE COMPARISON OF MADRL VS. SADRL UNDER VARYING NUMBERS OF RAT APS AND IOT DEVICES

Fig. 6b illustrates the physical throughput achieved by SADRL and MADRL under varying numbers of RAT APs and IoT devices. As shown, SADRL consistently achieves higher throughput than MADRL across different configurations. For instance, when $K = 2$ and $N = 5$, SADRL attains a throughput of 14.13 kbits/frame, compared to 13.73 kbits/frame for MADRL. This performance advantage can be attributed to the centralized nature of SADRL, which



(a) Episode reward vs. #episodes. (b) Throughput vs. #devices and #RAT-APs. (c) Throughput vs. #DT-PHY interactions.

Fig. 6. Comparison between SADRL and MADRL schemes and impact of the number of DT-PHY interactions.

jointly determines the link-adaptation parameters for all IoT links. However, as the number of devices increases, the joint action space in SADRL grows exponentially, making training and decision making more challenging. For example, with just 5 devices and 4 power levels and 4 modulation levels, the joint action space size becomes $(4 \times 4)^5 = 1,048,576$. Furthermore, Fig. 6b shows that the performance gap between SADRL and MADRL narrows as the networks scales. In contrast to SADRL, the MADRL framework adopts a decentralized learning paradigm in which decision making is distributed among agents. Each agent (i.e., IoT link) in MADRL framework operates within a relatively small local action space based solely on local observations, without requiring information exchange or signaling with neighboring agents. This design enhances scalability, accelerates convergence, and reduces computational overhead with a reasonable loss of optimality, all of which make MADRL well-suited for large-network settings.

F. IMPACT OF INTERACTION FREQUENCY WITH PHYSICAL ENVIRONMENT

In this section, we train our algorithm with $Z_K \in \{10, 50, 100, 200\}$, meaning the algorithm interacts with the physical environment every 10, 50, 100, or 200 DRL training episodes. Fig. 6c shows the throughput in the physical environment vs. the interaction frequencies for different values of ϕ. As can be seen in Fig. 6c, decreasing the interaction frequency with the physical environment leads to a reduction in throughput. This outcome is expected, as the DRL agent has a limited understanding of the physical environment, resulting in less accurate knowledge of its characteristics. For instance, considering $\phi = 0.4$, the proposed framework achieves a throughput of 120.11 kbits/frame and 112.04 kbits/frame when $Z_k = 10$ and $Z_k = 200$, respectively. However, increasing interaction frequency also incurs higher communication overhead. Therefore, it is crucial to select an optimal interaction frequency. Based on this trade-off, we set $Z_K = 100$ in our experiments, i.e., the DTN periodically collects feedback from the physical network after every 100 episodes of DRL training.

In addition, Fig. 6c shows that increasing ϕ decreases the sum throughput. For instance, if $Z_k = 50$ is considered, the proposed framework achieves a throughput of 118.98 kbits/frames and 86.04 kbits/frames when $\phi = 0.4$ and $\phi = 0.9$, respectively. This outcome is expected, as increasing ϕ amplifies uncertainty in modeling ACI and HWI in the DTN, requiring more frequent interactions with the physical network to refine the DQN model.

G. PERFORMANCE COMPARISON AGAINST BENCHMARK SCHEMES

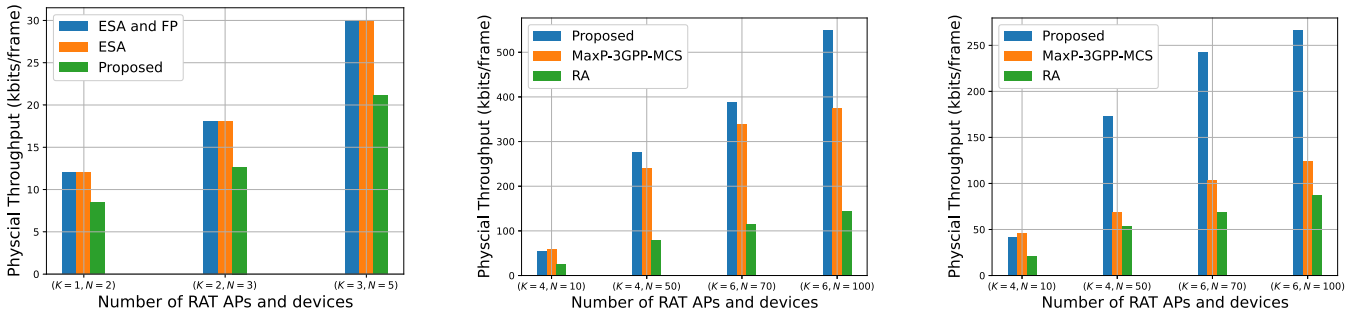
1) COMPARISON WITH THE ESA AND ESA-FP BENCHMARKS

In Fig. 7a, we compare performance of the proposed resource allocation algorithm against the exhaustive-search-based optimal schemes. Due to the significant complexity of performing exhaustive search, this evaluation is conducted in a small-scale scenario with a limited number of IoT devices and RAT APs. Both benchmark schemes achieve the same optimal throughput for the considered small-scale scenarios ($(K = 1, N = 2)$, $(K = 2, N = 3)$, and $(K = 3, N = 5)$), while the proposed DRL-based method attains approximately 70.4% of this optimum.

It is noteworthy that both ESA approaches require large computational complexity even for the small network settings. For example, with only three devices, the exhaustive search iterates over $4^3 = 64$ modulation combinations and $3^3 = 27$ coding-rate combinations, yielding $64 \times 27 = 1728$ unique (m, c) configurations. For each of these, FP must be applied to optimize transmit power, further increasing complexity. In contrast, ESA over (m, p, c) requires $1728 \times 64 = 110{,}592$ evaluations, making the complexity orders of magnitude higher. Consequently, for large-scale IoT networks, both ESA-FP and ESA methods become infeasible due to prohibitively increasing computational cost. This observation underscores that our proposed DRL solution achieves a suitable trade-off between performance and computational complexity.

2) COMPARISON WITH MAXP-3GPP-MCS AND RA BENCHMARKS

In Fig. 7b and 7c, we compare the proposed framework with state-of-the-art algorithms under varying numbers of RAT APs and IoT devices. The benchmarks considered are the RA allocation scheme and the MaxP-3GPP-MCS.



(a) Comparison with exhaustive search. (b) Comparison with benchmarks for $\sigma_{rt} = 0.05$. (c) Comparison with benchmarks for $\sigma_{rt} = 0.2$.

Fig. 7. Throughput comparison between the proposed scheme, exhaustive search, and benchmark link adaptation schemes.

As the number of RAT APs and IoT devices increases, distinct performance trends emerge. For small-scale settings with few IoT links, the proposed scheme achieves slightly lower throughput than MaxP-3GPP-MCS. This is because with only a few RAT IoT links, transmissions can be scheduled over non-adjacent RRBs, making ACI negligible. In the absence of ACI, the MaxP-3GPP-MCS strategy can achieve near-optimal sum throughput performance. However, as the number of devices and RAT-APs increases, the number of adjacent interfering links also grows. In this context, the proposed scheme outperforms the benchmark schemes by intelligently optimizing the link adaptation parameters to mitigate the deleterious impact of ACI. For instance, in Fig. 7b, when $N = 70$ and $K = 6$, the proposed framework's sum throughput is 49.01 kbits/frame and 274.21 kbits/frame higher than that of the MaxP-3GPP-MCS and RA schemes, respectively. Likewise, the proposed framework achieves a 46% higher sum throughput compared to the MaxP-3GPP-MCS scheme, evaluated for $K = 6$ RAT APs and $N = 100$ IoT devices.

The performance advantage of our framework becomes more pronounced as the HWI level increases (see Fig. 7c). Notably, higher impairment levels increase the denominator of the SINR expression. In such scenarios, judicious selection of transmit power, modulation, and coding schemes becomes imperative to enhance the sum throughput. Thanks to its DRL-based link adaptation strategy, which intelligently adjusts parameters by leveraging both current and past states of IoT links, our proposed scheme achieves notable performance gain over the benchmarks for large HWI levels. For instance, when $N = 70$ and $K = 6$, the proposed framework's sum throughput is 138.46 kbits/frame and 173.52 kbits/frame higher than that of the MaxP-3GPP-MCS and RA schemes, respectively. Moreover, under high HWI levels, the proposed framework achieves 114.16% higher sum throughput than the MaxP-3GPP-MCS scheme, in a system with $K = 6$ RAT APs and $N = 100$ IoT devices. Overall, the results confirm robustness of our learning-based framework in dense IoT deployments with practical impairments caused by ACI and HWI.

TABLE 4. Required time vs. different numbers of IoT devices.

	$N = 1$	$N = 20$	$N = 50$	$N = 100$
Time required (ms)	10	12.5	13.9	15.1

H. RUN TIME DURATION OF THE PROPOSED ALGORITHM

The DQN model is saved after training for 2000 episodes, and subsequent testing is conducted using the trained model in a testing scenario. Table 4 presents the average decision making time (i.e., time required to compute the power level and select the MCS) for a single device and a single TS. This duration includes the time required for data collection (e.g., device locations, ACI and channel gain estimation, SINR computation, and PF calculation), device clustering and RRB scheduling, and the final link adaptation decisions. Consistently with our expectations, the execution time increases with the number of devices, since the clustering and scheduling processes must accommodate all existing devices. For instance, the proposed framework achieves an execution time of 12.5 ms for $N = 20$ IoT devices and 15.1 ms for $N = 100$ IoT devices. These observations indicate that the proposed scheme enables efficient resource allocation with minimal execution delay, making it well-suited for timely and effective decision making in multi-RAT IoT networks.

I. PERFORMANCE EVALUATION OF THE PROPOSED ALGORITHM UNDER NON-STATIONARY ENVIRONMENT

To evaluate the continual learning capability of the proposed DRL agent, we simulate a non-stationary environment where key parameters, including shadowing (σ_s) and the non-linearity uncertainty parameter ϕ, are kept constant for a block of $Z_m \in \{10, 30, 50\}$ consecutive episodes and then randomly re-sampled for each IoT link at the beginning of the next block. The shadowing component σ_s is independently sampled for each IoT link from a normal distribution with zero mean and standard deviation of 4, 6, or 8 dB, while the non-linearity uncertainty parameter ϕ is randomly selected from the set $\{0, 0.4, 0.6, 0.9\}$. This piecewise-stationary design induces controlled distribution

TABLE 5. A qualitative comparison of performance and complexity trade-offs across different schemes.

Criteria	ESA-FP	MaxP-3GPP-MCS	RA	Proposed
Complexity	Very High	Low	Low	Moderate
Execution Time	Very long	Low	Low	Low
Scalability	Poor	Excellent	Poor	Excellent
Adaptability	Low	High	Moderate	High
Optimality	Optimal	Suboptimal	Suboptimal	Near-optimal

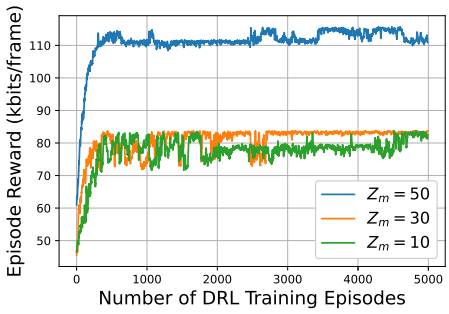


Fig. 8. Physical throughput vs. DRL training episodes in non-stationary environment.

shifts, thus creating a globally non-stationary environment over the entire training horizon. The agent interacts with this environment and updates its policy incrementally using experiences drawn from a shared replay memory. Here, smaller values of Z_m correspond to more frequent environment changes, creating a rapidly varying scenario, while larger values of Z_m represent slower changes, allowing the agent more time to adapt before the next distribution shift. Fig. 8 shows that the proposed framework converges under all considered scenarios, while intuitively generating the highest and lowest reward for slow and fast varying non-stationary environments. Consequently, the proposed framework allows the DRL agent to continuously adapt to changing channel conditions while retaining knowledge from previously encountered states.

J. DISCUSSION ON PERFORMANCE–COMPLEXITY TRADEOFF

This subsection analyzes the tradeoff between performance and complexity to identify the most suitable scheme for large-scale IoT deployments. Five key criteria, such as **complexity**, **execution time**, **scalability**, **adaptability**, and **optimality** are considered. Table 5 provides a qualitative comparison across the considered schemes. The ESA-FP scheme achieves the highest optimality but comes with very high complexity and execution time, making it impractical beyond small-scale settings. The MaxP-3GPP-MCS and RA baselines offer low complexity but at the cost of reduced optimality, thus serving as feasible but performance-limited solutions. By contrast, the proposed continual MADRL-based method achieves a balanced trade-off across the five criteria. While its computational complexity is slightly higher than that of simple heuristics, it provides a near-optimal solution with reasonably low execution time, remains scalable as the network size increases, and adapts to dynamic IoT conditions with reduced overhead. These features make the proposed scheme instrumental for addressing multi-RAT IoT coexistence in the presence of inevitable ACI and HWIs.

VII. CONCLUSION

In this paper, we proposed a DT-enhanced continual DRL framework for radio resource allocation in multiple RAT IoT networks. A radio resource optimization problem was devised aimed at maximizing network sum throughput by jointly optimizing transmit power allocation and MCS selection while considering the challenges posed by ACI and HWI arising from low-cost RF front-end components. By treating each IoT link as an independent learning agent, we devised an MARDL approach to solve this optimization problem in a distributed and computationally efficient manner. Unlike conventional methods, our innovative MADRL training framework uses a DTN to enhance learning efficiency and mitigate real-world deployment risks. Specifically, our framework enables DRL model training within a DTN, with periodic updates based on data collected from the physical network. Such continual learning approach improves the model's adaptability and accuracy in dynamic environments. Our simulation results confirm the following key findings: (i) The proposed MADRL achieves performance close to SADRL, with throughput differences within 2.8–10.9%. (ii) The continual learning strategy provides up to 14.4% higher throughput compared to DT-only training. (iii) The proposed scheme achieves near-optimal performance—within 70.4% of the ESA-FP benchmark—while maintaining scalability for large-scale IoT deployments. (iv) Compared to the MaxP-3GPP-MCS baseline, the proposed framework achieves up to 46% and 114.16% higher sum throughput in dense IoT networks under two different HWI levels.

APPENDIX

By definition, an optimization problem is NP-hard if any instance of a known NP-complete problem can be reduced to it in polynomial time. Of note, the MCS for the coexisting IoT links can be typically determined using a traditional lookup table approach (e.g., [52, Table 7.2.3-1]). This method, which maps a reported CQI to a suitable MCS,

operates in polynomial time and does not account for the detrimental effects of ACI and HWI-induced distortions. Assuming that the MCS selection for all IoT devices is obtained via this lookup table, the original problem P0 can be reduced to a transmit power allocation problem, denoted by $\hat{\text{P0}}$ (see Eq. (39)):

$$\hat{\text{P0}} \max_{\boldsymbol{P}^t_{n,k}} \sum_{n=1}^{N} R^t_n$$
$$\text{s.t. C1: } P_{min} \leq P^t_{n,k} \leq P_{max}, \forall n \in \mathcal{N} \qquad (39)$$

Here, $\hat{\text{P0}}$: aims to maximize the network's total capacity via power allocation only. Since the reduction from P0 to $\hat{\text{P0}}$ involves only the polynomial-time lookup table for MCS determination, this reduction is itself polynomial.

It is known that optimizing a sum of functions of ratios (as in $\hat{\text{P0}}$) is a classic NP-complete problem [53]. Accordingly, P0 can be reduced in polynomial time to an NP-complete problem. Hence, P0 must be an NP-hard optimization problem. □

REFERENCES

[1] K. Shafique, B. A. Khawaja, F. Sabir, S. Qazi, and M. Mustaqim, "Internet of Things (IoT) for next-generation smart systems: A review of current challenges, future trends and prospects for emerging 5G-IoT scenarios," *IEEE Access*, vol. 8, pp. 23022–23040, 2020.

[2] J. G. Andrews et al., "What will 5G be?," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 6, pp. 1065–1082, Jun. 2014.

[3] H. Mohammadi, W. AlQwider, T. F. Rahman, and V. Marojevic, "AI-driven demodulators for nonlinear receivers in shared spectrum with high-power blockers," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Austin, TX, USA, Apr. 2022, pp. 644–649.

[4] Z. Chu et al., "RIS assisted wireless powered IoT networks with phase shift error and transceiver hardware impairment," *IEEE Trans. Commun.*, vol. 70, no. 7, pp. 4910–4924, Jul. 2022.

[5] N. B. Mohamed, M. Z. Hassan, and G. Kaddoum, "RSMA-enabled interference management for industrial Internet of Things networks with finite blocklength coding and hardware impairments," *IEEE Trans. Mach. Learn. Commun. Netw.*, vol. 2, pp. 1319–1340, 2024.

[6] A. E. Jayati and M. Sipan, "Impact of nonlinear distortion with the Rapp model on the GFDM system," in *Proc. 3rd Int. Conf. Vocational Educ. Electr. Eng. (ICVEE)*, Surabaya, Indonesia, Oct. 2020, pp. 1–5, doi: 10.1109/ICVEE50212.2020.9243295.

[7] M. M. Shammasi and S. M. Safavi, "Performance of a predistorter based on saleh model for OFDM systems in HPA nonlinearity," in *Proc. 14th Int. Conf. Adv. Commun. Technol. (ICACT)*, Feb. 2012, pp. 148–152.

[8] J. C. Pedro and S. A. Maas, "A comparative overview of microwave and wireless power-amplifier behavioral modeling approaches," *IEEE Trans. Microw. Theory Techn.*, vol. 53, no. 4, pp. 1150–1163, Apr. 2005.

[9] K. Davaslioglu, S. Kompella, T. Erpek, and Y. E. Sagduyu, "Continual deep reinforcement learning to prevent catastrophic forgetting in jamming mitigation," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, Washington, DC, USA, Oct. 2024, pp. 740–745.

[10] Q. Guo, F. Tang, and N. Kato, "Resource allocation for aerial assisted digital twin edge mobile network," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 10, pp. 3070–3079, Oct. 2023.

[11] M. Kim and I.-Y. Ko, "An efficient resource allocation approach based on a genetic algorithm for composite services in IoT environments," in *Proc. IEEE Int. Conf. Web Services*, Jun. 2015, pp. 543–550.

[12] A. Alwarafy, M. Abdallah, B. S. Ciftler, A. Al-Fuqaha, and M. Hamdi, "Deep reinforcement learning for radio resource allocation and management in next generation heterogeneous wireless networks: A survey," 2021, *arXiv:2106.00574*.

[13] T. T. Nguyen, N. D. Nguyen, and S. Nahavandi, "Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications," *IEEE Trans. Cybern.*, vol. 50, no. 9, pp. 3826–3839, Sep. 2020.

[14] M. P. Mota, D. C. Araujo, F. H. C. Neto, A. L. F. de Almeida, and F. R. Cavalcanti, "Adaptive modulation and coding based on reinforcement learning for 5G networks," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2019, pp. 1–6.

[15] R. Bruno, A. Masaracchia, and A. Passarella, "Robust adaptive modulation and coding (AMC) selection in LTE systems using reinforcement learning," in *Proc. IEEE 80th Veh. Technol. Conf. (VTC-Fall)*, Sep. 2014, pp. 1–6.

[16] H. V. Vu, M. Farzanullah, Z. Liu, D. H. N. Nguyen, R. Morawski, and T. Le-Ngoc, "Multi-agent reinforcement learning for channel assignment and power allocation in platoon-based C-V2X systems," 2020, *arXiv:2011.04555*.

[17] Z. Xu, Y. Wang, J. Tang, J. Wang, and M. C. Gursoy, "A deep reinforcement learning based framework for power-efficient resource allocation in cloud RANs," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Paris, France, May 2017, pp. 1–6.

[18] L. Zhang, J. Tan, Y.-C. Liang, G. Feng, and D. Niyato, "Deep reinforcement learning-based modulation and coding scheme selection in cognitive heterogeneous networks," *IEEE Trans. Wireless Commun.*, vol. 18, no. 6, pp. 3281–3294, Jun. 2019.

[19] K. I. Ahmed and E. Hossain, "A deep Q-learning method for downlink power allocation in multi-cell networks," 2019, *arXiv:1904.13032*.

[20] F. Meng, P. Chen, L. Wu, and J. Cheng, "Power allocation in multi-user cellular networks: Deep reinforcement learning approaches," *IEEE Trans. Wireless Commun.*, vol. 19, no. 10, pp. 6255–6267, Oct. 2020.

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

[22] A. Parsa, N. Moghim, and P. Salavati, "Joint power allocation and MCS selection for energy-efficient link adaptation: A deep reinforcement learning approach," *Comput. Netw.*, vol. 218, Dec. 2022, Art. no. 109386.

[23] S. Jamshidiha, V. Pourahmadi, A. Mohammadi, and M. Bennis, "Link-level throughput maximization using deep reinforcement learning," *IEEE Netw. Lett.*, vol. 2, no. 3, pp. 101–105, Sep. 2020.

[24] Z. Zhang, Y. Huang, C. Zhang, Q. Zheng, L. Yang, and X. You, "Digital twin-enhanced deep reinforcement learning for resource management in networks slicing," *IEEE Trans. Commun.*, vol. 72, no. 10, pp. 6209–6224, Oct. 2024.

[25] Y. Cui, T. Lv, W. Ni, and A. Jamalipour, "Digital twin-aided learning for managing reconfigurable intelligent surface-assisted, uplink, user-centric cell-free systems," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 10, pp. 3175–3190, Oct. 2023.

[26] M. Elloumi, M. Z. Hassan, and G. Kaddoum, "Spectrum sharing in Internet-of-Vehicles networks: Digital twin-empowered proactive interference management approach," *IEEE Trans. Netw. Service Manage.*, vol. 22, no. 4, pp. 3228–3248, Aug. 2025, doi: 10.1109/TNSM.2025.3541977.

[27] M. Elloumi, G. Kaddoum, M. Zoheb Hassan, and B. Selim, "Digital-twin-empowered interference management for multihop Internet of Vehicles networks over millimeter wave bands," *IEEE Internet Things J.*, vol. 12, no. 11, pp. 17807–17827, Jun. 2025, doi: 10.1109/JIOT.2025.3540750.

[28] M. Haider, I. Ahmed, Z. Hassan, T. J. O'Shea, L. Liu, and D. B. Rawat, "Digital twin enabled site specific channel precoding: Over the air CIR inference," 2025, *arXiv:2501.16504*.

[29] M. Sarker, M. Zoheb Hassan, and G. Kaddoum, "Joint user association and bandwidth assignment for digital twin-assisted multi-RAT networks," 2025, *arXiv:2505.04829*.

[30] ITU.(2024). *Report on Activities and Proposals for the World Radiocommunication Conference 2024 (WRC-24)*. [Online]. Available: https://www.itu.int/wrc-23/

[31] L. Baldesi, F. Restuccia, and T. Melodia, "ChARM: NextG spectrum sharing through data-driven real-time O-RAN dynamic control," in *Proc. IEEE Conf. Comput. Commun. (IEEE INFOCOM)*, May 2022, pp. 240–249.

[32] Y. S. Nasir and D. Guo, "Multi-agent deep reinforcement learning for dynamic power allocation in wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2239–2250, Oct. 2019.

[33] T. Li et al., "Generative AI empowered network digital twins: Architecture, technologies, and applications," *ACM Comput. Surv.*, vol. 57, no. 6, pp. 1–43, Jun. 2025.

[34] Claus Hetting, *Wi-Fi Industry Scores Important 6 GHz Victory at WRC-23*. Accessed: Nov. 2024. [Online]. Available: https://wifinowglobal.com/news-blog/our-take-wi-fi-industry-scores-important-6-ghz-victory-at-wrc-23-the-rest-is-up-to-us/

[35] N. Naderializadeh, J. J. Sydir, M. Simsek, and H. Nikopour, "Resource management in wireless networks via multi-agent deep reinforcement learning," *IEEE Trans. Wireless Commun.*, vol. 20, no. 6, pp. 3507–3523, Jun. 2021.
[36] L. Liang, J. Kim, S. C. Jha, K. Sivanesan, and G. Y. Li, "Spectrum and power allocation for vehicular communications with delayed CSI feedback," *IEEE Wireless Commun. Lett.*, vol. 6, no. 4, pp. 458–461, Aug. 2017.
[37] *Study on Channel Model for Frequencies From 0.5 to 100 GHz*, document TR 38.901, 3GPP, 2020.
[38] *Derivation of a Block Edge Mask for Terminal Stations in the 2.6 GHz Frequency Band (2500–2690 MHz)*, ECC REPORT, Dublin, Jan. 2009.
[39] Y. Chen, Z. Yang, J. Zhang, and M.-S. Alouini, "Further results on detection and channel estimation for hardware impaired signals," *IEEE Trans. Commun.*, vol. 69, no. 11, pp. 7167–7179, Nov. 2021.
[40] M. Matthaiou, A. Papadogiannis, E. Bjornson, and M. Debbah, "Two-way relaying under the presence of relay transceiver hardware impairments," *IEEE Commun. Lett.*, vol. 17, no. 6, pp. 1136–1139, Jun. 2013.
[41] A. Baghel, D. Singh, A. S. Parihar, V. Bhatia, N. Rajatheva, and M. Latva-Aho, "Robustness of NOMA in HetNets: Impact of hardware impairments and imperfect CSI," *IEEE Wireless Commun. Lett.*, vol. 14, no. 12, pp. 3887–3891, Dec. 2025, doi: 10.1109/LWC.2025.3605209.
[42] S. Javed, O. Amin, B. Shihada, and M.-S. Alouini, "Improper Gaussian signaling for hardware impaired multihop full-duplex relaying systems," *IEEE Trans. Commun.*, vol. 67, no. 3, pp. 1858–1871, Mar. 2019.
[43] M. Soleymani, C. Lameiro, I. Santamaria, and P. J. Schreier, "Improper signaling for SISO two-user interference channels with additive asymmetric hardware distortion," *IEEE Trans. Commun.*, vol. 67, no. 12, pp. 8624–8638, Dec. 2019.
[44] S. Mashhadi, N. Ghiasi, S. Farahmand, and S. M. Razavizadeh, "Deep reinforcement learning based adaptive modulation with outdated CSI," *IEEE Commun. Lett.*, vol. 25, no. 10, pp. 3291–3295, Oct. 2021.
[45] J. Bauwens, P. Ruckebusch, S. Giannoulis, I. Moerman, and E. D. Poorter, "Over-the-air software updates in the Internet of Things: An overview of key principles," *IEEE Commun. Mag.*, vol. 58, no. 2, pp. 35–41, Feb. 2020.
[46] J. G. Andrews, T. E. Humphreys, and T. Ji, "6G takes shape," *IEEE BITS Inf. Theory Mag.*, vol. 4, no. 1, pp. 2–24, Mar. 2024.
[47] H. Tong et al., "Continual reinforcement learning for digital twin synchronization optimization," *IEEE Trans. Mobile Comput.*, vol. 24, no. 8, pp. 6843–6857, Aug. 2025.
[48] Z. Han, D. Niyato, W. Saad, and T. Başar, *Game Theory for Next Generation Wireless and Communication Networks: Modeling, Analysis, and Design*. Cambridge, U.K.: Cambridge Univ. Press, 2019.
[49] J. Cui, Y. Liu, and A. Nallanathan, "Multi-agent reinforcement learning-based resource allocation for UAV networks," *IEEE Trans. Wireless Commun.*, vol. 19, no. 2, pp. 729–743, Feb. 2020.
[50] X. Mu, X. Zhao, and H. Liang, "Power allocation based on reinforcement learning for MIMO system with energy harvesting," *IEEE Trans. Veh. Technol.*, vol. 69, no. 7, pp. 7622–7633, Jul. 2020.
[51] H.-H. Chang, "Deep reinforcement learning for next generation wireless networks with echo state networks," Ph.D. dissertation, Dept. Elect. Eng., Virginia Polytech. Inst. State Univ., Blacksburg, VA, USA, Aug. 2021.
[52] *Technical Specification Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA); Physical Layer Procedures (Release 18)*, document TS 36.213 V18.3.0, 3GPP, Dec. 2024.
[53] K. Shen and W. Yu, "Fractional programming for communication systems—Part I: Power control and beamforming," *IEEE Trans. Signal Process.*, vol. 66, no. 10, pp. 2616–2630, May 2018.

NAHED BELHADJ MOHAMED received the B.E. degree in telecommunication engineering from the Higher School of Communication of Tunis (SUP'COM), Ariana, Tunisia, in 2018. She is currently pursuing the Ph.D. degree in electrical engineering program with École de Technologie Supérieure (ÉTS), Montreal, QC, Canada. Her research interests include wireless communications, the Internet of Things, radio resource management, and the application of machine learning in physical layer communications. She serves as a reviewer for IEEE ICC, IEEE ICMLCN, IEEE GLOBECOM, and IEEE INTERNET OF THINGS JOURNAL.

GEORGES KADDOUM (Senior Member, IEEE) received the bachelor's degree in electrical engineering from the École Nationale Supérieure de Techniques Avancées, Brest, France, the M.S. degree in telecommunications and signal processing from the Université de Bretagne Occidentale and Télécom Bretagne, Brest, in 2005, and the Ph.D. degree (Hons.) in signal processing and telecommunications from the National Institute of Applied Sciences, University of Toulouse, Toulouse, France, in 2009. He is currently a Professor and the Research Director of the Resilient Machine Learning Institute, and the Tier 2 Canada Research Chair with the École de Technologie Supérieure (ÉTS), Canada. He has a prolific publication record with more than 300 journal articles and conference papers, two chapters in books, and eight pending patents. His recent research interests include wireless communication networks, tactical communications, resource allocations, and network security. He has received several accolades, including the Best Paper Awards at prestigious conferences, such as IWCMC 2023, PIMRC 2017, and WiMob 2014. Notably, he has been recognized with the IEEE Transactions on Communications Exemplary Reviewer Award in 2019, 2017, and 2015, the Research Excellence Award from the Université du Québec in 2018, and the Research Excellence Award from ÉTS in 2019 and 2025. His outstanding contributions were further acknowledged with the 2022 IEEE Technical Committee on Scalable Computing Award for Excellence (Middle Career Researcher), the 2023 MITACS Award for Exceptional Leadership, the 2025 NSERC Synergy Award, and 2025 the Gold Medal Award from Engineers Canada. He served as an Associate Editor for IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY and IEEE COMMUNICATIONS LETTERS. He is serving as an Area Editor for IEEE TRANSACTIONS ON MACHINE LEARNING IN COMMUNICATIONS AND NETWORKING and an Editor for IEEE TRANSACTIONS ON COMMUNICATIONS.

MD. ZOHEB HASSAN (Member, IEEE) received the Ph.D. degree from the Electrical and Computer Engineering Department, The University of British Columbia, Vancouver, BC, Canada. He is currently an Assistant Professor with the Department of Electrical and Computer Engineering, Université Laval, Canada. Prior to joining Université Laval, he was a Senior Post-Doctoral Research Fellow with the École de Technologie Superiéure (ETS) and a Research Assistant Professor with the ECE Department, Virginia Tech, Blacksburg, VA, USA. He has authored and co-authored more than 45 journal articles and 25 conference papers on digital twin, radio resource optimization, spectrum sharing, and optical wireless communications in renowned journals and conferences of the IEEE Communications Society. He was a recipient of the prestigious Natural Sciences and Engineering Research Council of Canada (NSERC) Post-Doctoral Fellowship grant and recognized as the top-ranked applicant. He serves/served as an Editor for IEEE TRANSACTIONS ON COMMUNICATIONS, an Associate Editor for IEEE INTERNET OF THINGS OF JOURNAL, and a Guest Editor for IEEE OPEN JOURNAL OF THE COMMUNICATIONS SOCIETY.