IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received January 28, 2019, accepted February 13, 2019, date of publication February 21, 2019, date of current version March 25, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2900445

Artificial Intelligence-Based Handoff Management for Dense WLANs: A Deep Reinforcement Learning Approach

ZIJUN HAN, TAO LEI, ZHAOMING LU, XIANGMING WEN, WEI ZHENG, (Member, IEEE), AND LINGCHAO GUO
Beijing Key Laboratory of Network System Architecture and Convergence, School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China
Corresponding author: Zhaoming Lu (lzy0372@bupt.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61671073, and in part by the WLAN Achievement Transformation Based on Software-Defined Networking (SDN) of the Beijing Municipal Commission of Education under Grant 201501001.

ABSTRACT So far, the handoff management involved in the wireless local area network (WLAN) has mainly fallen into the handoff mechanism and the decision algorithm. The traditional handoff mechanism generates noticeable delays during the handoff process, resulting in discontinuity of service, which is more evident in dense WLANs. Inspired by software-defined networking (SDN), prior works put forward many seamless handoff mechanisms to ensure service continuity. With respect to the handoff decision algorithm, when to trigger handoff and which access point to reconnect to, however, are still tricky problems. In this paper, we first design a self-learning architecture applicable to the SDN-based WLAN frameworks. Along with it, we propose DCRQN, a novel handoff management scheme based on deep reinforcement learning, specifically deep Q-network. The proposed scheme enables the network to learn from actual users' behaviors and network status from scratch, adapting its learning in time-varying dense WLANs. Due to the temporal correlation property, the handoff decision is modeled as the Markov decision process (MDP). In the modeled MDP, the proposed scheme depends on the real-time network statistics at the time of decisions. Moreover, the convolutional neural network and the recurrent neural network are leveraged to extract fine-grained discriminative features. The numerical results through simulation demonstrate that DCRQN can effectively improve the data rate during the handoff process, outperforming the traditional handoff scheme.

INDEX TERMS Deep reinforcement learning, handoff, SDN, WLAN.

I. INTRODUCTION

IEEE 802.11 based Wireless Local Area Network (WLAN) has gained popularity due to its simplicity of deployment and broadband connectivity [1]–[3]. With the proliferating demands for wireless access, more and more access points (APs) are deployed in a specific scenario, which may lead to a dense WLAN [4]. It is inevitable for the mobile stations (STAs) to traverse diverse basic service set (BSS) areas covered by APs. Due to the limited coverage of AP, the STAs will be subject to frequent handoffs [5].

The handoff process refers to the mechanism of messages exchanged by APs and STA, resulting in a transfer of physical layer connectivity from one AP to another [6]. The traditional WLAN handoff scheme generally takes the received signal strength (RSS) as the criterion. To be specific, the process can be divided into three logic steps: 1) *Handoff initiation*: it is based on real-time and reliable detection of RSS, e.g., the handoff will be triggered when the RSS falls below a pre-configured threshold. 2) *Discovery process*: the STA performs channel scanning for target APs by listening to the beacon messages or dwelling on each channel for a while to wait for probe response. 3) *Re-authentication*: it involves an authentication and a re-association to the target AP. The wireless link is finally transferred from the previous AP to the target one [7]. During the handoff process, no data frame can be transmitted, and the interruption is called handoff delay [8]. In the initial design of the IEEE 802.11 protocol, the primary is to provide broadband wireless access for users through a single AP. The 802.11 standard lets STAs make AP

The associate editor coordinating the review of this manuscript and approving it for publication was Tallha Akram.

associations on the basis of locally made decisions. However, there are no mechanisms for ensuring the seamless handoff. Typically the handoff delay in WLAN is approximately 300 ms or more [6]. The real-time services represented by Voice over Internet Protocol (VoIP) are so demanding on the instantaneity of transmission that the handoff delay in WLAN will seriously degrade the user experience (UE).

Recently, software-defined networking (SDN), which allows administrators to manage network services through the abstraction of lower level functionality, has been widely used in the field of wireless communication [9]. SDN offers innovation by providing centralized view of entire network where intelligence is shifted towards SDN controller. The controller interacts with data forwarding nodes (switches) to simplify the network management. It operates on OpenFlow protocol [10] and deploys forwarding rules into programmable switches in order to alter network behavior in real-time [11]. In SDN-based WLAN, a logically centralized controller is employed to manage and configure the network by programming network applications. The SDN controller has two interfaces: southbound and northbound. The southbound interface is standardized [12] and it controls data plane in the network. The northbound interface is determined for services in the form of applications on the top of the SDN controller. These applications are independent from infrastructure and network devices. The controller has a global view of the network so that the real-time network status and statistics can be detected through southbound interface. From the perspective of the network administrators, various STAs connected to the same physical AP are regarded as a set of logically isolated STAs which are connected to diverse ports of a switch [9]. Mobile STAs, therefore, are enabled to handoff seamlessly by tracking the location of users and migrating the virtual APs accordingly [13]. Due to the changes in channel conditions, along with the dynamics of each new connection or disconnection, the system state of WLAN evolves over time. The seamless handoff mechanism ensures the continuity of service, but the AP connected previously might not provide the best service at the next instant during data delivery. Therefore, with respect to the handoff decision algorithm, when to trigger handoff and which AP to reconnect to are still tricky problems.

With the RSS in MAC (Media Access Control) layer as a case, it is widely adopted for handoff decision-making in commodity 802.11 devices [14]. In the *handoff initiation*, the parameters (e.g., the pre-configured thresholds) rely heavily on specific network scenarios, without universality or adaptability. In various WLAN scenarios, it is generally true that one handoff scheme exhibits diverse performance. The personnel responsible for network planning and deployment have to set corresponding parameters manually according to the realistic environment. However, the parameters set in a previous scenario fail to apply to new one, leading to the extra rounds of network testing and configuration, which enormously augments the deployment workloads [15]. On other hand, it is hard to accurately characterize the STA's state by taking into account merely two or less indicators. For SDN-based WLAN, there is no channel scanning in *discovery process*. The generality of decision logic, however, becomes an obstacle to inferring the AP status, e.g., RSS or its variants based scheme could not reflect the reliability of wireless link due to the lack of the interference expression. Mobile STAs which are programmed to access to the network with the largest RSS may drain the single AP's bandwidth [16], [17]. In addition, the network, by itself, may be subject to changes, e.g., a certain AP fails to work properly or new APs are deployed to enhance coverage areas. Therefore, in SDN-based WLANs with dynamic variability, traditional handoff scheme cannot adjust parameters or decision logic adaptively, which puts forward challenges to operate the handoff management effectively [18]. The wireless link is often inferior in quality during the handoff process. Unreasonable decision-making may give rise to a low data rate, which will also degrade the UE to a certain extent [19]. In brief, the fundamental problem with such an oversimplified strategy lies in its inability to ensure the effectiveness and reliability of handoff, which may even lead to handoff failure [20]. In this context, it is of paramount importance to optimize the handoff algorithm with the goal of data rate maximization.

A. MOTIVATIONS

Reinforcement learning (RL) puts forward a promising solution to the aforementioned problems. By enabling an agent to learn in interaction with the environment, an optimal policy capable of maximizing the long-term accumulated returns is eventually obtained [21]. RL based handoff scheme has obvious superiorities in terms of generality to various WLAN scenarios and adaptability to network transformations. This significant performance gain comes at the cost of the self-regulated learning individual going through a learning phase to obtain the decision policy. RL also embodies the properties of on-line learning [22]. In the case of network deployment changes, previously learned policy is no longer the optimal one. The accumulated returns will be potentially declined. On-line learning can definitely perceive this declination and the optimal policy will be refreshed through another round of training. Q-learning, a branch of RL, has been applied to the field of wireless communication to deal with the problems related to policy optimization [17], [23], [24]. However, Q-learning is in demand for a two-dimensional Q-table to store all affirmatory states and the Q-values of actions for each state in the environment [25]. It is worth noting that Q-learning is unable to make appropriate decisions on state beyond the Q-table. As for the WLAN handoff management, due to the immense state space caused by the mobile STA, it is impractical to store the Q-values of all the state-action pairs by means of Q-table. Even though the hardware could satisfy the storage constraints for states, querying for state in such a large Q-table is quite time-consuming. Therefore, it is challenging to apply Q-learning to the WLAN handoff scheme directly.

With the rapid development of artificial intelligence, machine learning represented by deep learning has made great breakthroughs in recent years. Deep learning could explore the distributed features of data by combining low-level features into more abstract high-level representations (e.g., attributes, categories or states) [26]. In 2015, Volodymyr Mnih *et al.* [27] from Google's DeepMind team combined deep learning with RL and put forward the concept of Deep Q-Network (DQN). By integrating both the extraction of deep learning and decision making of Q-learning, DQN is able to output control signals directly for high-dimensional environments. Benefits promised by DQN have addressed the problems including the state representation, the feature extraction along with the mapping of state to action. By establishing neural networks to estimate the Q-value function, the limitations of Q-learning can be eliminated eventually [27]. Since then, DQN has been widely used in various fields [28], [29]. Naparstek and Cohen [30] applied deep RL (DRL) theory to deal with the problems of accessing to distributed dynamic spectrum in multi-channel wireless networks with the goal of maximizing network utility and finally achieved desired effects. We focus on this state-of-the-art paradigm to incorporate learning for the handoff decision-making in WLAN. Moreover, the motion of STAs leads to obvious fluctuation in RSS during propagation. Note that there are significant correlations between the consecutive wireless signals for some time. Motivated by above intrinsic properties, we turn to convolutional neural network (CNN) and recurrent neural network (RNN) to extract the spatial and temporal features of wireless signals, thereby inferring the mobile velocity, moving direction or relative location of the STA [31].

B. CONTRIBUTIONS

In the paper, we propose DCRQN, a DQN-based handoff management scheme for dense WLANs. Due to the temporal correlation property, the handoff decision-making is modeled as the Markov Decision Process (MDP).In modeled MDP, the handoff scheme depends on the real-time STA's state at time of decisions. Focusing on improving the data rate during handoff process, DCRQN aims at learning the optimal handoff decision algorithm to ensure UE. The principal contributions of the paper are as follows:

- A self-learning architecture which is applicable to the SDN-based WLAN frameworks is designed. DCRQN working as a novel management scheme enables the network to learn from actual users' behaviors and network status from the scratch, setting the network free from parameter configurations.
- The uplink signal-to-interference-plus-noise ratio (SINR) is leveraged to characterize the STA's state. DCRQN extracts the spatial and temporal features of wireless signals by means of CNN and RNN, which can well express the location and movement information of STA. With generality and adaptability, DCRQN can effectively improve the data rate during handoff process, outperforming the traditional scheme. In addition, the proposed scheme is software-based so that our work presented can be extended to various network scenarios.

C. PAPER ORGANIZATION

The remainder of the paper is organized as follows. Section II gives the related work. Section III provides the system model for dense WLANs, and section IV provides the technical description of the proposed scheme. Section V presents the experimental investigation and section VI gives the cost effectiveness. Concluding remarks are introduced in section VII.

II. RELATED WORK

So far, much literature has focused on the research of the handoff management in WLAN, including the handoff decision algorithm and mechanism. As for the former, a cross-layer approach has been presented in [2], enabling transport layer to perceive the wireless link so as to avoid performance degradation. The probe-wait time which makes up the probe phase is discussed in [6] to help reduce handoff delay. Shin *et al.* [32] adopt neighbor graphs and non-overlap graphs to reduce entire probe time spent dwelling on channels. Also, a feasible fast handoff scheme is proposed in [33] to reduce the handoff delay, which adopts a novel zero-channel-dwell-time architecture when probing each channel without dwelling to wait for probe response. The manners in [2], [6], [32], and [33] can lower the delay involved with handoffs by STAs but cannot work around it. Besides, they are all decentralized which are not conducive to the concentrated management and to make full use of network resources. As for handoff mechanism, several SDN state of the art solutions improve user mobility. The work presented in [9] abstracts the connections between the STAs and APs by means of software APs, which are migrated between physical APs to realize seamless handoff. However, its protocol restricts APs from working on the same channel, considering no carrier-sense multiple access with collision avoidance mechanism (CSMA/CA) [34], which constitutes a severe limitation for its use in real deployments. In [12] and [35], seamless handoffs between APs in different channels are realized, maintaining the Quality of Service of real-time services. The potential scalability issues associated to the beacon generation and channel assignment have been addressed. However, both solutions are suffering from similar drawbacks usage of separated control channels for management. Reference [36] bridges virtual and physical APs in order to achieve clean and transparent architecture of SDN network, resulting in the incorporation of control channels for wired portion of the network to one control channel. But the handoff decision algorithm still leaves much to be optimized. By combining SDN, Zhang *et al.* [8] propose a pre-scan based fast handoff algorithm for the SDN-based WLAN system, where the STA performs the authentication and re-association process towards the alternative AP directly. Zhang *et al.* [37] exploit the scan based on information of neighboring APs, RSS, and variable bitrate video coding to reduce handoff delay. The schemes in [8] and [37] have a centralized view of the network but they also introduce handoff delay and bring in hidden costs. Therefore, in this paper, we propose DCRQN to optimize the handoff decision algorithm using SDN to address aforementioned shortcomings.

III. SYSTEM MODEL

Analysis in the paper considers a SDN-based WLAN that consists of M APs and N STAs, in which mobile STAs access to network with specific logic. The STAs could generate uplink traffic continuously and each is only permitted to connect to one AP simultaneously. The downlink traffic transmitted by APs can be neglected, i.e., the paper focuses on the handoff scheme under the condition of saturated uplink traffic. APs are in different channels to reduce the scalability issues caused by the beacon generation for each STA. The CSA (Channel Switch Announcement) element in the AP beacon of 802.11 standard is carried out (as done in [38]) to move the STAs associated to an AP with a specific channel, without any change on the client side. Refer to the SDN framework in [35], an extra interface can be established for each AP to monitor traffic in other channels, without interrupting the normal operation of APs. According to the working channel of STA's serving AP, the SDN controller makes all APs switch their auxiliary interfaces to the channel and continuously listen to packets originated by the STA. In this case, the proposed scheme runs on SDN controller in the form of application, depending on the real-time network status (e.g., STA's state) perceived through southbound interface at time of decisions.

In this paper, the Log-distance Path Loss Model is leveraged to serve as the channel transmission model. The uplink RSS detected on AP$_j$ at the distance d_{ij} from STA$_i$ can be assigned by:

$$P_{d_{ij}} = P_{tx} + G_t + G_r - 20\log_{10}(4\pi f d_{ref}\sqrt{L}/c) \\ -10\delta\log_{10}(d_{ij}/d_{ref}). \quad (1)$$

where P_{tx} represents the transmitting power of STAs, G_t and G_r are the transmitting and receiving antenna gain respectively. f denotes the frequency band, d_{ref} is the reference distance to calculate the loss. L is the system loss, c corresponds to the light velocity and δ represents the path loss factor. In this case, the uplink SINR from STA$_i$ to AP$_j$ is:

$$SINR_{ij} = \frac{P_{d_{ij}}}{I_r + \sigma^2}. \quad (2)$$

where $I_r = \sum_{k \in N, k \neq i} P_{d_{ij}}(1 \leq k \leq N, 1 \leq j \leq M)$ indicates the cumulative interference caused by other STAs, and σ^2 is additive white Gaussian noise. In wireless networks, SINR gives theoretical upper bounds on channel capacity or the rate of information transfer in wireless networks, which can measure the quality of wireless connections. Therefore, we take the uplink SINR on each AP to jointly characterize the STA's state for a handoff.

IV. THE PROPOSED SCHEME

In this paper, we propose DCRQN to cope with the problems aforementioned. The section that follow presents the description of proposed scheme, including the basic theory of RL, the self-learning architecture along with the implementation of DCRQN in detail.

A. RL THEORY

RL falls into machine learning. The problem it mainly deals with is how an autonomous individual (i.e., agent) can learn the optimal policy that maximizes a specific metric by adapting its behavior within the environment. Fig.1 shows the basic form of RL. MDP defines the general form of the problems involved in RL [39]. In MDP, the agent is enabled to perceive the diverse state space S in the environment, with an executable action space A. At each time slot t, the agent senses the environment by identifying the current state $s_t \in S$ and then selects the action $a_t \in A$ to execute. The environment subsequently feeds back a return $r_t = r(s_t, a_t)$ while transitioning into a successor state $s_{t+1} = \varphi(s_t, a_t)$ with the probability $P(s_t, a_t, s_{t+1})$, in response to the action a_t. The goal is to learn a policy $\pi\colon S \rightarrow A$ that maximizes the accumulated rewards. Given a policy π, the expectation of accumulated rewards from state s_t can be defined as:

$$V^{\pi}(s) = E_{\pi}[\sum_{i=0}^{\infty} \gamma^i r_{t+i} | s_t = s]. \quad (3)$$

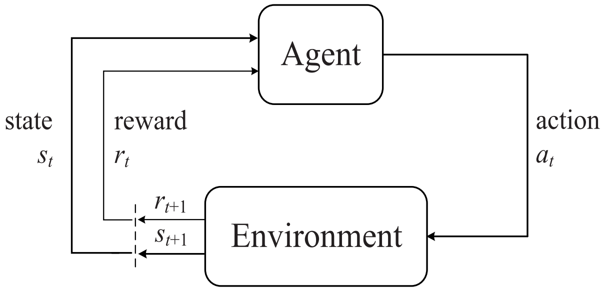


FIGURE 1. The basic component and form of RL.

where r_{t+i} indicates the reward by following π to take actions from the state s_t. In stochastic environment, the sequence of reward varies for same actions, and the more agent sees into the future from time t, the more it may diverge. Therefore, $\gamma \in [0, 1]$ is leveraged to discount future reward. The optimal policy π^* learned by the agent is:

$$\pi^* \equiv \arg\max_{\pi} V^{\pi}(s), \forall s \in S. \quad (4)$$

The function $V^{\pi}(s)$ of π^* can be denoted as $V^*(s) = \max_{\pi} V^{\pi}(s), \forall s \in S$. However, the optimal policy through learning V^* is definitely premised on the fact that the agent grasps immediate reward function r and the state transition function φ, i.e., it is demanded to predict the immediate reward and subsequent state of any state transition with precision. In many practical problems, however, the agent is not well aware of r and φ in a stochastic environment (e.g., handoff decision-making in WLAN). Therefore, learning V^*, in default of r and φ, is not conducive to making the optimal decisions.

The state-action value function, i.e., Q-function, works by improving evaluations of the quality of an action at particular states iteratively, leaving its learning free from function r and φ. The Q-value of a state-action pair (s, a) by policy π, denoted as $Q^{\pi}(s, a)$, represents the expected discounted reward given an initial action a at state s and following the policy π thereafter, which can be assigned by:

$$Q^{\pi}(s, a) = E_{\pi}[\sum_{i=0}^{\infty} \gamma^{i} r_{t+i} | s_t = s, a_t = a]. \tag{5}$$

According to the Markov transition model of the state, the equation above can be expanded:

$$\begin{aligned} &Q^{\pi}(s, a) \\ &= E_{\pi}[r_t | s_t = s, a_t = a] + E_{\pi}[\sum_{i=1}^{\infty} \gamma^{i} r_{t+i} | s_t = s, a_t = a] \\ &= R(s, a) + \gamma E[\sum_{i=0}^{\infty} \gamma^{i} r_{t+1+i} | s_t = s, a_t = a] \\ &= R(s, a) + \gamma \sum_{s' \in S} P(s, a, s') Q(s', a'). \end{aligned} \tag{6}$$

It is the basic form of the Bellman Equation, where $R(s, a)$ is the expectation of reward $r(s, a)$. The Q-function of the optimal policy is $Q^*(s, a) = \max_{\pi} Q^{\pi}(s, a)$, according to equation (6):

$$Q^*(s, a) = R(s, a) + \gamma \sum_{s' \in S} P(s, a, s') \max_{a' \in A} Q(s', a'). \tag{7}$$

Due to $V^*(s) = \max_{\pi} V^{\pi}(s)$, $\forall s \in S$, the $V^*(s)$ in Q-function can be expressed as:

$$\begin{aligned} V^*(s) &= \max_{a \in A} Q^*(s, a) \\ &= \max_{a \in A} [R(s, a) + \gamma \sum_{s' \in S} P(s, a, s') Q(s', a')]. \end{aligned} \tag{8}$$

Once $Q^*(s, a)$ is derived, for any state $s \in S$, the optimal action is the one with the largest $Q^*(s, a)$ value. That is,

$$\pi^* \equiv \arg\max_{a \in A} Q^*(s, a), \forall s \in S. \tag{9}$$

$Q^*(s, a)$ learning amounts to the optimal policy learning. The recursive definition of the Q-function provides the foundation for the iterative approximation of $Q^*(s, a)$. In Q-learning, each state-action pair (s, a) in the Q-table has an entry to store its Q-value. Among the common Q^* approximation algorithms, the Q-table is generally initialized at random. The process of sensing state s_t, selecting an action a_t, and obtaining a reward $r_t = r(s_t, a_t)$ along with a next state $s_{t+1} = \varphi(s_t, a_t)$ repeats. Then the updating rule of Q-values with learning rate $\alpha (0 \leq \alpha \leq 1)$ is given as follows:

$$\begin{aligned} Q(s_t, a_t) \leftarrow & Q(s_t, a_t) \\ & + \alpha(r(s_t, a_t) + \gamma \max_{a' \in A} Q(s_{t+1}, a') - Q(s_t, a_t)). \end{aligned} \tag{10}$$

Extensive literature [17], [40] has proved that the formula above can definitely converge to the Q-function of optimal policy. The idea behind Q-learning is to successively approximate the Q-function in accordance with Bellman Equation which says that the expected long-term rewards for a given action is equal to the immediate reward from the current action combined with expected reward from the best future action taken at the following state [22]. In the context of handoff decision-making in WLAN, the problem can be converted to an MDP, then Q-learning can be incorporated in this case.

Q-learning operates well on the premise that the state space is small, using a look-up Q-table to update Q-values. However, Q-learning cannot achieve desired results in the case of the infinite numbers of state space. To be specific, it is impractical to store the Q-values of all state-action pairs in the form of Q-table. Even if such a massive storage capacity is provided, the prohibitive computation complexity problems of querying for state-action pairs matter as well. Besides, the sample is too sparse for the sampling based algorithm to converge. As a result, the effectiveness of Q-learning will be degraded seriously.

Referring to human learning methods, it is impossible to undergo all circumstances in life as well. We are more likely to compare the new situations encountered with the experiences stored in memory. The similar actions are supposed to be taken for the similar cases, providing the reference to processing policies for the agent in RL. By creating functions to predict the Q-values, the corresponding decision could be obtained for any input state. In this way, the Q-table updating problem is transformed into a function fitting problem. Deep RL, in particular DQN, does apply this idea, taking forward neural networks as the Q-network. In such cases, Q-learning are approximated by neural network that takes the state s as input and the Q-value $Q(s, a)$ for all probable actions as output [26]. Theoretically, it turns out that neural network can approximate any function as long as its depth and width are sufficient [41].

B. DQN-BASED SELF-LEARNING ARCHITECTURE

The DQN-based architecture, which can be mainly divided into the network environment and the decision brain, is designed in Fig. 2. Its core is to provide a universal data access and instruction execution interface to decision brain. The decision brain is in charge of obtaining and converting the data to instructions, and operating the environment platform thereafter. The decision brain consists of three parts: the agent module, the feature extraction module and the decision module.

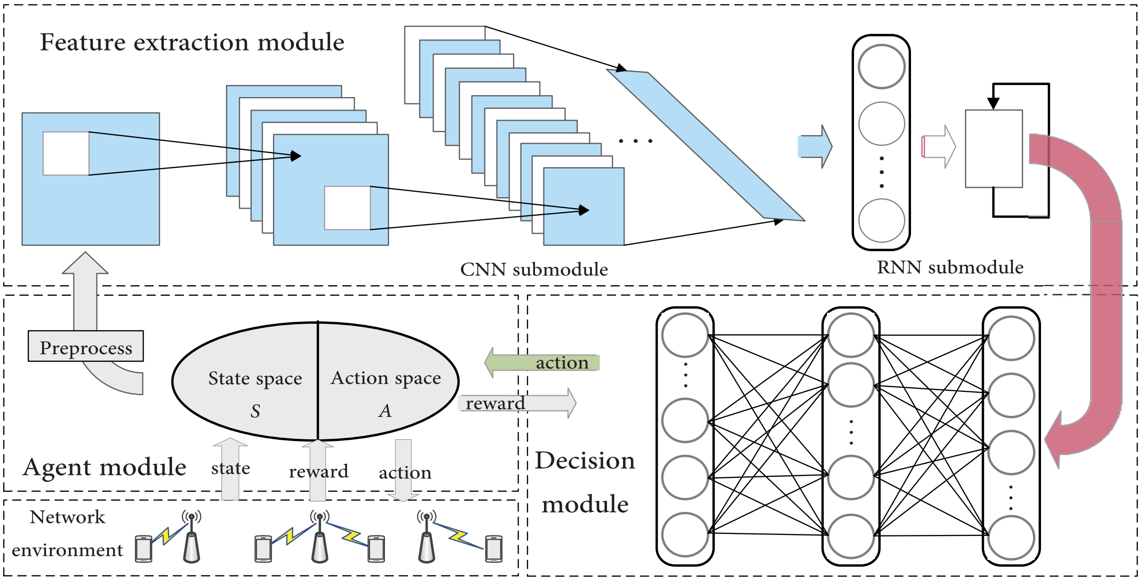


FIGURE 2. The DQN architecture applicable to the SDN-based WLAN.

To be specific, the agent module is an entity that perceives current state and takes actions in WLAN. Its principal functions are as follows:

- Perceive the network state (i.e., STA's state) through southbound interface;
- Preprocess the state information, then feed it to the feature extraction module;
- Take actions output by the decision module and get reward from environment.

To achieve the fine-grained attributes of wireless signals, the preprocessed state is fed to CNN submodule, followed by an RNN submodule. In general, the network density and layout of APs are inconsistent in diverse WLANs, which may have a direct impact on the handoff decision-making. The CNN is adopted to extract relative locations properties of STA in WLAN. In the case of the same other run conditions, to exemplify this, the decision module tends to handoff to the nearest AP for the STA. Note that "the nearest AP" is the true feature we need. The output of CNN can characterize the spatial location of the STA at a certain time, which will be fed to RNN submodule. In WLAN, the motion of STA leads to obvious fluctuation in received signals during propagation. In addition, there are significant correlations between consecutive wireless signals for some time. Inspired by the intrinsic properties above, we turn to RNN to extract temporary features of the affected wireless signals, thereby inferring the mobile velocity or moving direction of the STA. The temporary features integrate the spatial features, serving as the input to the decision module. The major functions are summarized:

- Read the preprocessed state from the agent module and take it as the input;
- Sequentially extract the spatial and temporary features of wireless signals;
- Read the cost returned by the decision module and update neural networks with back-propagation algorithm.

The decision module which is essentially a deep neural network (DNN) is leveraged to realize the mapping of state to actions. In other words, the optimal policy corresponds to the optimal decision-making function. By virtue of its nonlinear attribute, the model can approximate any function as long as the network is wide and deep enough [24]. Similarly, its major functions are as follows:

- Read the output of the feature extraction module and take it as the input;
- Use DNN instead of Q-value function to output actions' values for input state;
- Make decisions, and then notify the agent module to execute it accordingly;
- Read reward from the agent module. Calculate and feed the cost to the feature extraction module to update neural networks.

In summary, only the agent module needs to interact with the real environment, whereas other modules are irrelevant to the application scenarios. The role of the agent module is to package data and execute instructions, so it is in essence a middleware between the environment and the decision-making network. This module can be easily applied to diverse scenarios and tasks by modifying the date type. Due to the fact that deep learning can easily accommodate heterogeneous inputs, the work presented can be extended to various scenarios with slight modifications to reflect the change in the input size. In addition, the architecture is adaptive to users. Although there is a gap in the absolute range scale of the data for diverse users, our proposed scheme does depend more on the relative changes of the data (e.g., upward or downward trend). Therefore, the DQN-based architecture has generality and adaptability in terms of application scenarios or users.

C. DCRQN-BASED HANDOFF DECISION-MAKING

The main idea of DCRQN is to model the SDN controller as the RL agent and the WLAN scenario as the environment. Three main aspects about the MDP model, the Q-network for DCRQN along with the Q-network update are described in the following.

1) MDP MODEL

MDP is a mathematical framework that are used for modeling decision-making problem in which outcomes are partly random and partly depends on the action of agent. Due to the temporal correlation property, the handoff management is modeled as an MDP which is the dominant analytical approach in RL. The state, action, and reward are defined in the part that follow. Environment can be defined as the region of physical or virtual space with characteristic state space S. As mentioned in section III, the tuple composed of uplink SINR on APs is to jointly characterize the STA's state for a handoff. At time n, the state $S_n \in S$ perceived by the agent module is defined as:

$$S_n = \{s_1, s_2, \ldots, s_m, \ldots, s_{M-1}, s_M\}. \tag{11}$$

where $s_m \in S_n (0 < m \leq M)$ is the state information (i.e., SINR) perceived on AP$_m$ and M is the number of APs.

The agent module termly refreshes the state at a time interval τ, and preprocesses state S_n to an image-like tensor,

$$\phi(S_n) = \{S_{n-l+1}, S_{n-l+2}, \ldots, S_{n-l+j}, \ldots, S_{n-1}, S_n\}^{\mathrm{T}}. \tag{12}$$

where l is the time length for perceiving, indicating the size of the reformulated tensor $\phi(S)$ for feature extraction. It depends on the actual networks.

Besides, an action space A for agent is defined to interact with the network. The control action is denoted by $a \in A$ and takes on one of the values $1, 2, \ldots, M$, determining the handoff to target AP. That is

$$A = \{a_1, a_2, \ldots, a_m, \ldots, a_{M-1}, a_M\}. \tag{13}$$

In such case, the handoff to AP$_m$ is taken as the action $a_m, a_m \in A$. Meanwhile, we adopt the model-free method meaning that there is no knowledge of transition probabilities $P(s_t, a_t, s_{t+1})$. In modeled MDP, the agent tries to maximize the accumulated rewards. With the goal of average data rate maximization, we take the real-time uplink throughput as the reward r. It is considered as a positive handoff when a relatively higher value of throughput compared to other actions is returned, and a negative handoff when a lower value is returned. Considering that the proposed scheme is based on reward-driven, the AP with the largest SINR is not necessarily the target AP at time of handoffs.

2) Q-NETWORK FOR DCRQN

The Q-network acts as the decision-making function mapping the input state $\phi(S)$ to the Q-values of actions. In the proposed scheme, the Q-network adopts CNN and RNN for the fine-grained feature extraction of wireless signals, and DNN for the handoff decision.

Two successive convolutional layers are constructed to perform convolutional operations on the tensor $\phi(S)$. The first layer consists of 16 kernels that each size is 5 × 5, with "ReLU" activation function. And 32 kernels that each size is 3 × 3, are set up for the second layer, with "ReLU" function as well. For the generalization of model, we resort to max pooling function for dimensionality reduction. Each pooling kernel is 2×2 and the size of output feature map is maintained with zero padding. In that case, the convolutional layer and pooling layer work alternately. The CNN is in essence a nonlinear function with self-learning parameters, which can be defined as:

$$C = f_C(\phi(S); \vartheta_C(k, p; \beta_C)). \tag{14}$$

where $f_C(\cdot)$ is the nonlinear mapping function of CNN, $\vartheta_C(\cdot)$ is the parameter set. k and p indicate the parameters in convolutional and pooling layers, respectively. β_C denotes the variable parameters such as weights and biases.

Through convolutional processing of tensor $\phi(S)$, a 3D feature map C is obtained. The RNN submodule contains two RNN cells that each size is 256, with "Tanh" activation function. To cater for the RNN structure, the feature map C is reformulated to a 2D feature map C'. Likewise, the mapping function of RNN is assigned by:

$$\chi = f_R(C'; \vartheta_R(u; \beta_R)). \tag{15}$$

where $f_R(\cdot)$ is the nonlinear function, $\vartheta_R(\cdot)$ is the parameter set, and u indicates the layer size of RNN cells. The variable parameters are denoted by β_R. The state vector χ is a summary of tensor $\phi(S)$ of the whole feature extraction module. It can not only reflect the spatial properties of STAs in WLAN, but also reflect the temporary properties about the mobility information.

The decision module by itself belongs to DNN, containing two fully connected layers, which is backward connected to a softmax classifier to compute Q-values of M available actions. L2-norm regularization with weight decay of 0.001 is leveraged to avoid over-fitting [42]. Given an input χ, the Q-values of actions can be calculated by:

$$Q(\chi, a_i; \vartheta_D(v; \beta_D)) = f_D(\chi, a_i; \vartheta_D(v; \beta_D)), \quad a_i \in A. \tag{16}$$

where $f_D(\cdot)$ is the nonlinear function approximator in the decision process, $\vartheta_D(\cdot)$ is the set of parameters and v is related to the model scale. β_D, similarly, indicates the variable parameters in DNN. The decision-making action can be identified following the policy:

$$\pi(a) = \arg\max_{a_i \in A} Q(\chi, a_i; \vartheta_D(v; \beta_D)). \tag{17}$$

TABLE 1. Parameters of the *Q*-network for DCRQN.

Name	Symbol	Number	Size	Activation
Convolutional layer	k	2	$5\times5(16), 3\times3(32)$	ReLU
Pooling layer	p	2	$2 \times 2, 2 \times 2$	NA
RNN cell	u	2	256, 256	Tanh
DNN hidden layer	v	2	256, 128	ReLU
Softmax classifier	NA	1	M	Sigmoid

The decision module will instantly inform the agent module of the decisions for the handoff management. The Q-network for DCRQN is illustrated in Table 1.

3) Q-NETWORK UPDATE

Three types of neural networks are involved in the decision-making process. For the sake of simplicity, they are integrated into an integral Q-network. Based on this observation, the mapping from tensor $\phi(S_t)$ to Q-values can be expressed:

$$Q(\phi(S_t), a_i; \theta) = F_N(\phi(S_t), a_i; \theta). \tag{18}$$

where $F_N(\cdot)$ is the joint nonlinear function of CNN, RNN along with DNN. The parameter θ is the variable parameter set including β_C, β_R and β_D. Likewise, $Q(\phi(S_t), a_i; \theta)$ is the Q-value of action a_i given the input $\phi(S)$ at time t, i.e., it denotes the preference degree of action a_i. Correspondingly, the decision-making action can be identified as:

$$\pi(a) = \arg\max_{a_i \in A} Q(\phi(S_t), a_i; \theta). \tag{19}$$

For the optimal mapping policy of state to action, $F_N(\cdot)$ is supposed to optimize Q-network parameters through training. However, traditional RL is considered unstable or even diverged when using a nonlinear function approximator such as neural network to represent the Q-value function [43]. To cope with it, we turn to the following two methods to make the learning more stable and efficient.

During each mapping, Q-network generates a piece of memory consisting of current state s_t, current action a_t, instant reward r, the next state s_{t+1} and stores it into the replay memory D. The form of D is:

$$D = \underbrace{\{\ldots, (\phi(S_t), a_t, r_t, \phi(S_{t+1})), \ldots\}}_{\omega}^{\mathrm{T}}. \tag{20}$$

where ω is the capacity and t indicates the historical memory at time t. During the training process, each generated memory tuple is stacked into D. In this case, Q-network can upset the correlations between memories, thereby improving the learning efficiency.

There are two Q-networks in DQN actually, Q-evaluation network and Q-target network. The evaluation network is used for learning and updating parameters including weights and biases in real time. The "temporarily frozen" target network $\hat{Q}(\phi(S), a; \hat{\theta})$ is established to decouple the action decision process. At each time step, the target network together with memories randomly sampled from D calculates the cost and trains the evaluation network.

For each sampled tuple like $(\phi(S_j), a_j, r_j, \phi(S_{j+1}))$, the cost function $Cost(\theta)$ can be defined:

$$Cost(\theta) = E_d[(y_j - Q(\phi(S_j), a_j; \theta))^2]. \tag{21}$$

where y_j is the target Q-value and calculated by:

$$y_j = r_j + \gamma \max_{a' \in A} \hat{Q}(\phi(S_{j+1}), a'; \hat{\theta}). \tag{22}$$

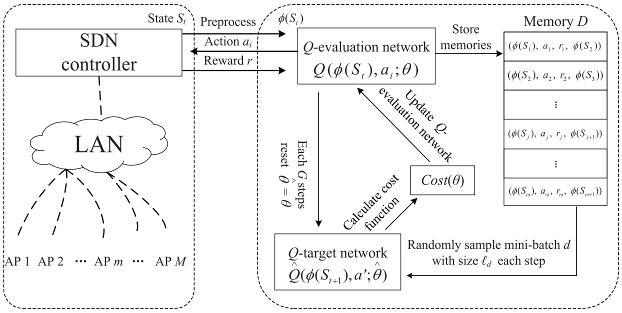

State S_t
Preprocess
$\phi(S_t)$
SDN controller
Action a_i
Reward r
Q-evaluation network
$Q(\phi(S_t), a_i; \theta)$
Store memories
Memory D
$(\phi(S_1), a_1, r_1, \phi(S_2))$
$(\phi(S_2), a_2, r_2, \phi(S_3))$
⋮
$(\phi(S_j), a_j, r_j, \phi(S_{j+1}))$
⋮
$(\phi(S_\omega), a_\omega, r_\omega, \phi(S_{\omega+1}))$
LAN
Each G steps reset $\hat{\theta} = \theta$
Update Q-evaluation network
$Cost(\theta)$
Calculate cost function
Q-target network
$\hat{Q}(\phi(S_{t+1}), a'; \hat{\theta})$
Randomly sample mini-batch d with size ℓ_d each step
AP 1
AP 2
···
AP m
···
AP M

FIGURE 3. *Q*-network training process for DCRQN.

Fig. 3 depicts the overall training process of Q-network. The network state S_t is perceived and then preprocessed as tensor $\phi(S_t)$. The evaluation network then generates a piece of memory and stores it into D. Only when the memory storage is large enough, the target network samples a mini-batch d with size of ℓ_d and calculates the cost together with the evaluation network. According to the cost, gradient descent algorithm is leveraged to train the evaluation network each step to minimize the cost. For each G iterations, the target network is copied from the evaluation network. Therefore, it is said to be "temporarily frozen".

Besides, we leverage the ϵ-greedy policy to improve the learning ability in breaking away from the local optimum. DQN is an on-line learning scheme that allows the simultaneous performance of two tasks: exploration and exploitation. In ϵ-greedy, with probability $\epsilon \in [0, 1]$, the agent will select an action $a \in A$ at random (exploration), or with probability $1-\epsilon$ it will select according to the policy $\pi(a)$ (exploitation). As the training proceeds, the policy will gradually converge. Hence a linear function to decrease the exploration rate ϵ from initial value ϵ_i to final value ϵ_f is set up to stick to our decisions, represented as: $\epsilon = \epsilon - (\epsilon_i - \epsilon_f)/\zeta$, where ζ is the iteration cycle and the initial value of ϵ is ϵ_i.

To be specific, the DCRQN can be divided into the training phase (Algorithm 1) and the inference phase (Algorithm 2). In training phase, the parameters related to WLAN scenario and Q-networks are initialized first. The training iteration cycle ζ is determined, working as the condition derived from the training that trigger the inference phase. The agent module reads the state information S_t and reformulates it to $\phi(S_t)$. The evaluation network takes $\phi(S_t)$ as input. Q-values of actions are output through the feature extraction module and decision module. Following the ϵ-greedy, the decision module selects a random action $a_t \in A$ with probability ϵ, or selects

```
Algorithm 1 DCRQN Training Phase
Input: State information, $\phi(S_t)$.
Output: Handoff decision to the target AP, $a_t$.
Initialize the scenario parameters $M, A, l, \tau$
Initialize the statistic parameters $\zeta, \epsilon, \epsilon_i, \epsilon_f, \omega, \ell_d, \alpha, G$
Initialize the $Q$-evaluation network with random parameters $\theta$
Initialize the $Q$-target network with $\hat{\theta} = \theta$
Read the state $S_t$ and preprocess it to $\phi(S_t)$
for episode = 1 to $\zeta$ do
    Input $\phi(S_t)$ to the evaluation network and output the $Q$-values of actions
    With probability $\epsilon$ select a random action $a_t \in A$
    Otherwise select $a_t = \arg\max_{a \in A} Q(\phi(S_t), a; \theta)$
    Set $\epsilon = \epsilon - (\epsilon_i - \epsilon_f)/\zeta$
    Execute action $a_t$, obtain reward $r_t$ and next state $S_{t+1}$
    Reformulate state $S_{t+1}$ to $\phi(S_{t+1})$
    Store experience tuple $(\phi(S_t), a_t, r_t, \phi(S_{t+1}))$ into $D$
    Sample a mini-batch of $(\phi(S_j), a_j, r_j, \phi(S_{j+1}))$ from $D$
    Set $y_j = r_j + \gamma \max_{a' \in A} \hat{Q}(\phi(S_{j+1}), a'; \hat{\theta}))$
    Perform gradient descent on $(y_j - Q(\phi(S_j), a_j; \theta))^2$ with respect to $\theta$
    Reset $\phi(S_t) = \phi(S_{t+1})$
    Every $G$ steps, update the target network with $\hat{\theta} = \theta$
end for
```

```
Algorithm 2 DCRQN Inference Phase
Input: State information, $\phi(S_t)$.
Output: Optimal handoff decision to the target AP, $a_t$.
Read the model saved in the training phase
Read the state $S_t$ and preprocess it to $\phi(S_t)$
for episode = 1 to $\zeta$ do
    Input $\phi(S_t)$ to the evaluation network and output the $Q$-values of all actions
    Select $a_t = \arg\max_{a \in A} Q(\phi(S_t), a; \theta)$
    Execute action $a_t$, obtain reward $r_t$ and next state $S_{t+1}$
    Reformulate state $S_{t+1}$ to $\phi(S_{t+1})$
    Reset $\phi(S_t) = \phi(S_{t+1})$
end for
```

$a_t = \arg\max_{a \in A} Q(\phi(S_t), a; \theta)$. ϵ is reset by $\epsilon = \epsilon - (\epsilon_i - \epsilon_f)/\zeta$ each step. The agent module takes action a_t to operate the network. The reward r_t and next state S_{t+1} are gained thereafter, followed by the preprocessing from S_{t+1} to $\phi(S_{t+1})$. The evaluation network stacks the memory tuple $(\phi(S_t), a_t, r_t, \phi(S_{t+1}))$ into D. At each step, the target network randomly samples a mini-batch from D and performs a gradient descent step on the cost calculated together with the evaluation network. For each G steps, the target network is updated with $\hat{\theta} = \theta$. The optimal handoff policy can be obtained through massive training iterations (to be explained in section VI). In inference phase, however, there will be no need to introduce ϵ-greedy and target network for training. We just need to be firm in decision-making. And the performance of the policy gained will be verified next.

V. EXPERIMENTAL INVESTIGATION

The WLAN system model is established by means of Mininet-WiFi emulator [44]. Mininet-WiFi is an extension of the SDN simulator Mininet, with additional standard Linux wireless drivers and 80211_hwsim wireless simulation drivers. It is thus probable to set up APs and STAs and their communication process can be emulated thereafter. In addition, we could define the node's location, AP's working mode, STA's moving direction and velocity in the form of scripting. These attributes help to realize the fine-grain control on the underlying wireless network packets. The learning phase is implemented on TensorFlow, a Google's open source platform for deep learning.

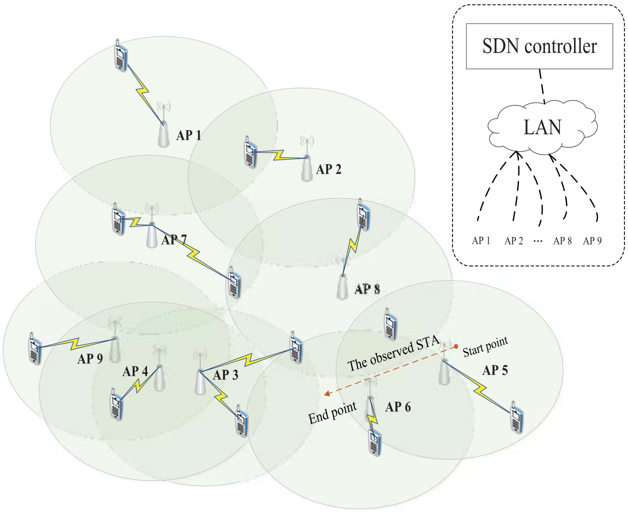


FIGURE 4. The WLAN scenario set up in Mininet-WiFi.

All the simulations used to verify the policy performance were done in the WLAN scenario as shown in Fig. 4. Analysis considers a dense WLAN consisting of nine APs and twelve STAs, where all APs are distributed at random and OpenFlow enabled by using Open vSwitch. Among the STAs, one serves as a mobile object observed by the agent module, while others are stationary to produce background traffic. Note that all APs are open authentication to STAs. The observed STA moves randomly throughout the radio coverage areas. Besides, uplink SINR can be gained according to the RSS and interference through the auxiliary interfaces on APs. *Iperf*, a survey instrument for network performance, is leveraged to measure the throughput between two ports by creating data streams. Moreover, *tcpdump* command is adopted to capture the time stamp of packets to depict the variations in throughput. The widely used utility *iw* is adopted, in response to the handoff instructions made by the decision module:

iw wlan connect or *disconnect* (SSID (Service Set Identifier)) (MAC). In that case, handoff process can be operated by disconnecting the old link and then connecting to the target AP. The SDN-based implementation ensures the seamless handoff.

The paper takes the following three representative handoff schemes as the control group, and carries out two experiments E_1 and E_2 to demonstrate the effectiveness of DCRQN.

- RSST based handoff scheme: It is the traditional WLAN handoff scheme which contains the *handoff initiation*, *discovery process* and *re-authentication*. In order to avoid the ping-pong effect, the scheme comes equipped with two RSS thresholds T_1 and T_2. T_1 is the triggering threshold to initialize the *discovery process*, i.e., the handoff will be triggered when the RSS drops to T_1; When the difference value of uplink RSSs between the largest one from neighboring APs and serving AP exceeds T_2, the largest one serves as the target AP.
- RSSS based handoff scheme: It is the SDN-based RSST scheme, with the same pre-configured thresholds T_1 and T_2. The STA needs no channel scanning (i.e., *discovery process*) when performing handoff, and the handoff can be considered seamless.
- DQNH based handoff scheme: Similar to DCRQN, DQN-based DQNH makes use of neural networks to map the state to handoff decisions. DCRQN turns to a feature extraction module to extract fine-grained features of wireless signals. In order to compare scheme effectiveness, DQNH involves no feature extraction modules. To control condition variables, the structures of the agent module and decision module of DQNH are the same as those of DCRQN.

TABLE 2. Network system model parameters.

Name	Symbol	Values
Frequency band	f	2.4 GHz
Working channels for APs	NA	combinations of 1, 6, and 11
Clear channel assessment threshold	NA	-62 dBm
Interference threshold	NA	-72 dBm
Transmitting power	P_{tx}	21 dBm
Path loss factor	δ	3.0
Receiving gain	G_r	5 dBm
Transmitting gain	G_t	5 dBm
Reference loss distance	d_{ref}	1.0 m
Background noise	σ^2	-92 dBm
System loss	L	1 dBm
Trigger threshold	T_1	-58 dBm
Handoff threshold	T_2	5 dBm

The network system model parameters are specified in Table 2, and the parameters regarding DCRQN are illustrated in Table 3. In particular, the input of neural network is $\phi(S)$ with size of 64 × 9, and the output size is 9 for handoff decision-making. Each value in $\phi(S)$ is a tuple composed of nine SINRs. The agent module refreshes $\phi(S)$ every 0.5 s which will not bring lots of load, i.e., the time length of input tensor is 32 s. The whole iteration cycle is set as 200,000 steps and capacity of memory D is 20,000 units. The initial exploration rate ϵ_i and final rate ϵ_f are set as 0.4 and 0.01 to stick to our decisions as learning proceeds. During training phase, Q-target network is copied from Q-evaluation network every 20 steps. Meanwhile, the learning rate α is 0.01 to optimize the network parameters. During simulation, all APs comply with IEEE 802.11g protocol. All the measurements are averaged over thirty runs.

TABLE 3. Parameters regarding DCRQN.

Name	Symbol	Values
State space size	M	9
Input tensor $\phi(S)$ length	l	64
State refreshing frequency	τ	0.5 s
Iteration cycle	ζ	200,000
Replay memory capacity	ω	20,000
Initial exploration rate	ϵ_i	0.4
Final exploration rate	ϵ_f	0.01
Mini-batch size	ℓ_d	32
Discounted factor	γ	0.6
Q-target network update frequency	G	20
Learning rate	α	0.01

A. EXPERIMENT E_1

To have an intuitive comparison between the RSST scheme, RSSS scheme, DQNH scheme and the proposed scheme aforementioned, we compare the trend of throughput variation in the case of a single handoff. With AP 5 and AP 6 in Fig. 4 as a case, according to the moving track, the STA moves at a constant velocity of 0.4 m/s between the APs. In particular, within 0-30 s, the STA was in the coverage area of AP 5; within 30-200 s, the STA was in the overlapping areas; Eventually, the STA entered the coverage area of AP 6.

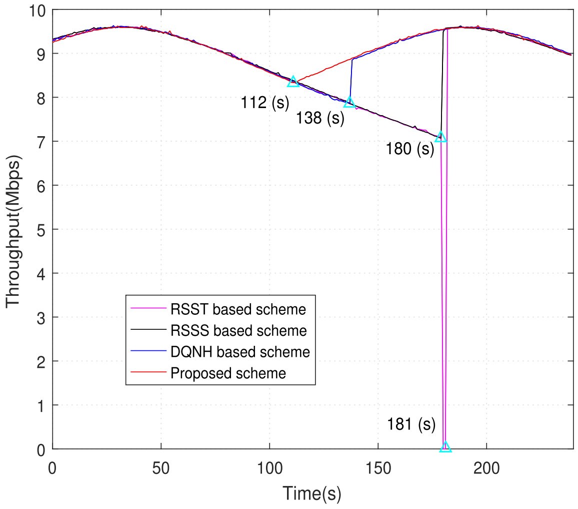


FIGURE 5. The variation of throughput during a single handoff process.

Fig. 5 shows the variations in throughput under each handoff scheme, respectively. Experimental results show that: 1) The RSST has an obvious handoff delay which is manifested by a throughput reduction to zero as shown at 181 s. In contrast, the RSSS, DQNH and our proposed scheme do not introduce any delay. 2) It can be seen that the throughput curves depicted are almost coincidence when STA was only in the coverage area of AP 5 or AP 6. With the movement of STA away from AP 5 to the overlapping area, the throughput value continues to decline as the channel condition deteriorates. When the value drops to a certain extent, at 112 s, the proposed scheme triggers the handoff at the earliest. The throughput afterwards gets a relatively gentle rebound. The DQNH makes the handoff decision somewhat delayed, at 138 s. The throughput does not pick up in the face of uncompleted handoff. While the RSSS makes the STA handoff significantly delayed, at 180 s, which tends to leave the STA to subject to low data rate for some time. The handoff timing performed by RSST leaves much to be desired, at 181 s, which results in the curve dropping to the nadir, followed by a sharp increase. The RSST is not only postponed but also introduces obvious handoff delay. By contrast, the throughput under the proposed scheme is more stable during handoff process, without massive jitters. It can be concluded that the lagging handoff decision prolongs the time when the throughput decreases continually, leading to a low data rate state for long during the handoff process.

From the above results, the RSST serving as the traditional WLAN handoff scheme does not trigger handoff until the RSS values satisfy the conditions regarding the thresholds, which generally leaves the handoff timing too late. Compared to RSST, the advantage of RSSS is being software-based, i.e., the handoff is considered seamless. It can be seen in the figure, however, when to trigger handoff and which AP to reconnect to are still problems to be addressed. The DQN-based DQNH could make relatively accurate handoff decisions to some extent through a learning phase. However, in default of the feature extractions, it cannot achieve the desired results. The proposed scheme first introduces SDN to implement seamless handoff in WLAN. Meanwhile, CNN and RNN are used to extract the spatial and temporal characteristics of wireless signals. DNN is then leveraged to make handoff decisions using the extracted results. In brief, the proposed scheme has a facility to make the optimal decisions with respect to the network state, which avoids the throughput dropping a lot.

B. EXPERIMENT E_2

Without loss of generality, the effectiveness of DCRQN is further analyzed in the entire WLAN. We take the Random Way Point Movement Model as the mobile model for STA. Similarly, a constant velocity of 0.4 m/s to traverse the network is provided. And again, the throughput is counted for 20,000 s under each handoff scheme.

Fig. 6 depicts the Cumulative Distribution Function (CDF) of the throughput, where each curve corresponds to each scheme. It can be observed from the figure that the probability of acquiring a higher throughput (8 to 10 Mbps) for the proposed scheme is around 53%, whereas the respective probabilities for DQNH, RSSS and RSST based scheme are 48%, 40% and 30%, respectively. The statistics above demonstrate the effectiveness of DCRQN. From another point of view, when the CDF value is around 0.5, the throughput of proposed scheme is 0.9 Mbps higher than that of RSST and 0.5 Mbps higher than that of RSSS. Similarly, it is also greater than that of DQNH by 0.3 Mbps. Note that STA moves along a consistent path each time (controlled by random seed). Therefore, the fundamental reason for these diversities lies in the diverse handoff schemes adopted by the network. For RSST based scheme, the specified parameters like T_1 and T_2 are challenging to apply to the global network due to the irregular distribution of APs. Besides, it may introduce noticeable delays during the handoff process, which can be reflected from the curve with zero value. The RSSS based scheme could ensure the continuity of service. However, the timing to trigger handoff and the selection of AP remain to be optimized, which emphasizes the necessity of studying new handoff algorithms. For DQNH based scheme, a handoff policy applicable to the WLAN can be gained through a learning phase, without configuring parameters. Its ability to

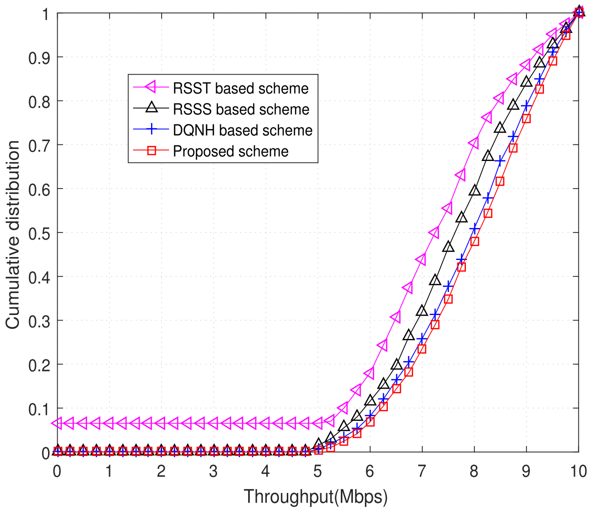


FIGURE 6. The CDF of throughput by traversing the WLAN.

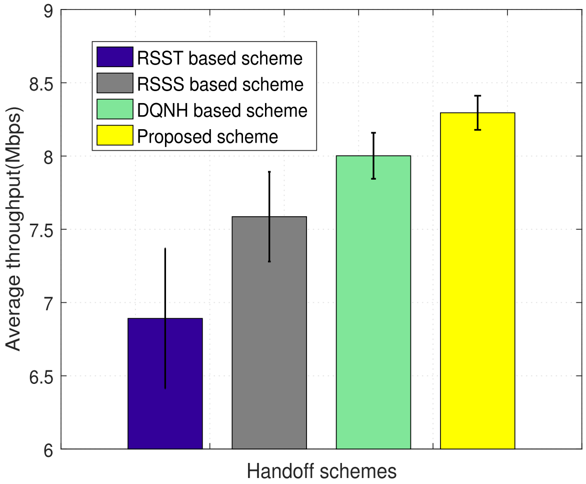
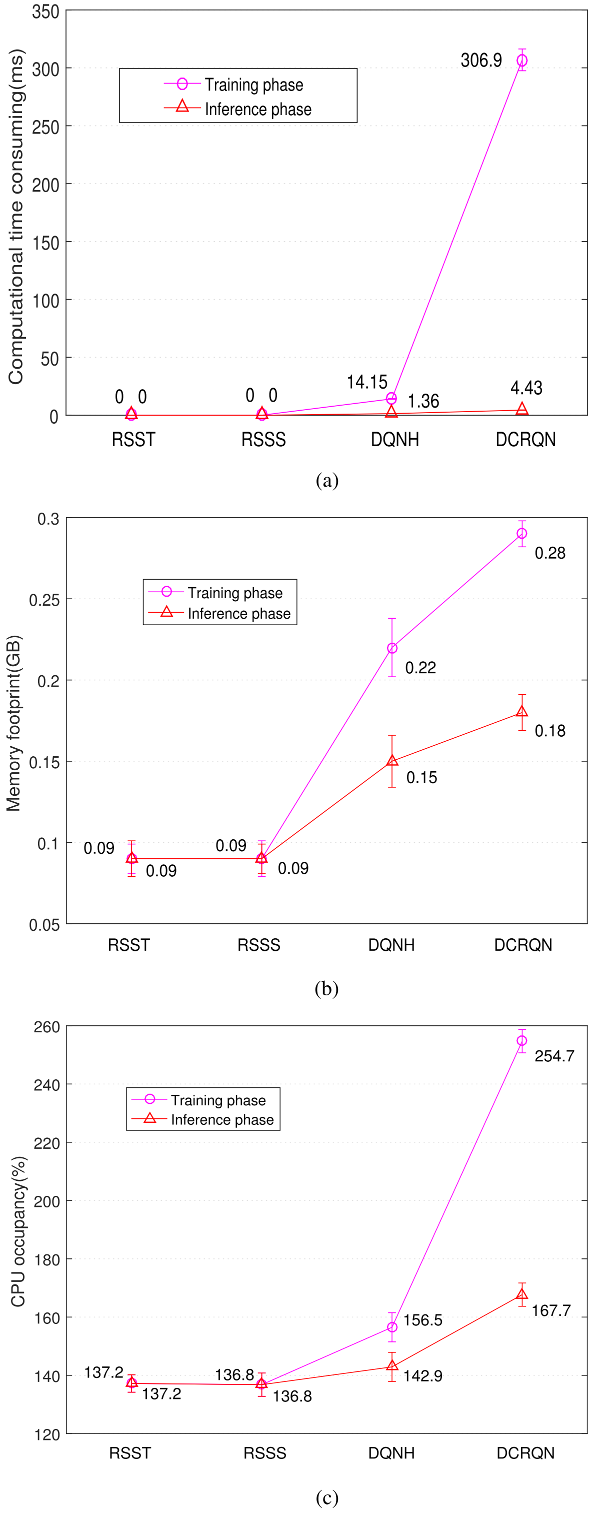


FIGURE 7. The throughput averaging E_2 over thirty runs.



(a)

(b)

(c)

FIGURE 8. **The complexity comparison in training and inference phase. (a) Time cost. (b) Memory footprint. (c) CPU occupancy.**

extract features, however, remains to be improved, and the necessity on sufficiency for feature extraction is underlined here.

For more impressive reflection of gain in throughput, analysis considers the throughput by averaging E_2 over thirty runs, which is shown in Fig. 7. It can be seen that proposed scheme has obvious superiorities in throughput gains, 21% improvements to RSST, 9.5% and 3.8% to RSSS and DQNH, respectively. In addition, the stability of DCRQN is better than other schemes, which is reflected in smaller standard deviation. In brief, DCRQN considers both the spatial and temporal features of wireless signals by means of CNN and RNN. Together with the goal of average throughput maximization, the optimal handoff policy can be gained.

VI. COST EFFECTIVENESS

Due to the random distributions of APs, STA needs to traverse most of the WLAN for optimal handoff policy. In the DCRQN training phase, the neural networks are trained for around 36 hours, with CPU. Considering that RL belongs to on-line learning, the overall time cost can be divided into the time for STA traversing and time for neural network computing.

The gain of throughput achieved by DCRQN comes at the cost of complexity. Therefore, a detailed analysis of the complexity is given in Fig. 8 to prove the feasibility of DCRQN. Fig. 8(a) depicts the time cost in training and inference phase under each handoff scheme. RSST or RSSS needs no learning, and its cost is so small that it can be ignored. For DQNH, the cost in training phase is about 14.15 ms and in inference phase is 1.36 ms. This is because there are additional parameters iterative update steps in training phase than in inference phase. For proposed scheme DCRQN, the cost in training phase is 306.9 ms which is much higher than that of others. Actually, the extra cost is spent optimizing the feature extraction module. Next to the inference phase, the cost is reduced to 4.43 ms, which is within acceptable range. Moreover, tensor calculation and neural network update can be accelerated greatly for more than 8 times by using GPU. Fig. 8(b) and 8(c) depicts the memory footprints and CPU occupancy in the two phases, respectively. The test environment is Ubuntu 16.04 system with Quad Core CPU (the CPU full occupancy is 400%) and 7.5GB memory. It can be seen that in training phase, the complexity of DCRQN is much higher than that of others, but in inference phase, the value is again reduced to a tolerable range. Note that we are more concerned about the performance in practical applications, i.e., the inference phase. It is worthwhile to sacrifice a small amount of complexity in exchange for a significant gain in throughput. Therefore, DCRQN trades off the complexity and gain, with practical feasibility.

VII. CONCLUSION

In this paper, we focus on the handoff management scheme in the SDN-based WLAN. In order to cope with the defects of the traditional WLAN handoff scheme, we first design a self-learning architecture which is applicable to SDN-based WLAN frameworks. Moreover, we propose DCRQN, a novel DQN-based handoff decision algorithm where SINR is leveraged to characterize the STA's state. The proposed scheme enables the network to learn from the real users' behaviors and the network status from the scratch, setting the network free from parameter configurations. Besides, CNN and RNN are resorted to extract the fine-grained features of the wireless signals, respectively. The experimental results show that the proposed scheme can significantly improve the data rate during the handoff process, outperforming the traditional handoff scheme. However, the shortcoming of the paper is that the WLAN scenario is built with simulator, which may deviate from the actual situations. And the work only considers one observed agent. In our future research, actual scenarios together with DRL with multiple STAs (i.e., multiple agents) will be taken into consideration.

REFERENCES

[1] *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Standard 802.11, 1999.

[2] K. Tsukamoto, R. Ijima, S. Kashihara, Y. Oie, "Impact of layer 2 behavior on TCP performance in WLAN," in *Proc. IEEE 62nd Veh. Technol. Conf. (VTC)*, Sep. 2005, pp. 1713–1717.

[3] W. Qin, Y. Teng, M. Song, Y. Zhang, and X. Wang, "AQ-learning approach for mobility robustness optimization in LTE-son," in *Proc. 15th IEEE Int. Conf. Commun. Technol.*, Nov. 2013, pp. 818–822.

[4] H. Zhang, X. Chu, W. Guo, and S. Wang, "Coexistence of Wi-Fi and heterogeneous small cell networks sharing unlicensed spectrum," *IEEE Commun. Mag.*, vol. 53, no. 3, pp. 158–164, Mar. 2015.

[5] D. Lee, D. Won, M. D. J. Piran, and D. Y. Suh, "Reducing handover delays for seamless multimedia service in IEEE 802.11 networks," *Electron. Lett.*, vol. 50, no. 15, pp. 1100–1102, Jul. 2014.

[6] A. Mishra, M. Shin, and W. Arbaugh, "An empirical analysis of the IEEE 802.11 MAC layer handoff process," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 2, pp. 93–102, 2003.

[7] T. Lei, X. Wen, Z. Lu, W. Jing, B. Zhang, and G. Cao, "Handoff management scheme based on frame loss rate and RSSI prediction for IEEE 802.11 networks," in *Proc. Int. Symp. Wireless Commun. Syst. (ISWCS)*, Sep. 2016, pp. 555–559.

[8] B. Zhang, X. Wen, Z. Lu, T. Lei, and X. Zhao, "A fast handoff scheme for IEEE 802.11 networks using software defined networking," in *Proc. 19th Int. Symp. Wireless Pers. Multimedia Commun. (WPMC)*, Nov. 2016, pp. 476–481.

[9] T. Lei, Z. Lu, X. Wen, X. Zhao, and L. Wang, "SWAN: An SDN based campus WLAN framework," in *Proc. 4th Int. Conf. Wireless Commun., Veh. Technol. Inf. Theory Aerosp. Electron. Syst. (VITAE)*, May 2014, pp. 1–5.

[10] *TS-020 OpenFlow Switch Specification Version 1.5.0 (Protocol version 0×06)*, Open Netw. Found., Menlo Park, CA, USA, Dec. 2014.

[11] J. Bukhari and W. Yoon, "Design of scalable SDN based eMBMS/WLAN network architecture assisted by fog computing," in *Proc. Int. Conf. Inf. Commun. Technol. Converg. (ICTC)*, Oct. 2018, pp. 216–218.

[12] *Software-Defined Networking: The New Norm for Networks*, Open Netw. Found., Menlo Park, CA, USA, 2012.

[13] L. Suresh, J. Schulz-Zander, R. Merz, A. Feldmann, and T. Vazao, "Towards programmable enterprise WLANs with odin," in *Proc. 1st workshop Hot Topics Softw. Defined Netw. (HotSDN)*, Aug. 2012, pp. 115–120.

[14] K. Yang, K. Yang, I. Gondal, B. Qiu, and L. S. Dooley, "Combined SINR based vertical handoff algorithm for next generation heterogeneous wireless networks," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Nov. 2007, pp. 4483–4487.

[15] S. Lal and D. K. Panwar, "Coverage analysis of handoff algorithm with adaptive hysteresis margin," in *Proc. 10th Int. Conf. Inf. Technol. (ICIT)*, Dec. 2007, pp. 133–138.

[16] K. Tsukamoto, T. Yamaguchi, S. Kashihara, and Y. Oie, "Experimental evaluation of decision criteria for WLAN handover: Signal strength and frame retransmission," in *Proc. 3rd Int. Conf Ubiquitous Comput. Syst. (UCS)*. Berlin, Germany: Springer, 2006, pp. 239–253.

[17] H. Tabrizi, G. Farhadi, and J. Cioffi, "Dynamic handoff decision in heterogeneous wireless systems: Q-learning approach," in *Proc. Int. Conf. Commun. (ICC)*, Jun. 2012, pp. 3217–3222.

[18] J. Yang and P. Zong, "Robust and fast inter-EBS handoff mechanism," U.S. Patent 8 190 158, May 29, 2012.

[19] L. Cai, Y. Xiao, and X.(Sherman).Shen, "VoIP over WLAN: Voice capacity, admission control, QoS, and MAC," *Int. J. Commun. Syst.*, vol. 19, no. 4, pp. 491–508, 2006.

[20] M. Abusubaih and A. Wolisz, "Interference-aware decentralized access point selection policy for multi-rate IEEE 802.11 wireless LANs," in *Proc. IEEE 19th Int. Symp. Pers. Indoor Mobile Radio Commun.*, Sep. 2008, pp. 1–6.

[21] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998.

[22] F. L. Lewis and D. Vrabie, "Reinforcement learning and adaptive dynamic programming for feedback control," *IEEE Circuits Syst. Mag.*, vol. 9, no. 3, pp. 32–50, 3rd Quart., 2009.

[23] A. G.-Serrano and L. Giupponi, "Distributed Q-learning for aggregated interference control in cognitive radio networks," *IEEE Trans. Veh. Technol.*, vol. 59, no. 4, pp. 1823–1834, May 2010.

[24] Y.-H. Chen, C.-J. Chang, and C. Y. Huang, "Fuzzy Q-learning admission control for WCDMA/WLAN heterogeneous networks with multimedia traffic," *IEEE Trans. Mobile Comput.*, vol. 8, no. 1, pp. 1469–1479, Nov. 2009.

[25] C. J. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, pp. 279–292, May 2014.

[26] Y. Bengio and O. Delalleau, "On the expressive power of deep architectures," in *Discovery Science*. Berlin, Germany: Springer, 1992, p. 1.

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

[28] V. Mnih *et al.* (Dec. 2013). "Playing atari with deep reinforcement learning." [Online]. Available: http://arxiv.org/abs/1312.5602

[29] J. Li, W. Monroe, A. Ritter, D. Jurafsky, M. Galley, and J. Gao, "Deep reinforcement learning for dialogue generation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Nov. 2016, pp. 1192–1202.

[30] O. Naparstek and K. Cohen, "Deep multi-user reinforcement learning for dynamic spectrum access in multichannel wireless networks," in *Proc. GLOBECOM IEEE Global Commun. Conf.*, Dec. 2017, pp. 3217–3222.

[31] A. Selim, F. Paisana, J. A. Arokkiam, Y. Zhang, L. Doyle, and L. A. DaSilva, "Spectrum monitoring for radar bands using deep convolutional neural networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2017, pp. 1–6.

[32] M. Shin, A. Mishra, and W. A. Arbaugh, "Improving the latency of 802.11 hand-offs using neighbor graphs," in *Proc. ACM SIGCOMM Comput. Commun. Rev.*, Jun. 2004, pp. 70–83.

[33] X. Chen and D. Qiao, "HaND: Fast handoff with null Dwell time for IEEE 802.11 networks," in *Proc. IEEE INFOCOM*, Mar. 2010, pp. 1–9.

[34] *802.11-2012—IEEE Standard for Information Technology—Telecommunications and Information Exchange Between Systems Local and Metropolitan Area Networks—Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999), Mar. 2012.

[35] L. Sequeira, J. L. de la Cruz, J. R. Mas, J. Saldana, J. F. Navajas, and J. Almodovar "Building an SDN enterprise WLAN based on virtual APs," *IEEE Commun. Lett.*, vol. 21, no. 2, pp. 374–377, Feb. 2017.

[36] R. Bencel, K. Košťál, I. Kotuliak, and M. Ries, "Common SDN control channel for seamless handover in 802.11," in *Proc. Wireless Days (WD)*, Apr. 2018, pp. 34–36.

[37] H. Zhang, Z. Lu, X. Wen, and Z. Hu, "QoE-based reduction of handover delay for multimedia application in IEEE 802.11 networks," *IEEE Commun. Lett.*, vol. 19, no. 11, pp. 1873–1876, Nov. 2015.

[38] M. E. Berezin, F. Rousseau, and A. Duda, "Multichannel virtual access points for seamless handoffs in IEEE 802.11 wireless networks," in *Proc. IEEE 73rd Veh. Technol. Conf. (VTC Spring)*, May 2011, pp. 1–5.

[39] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, vol. 1. Belmont, MA, USA: Athena Scientific, 1995.

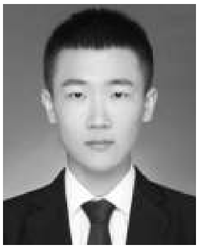
[40] J. N. Tsitsiklis, "Asynchronous stochastic approximation and Q-learning," *Mach. Learn.*, vol. 16, no. 16, pp. 185–202, Sep. 1994.

[41] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[42] P. Shah, U. K. Khankhoje, and M. Moghaddam, "Inverse scattering using a joint L_1–L_2 norm-based regularization," *IEEE Trans. Antennas Propag.*, vol. 64, no. 4, pp. 1373–1384, Apr. 2016.

[43] S. Liu, X. Hu, and W. Wang, "Deep reinforcement learning based dynamic channel allocation algorithm in multibeam satellite systems," *IEEE Access*, vol. 6, pp. 15733–15742, Feb. 2018.

[44] R. R. Fontes, S. Afzal, S. H. B. Brito, M. A. S. Santos, and C. E. Rothenberg, "Mininet-WiFi: Emulating software-defined wireless networks," in *Proc. 11th Int. Conf. Netw. Service Manage. (CNSM)*, Nov. 2015, pp. 384–389.



ZIJUN HAN received the B.S. degree in electronic and information engineering from the Nanjing University of Aeronautics and Astronautics, Nanjing, China. He is currently pursuing the M.S. degree in communications engineering with the Beijing University of Posts and Telecommunications. His current research interests include machine learning-based wireless network management and recognition using Wi-Fi.

TAO LEI received the B.S. degree in electronic and information engineering from the China University of Geosciences, Beijing, China. He is currently pursuing the Ph.D. degree in communications engineering with the Beijing University of Posts and Telecommunications. His current research interests include next generation wireless networks and software-defined wireless local access networks.

ZHAOMING LU received the Ph.D. degree from the Beijing University of Posts and Telecommunications, in 2012, where he joined the School of Information and Communication Engineering, in 2012. His research interests include open wireless networks, QoE management in wireless networks, software-defined wireless networks, and cross-layer design for mobile video applications.

XIANGMING WEN received the M.Sc. and Ph.D. degrees in information and communication engineering from the Beijing University of Posts and Telecommunications. He is currently the Director of the Beijing Key Laboratory of Network System Architecture and Convergence, where he is managing several projects related to open wireless networking. He is also the Vice President of the Beijing University of Posts and Telecommunications. His current research interests include radio resource and mobility management, software-defined wireless networks, and broadband multimedia transmission technology.

WEI ZHENG (M'07) received the B.E. and M.S. degrees in computer science and technology from the Shandong University of Science and Technology, China, in 2000 and 2003, respectively, and the Ph.D. degree in communication and information system from the Beijing University of Posts and Telecommunications, China, in 2006. She became the Staff of the Beijing Key Laboratory of Network System Architecture and Convergence, in 2011. She is currently an Associate Professor with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications. Her current research interests include wireless networks, radio resource management, and self-organizing networks.

LINGCHAO GUO received the B.S. degree in communication engineering from the China University of Petroleum, Qingdao, China, in 2017. She is currently pursuing the Ph.D. degree with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications. Her research interests include Wi-Fi sensing and Wi-Fi imaging.

• • •