Applying Deep Reinforcement Learning to Improve Throughput and Reduce Collision Rate in IEEE 802.11 Networks

Chih-Heng Ke[1], and Lia Astuti[2*]
[1] Department of Computer Science and Information Engineering, National Quemoy University
Kinmen, 892, Taiwan
[e-mail: smallko@gmail.com]
[2] Information Technology and Application, National Quemoy University
Kinmen, 892, Taiwan
[e-mail: liaastutii08@gmail.com]
*Corresponding author: Lia Astuti

Received August 7, 2021; revised November 21, 2021; accepted December 13, 2021; published January 31, 2022

Abstract

The effectiveness of Wi-Fi networks is greatly influenced by the optimization of contention window (CW) parameters. Unfortunately, the conventional approach employed by IEEE 802.11 wireless networks is not scalable enough to sustain consistent performance for the increasing number of stations. Yet, it is still the default when accessing channels for single-users of 802.11 transmissions. Recently, there has been a spike in attempts to enhance network performance using a machine learning (ML) technique known as reinforcement learning (RL). Its advantage is interacting with the surrounding environment and making decisions based on its own experience. Deep RL (DRL) uses deep neural networks (DNN) to deal with more complex environments (such as continuous state spaces or actions spaces) and to get optimum rewards. As a result, we present a new approach of CW control mechanism, which is termed as contention window threshold ($CW_{Threshold}$). It uses the DRL principle to define the threshold value and learn optimal settings under various network scenarios. We demonstrate our proposed method, known as a smart exponential-threshold-linear backoff algorithm with a deep *Q*-learning network (SETL-DQN). The simulation results show that our proposed SETL-DQN algorithm can effectively improve the throughput and reduce the collision rates.

Keywords: Contention window threshold ($CW_{Threshold}$), deep *Q*-learning network (DQN), deep reinforcement learning (DRL), smart exponential-threshold-linear backoff algorithm with deep *Q*-learning network (SETL-DQN), wireless networks.

1. Introduction

The number of internet users is increasing hugely as time goes on. At the start of October 2021, more than 62 percent of the world's total population, roughly 4.88 billion people worldwide, used the internet. Meanwhile, the data is continuously growing. Currently, the increase is at the annual rate of 4.8 percent, equal to an average of about 600,000 new users per day [1]. Hence, the wireless local area network (WLAN) has become very popular as one of the technologies that use radio waves for connecting to the internet by any device, such as laptop, smartphone, computer, public transportation, vehicle to vehicle, etc., anytime and anywhere [2].

Essentially, the use of Wi-Fi can fulfill the growing standards to evolve a much wider parameter space to set a richer usage. The performance of the radio waves used by Wi-Fi has been extensively studied in the past. There will always be an updated version with a faster speed and better performance when handling a multitude of devices. In 2020, the new IEEE 802.11 amendment (802.11ax) was launched. It improves the Wi-Fi network performance used by multiple devices and sends or receives the data packet faster [3]. Then, before we knew 802.11ax as the sixth version of Wi-Fi, 802.11ac as the fifth Wi-Fi had brought forward to see updating version of Wi-Fi using numbers and let the user easier to see the Wi-Fi version. It had started from 802.11b as the first version of Wi-Fi, 802.11a as the second version of the Wi-Fi, 802.11g as the third version of the Wi-Fi, 802.11n as the fourth version of the Wi-Fi. Therefore, if all versions are compared, we can technically see from the number that Wi-Fi 6 is the highest version of Wi-Fi generation [4].

As the popularity of IEEE 802.11 WLAN grows, the rise in the density of WLAN devices per access point has resulted in throughput performance degradation when transmitting the data packet. Sending and receiving the data packets at the same time will have incomplete inputs such as noise, disrupted or missing information, and so on. Moreover, the collision primarily occurs when a node accepts more than one packet at a time, meaning both nodes (stations) will clash because they have the same priority causing no packets correctly delivered [5]. Then, when the channel is sensed to be idle, it will wait for a random amount of time before transmitting data. This scheme is called random backoff, and the waiting duration corresponds to the contention window (CW). All stations select the random waiting duration uniformly over the range of [0, W] [6]. CW is used for carrier-sense multiple access (CSMA) with collision avoidance (CA) or distributed coordination function (DCF) as a basic access mechanism to prevent a collision by starting from the initial transmission [5], [6].

As a collision avoidance technique, CW optimization significantly impacts network performance. If the value of CW is small, the stations just need to wait less time for data transmission. However, the disadvantage is that stations have a higher probability of selecting the same CW value, which will easily cause collisions. If the value selected by CW is large, it is not easy for the stations to select the same CW value. Therefore, it will not cause collisions. However, the stations need to wait for a long time to send out the packet [7]. Accordingly, selecting the appropriate CW size is very important. Related studies try to increase network performance by proposing several algorithms in the references [8-12]. *Syed and Roh* [8] proposed an adaptive backoff algorithm for the CW (ABA-CW), in which the number of active stations was estimated by observing the channel status. *Gannoune and Robert* [9] used an enhanced distributed coordination function (EDCF), which allows each station to adjust the size of the minimum CW. Also, *Ksentini et al.* [10] proposed a determinist CW algorithm (DCWA), distinguished between the various backoff ranges associated and the various contention stages. *Karaca et al.* [11] proposed the backoff freezing mechanism that claimed

to achieve the maximum network throughput under the populated networks. *Choi et al.* [12] proposed the adaptive binary negative-exponential backoff (A-BNEB) to adjust the maximum CW size based on the number of competing stations.

From all the operation networking protocols optimization, there is a rise in attempts to improve network performance by using machine learning (ML) techniques, such as supervised learning, unsupervised learning, and reinforcement learning [13]. *Sutton and Barto* [14] explained that the effectiveness of the reinforcement learning (RL) algorithm learns certain interactions between actions and future outcomes over time, determining their performance in solving various tasks. Therefore, the RL has an advantage, such as its ability to interact with the surrounding environment from its own experience. There are several parameters, such as the agents (e.g., access point) that takes action (e.g., parameters optimization) at the state (e.g., collision probability) to get the rewards (e.g., optimizing throughput) as the network performance in an environment (e.g., radio waves). Many research works proposed to compare the RL algorithms in wireless networks to enhance the network performances [15-17]. Furthermore, some studies [15], [16] adopted *Q*-learning as one of the RL algorithms on the intelligent transport systems (ITS) as the potential roles at vehicular Ad-hoc Networks (VANETs). *Sun et al.* [17] proposed a generic framework of autonomous cell activation and customized physical resource allocation schemes, also adopted the *Q*-learning model to satisfy the QoS requirements of users in order to achieve low energy consumption.

However, the main disadvantage of *Q*-learning is its slow convergence rate due to its iterative nature and the fact that it does not rely on previous information when faced with new situations. *Zhang et al.* [18] explained that deep RL (DRL) uses deep neural networks (DNN) to deal with more complex environments. Furthermore, some studies [19], [20] adopt DRL to optimize various Wi-Fi parameters in highly dynamic and complex environments. In addition, the RL and DRL algorithms can now be used to study CW optimization because of the high computing capabilities in modern network devices. Some recent works [16], [21-25] discuss the CW optimization through the effectiveness of *Q*-learning and deep *Q*-learning network (DQN) algorithms, thus describing the problem of optimizing the CW value in mobile ad-hoc networks (MANETs), VANETs, and for both LTE-LAA and Wi-Fi networks.

Wydmański and Szott [26] applied centralized contention window optimization with DRL (CCOD) to predict the best CW values to improve saturation throughput in 802.11 wireless networks using DQN. CCOD used the legacy binary backoff algorithm with DQN for predicting the best CW values. The maximum CW value will be chosen at a high number of contending stations. Although, it succeeded in decreasing the collision rate. However, using the traditional algorithm will also increase the waiting time of extensive transmission delays before sending out the packet because the effectiveness of waiting time is also essential. Besides, *Ke et al.* [27] proposed a smart exponential-threshold-linear (SETL) algorithm scheme. After each transmission, the CW threshold value is applied to define how CW value behaves. When the CW value is smaller than the CW threshold value, the CW value is adjusted exponentially to reduce the collision possibility. Hence, the CW threshold value can quickly adapt to the networks with a low number of competing stations. Conversely, when the CW exceeds the CW threshold value, the CW size is adjusted linearly to avoid excessive transmission delays with a high number of competing stations.

According to the results of extensive simulations of the proposed SETL scheme, it outperforms any other related backoff algorithm methods, including binary exponential backoff (BEB), linear increase linear decrease (LILD), and exponential increase exponential decrease (EIED) in terms of saturation throughput and collision rate in both low and high network load. The CW threshold value of the SETL algorithm scheme is a fixed value.

Nevertheless, the scenario in this paper will set different values of CW threshold value at different network environments for packet transmission efficiency. However, the CW threshold value is not easy to define. Therefore, the DQN algorithm will be used via exploration during the learning phase to determine the CW threshold value. This research proposed the SETL-DQN algorithm, which aims directly to learn the optimum policy and leads to a better performance than related backoff algorithms, including legacy CSMA/CA, SETL [27], CCOD-DQN [26]. The legacy CSMA/CA will use BEB as a standard of 802.11 wireless networks. Hence, 802.11ac will be used to exhibit the experiment parameter operation.

The research paper will be divided into six-section to present more systematically. The first section will explain the introduction. The second section will explain the related works. The basic explanation of DQN will be presented in the third section, which is divided into the basic explanation of DRL and partially observable Markov decision process (POMDP) definition. The fourth section will explain the proposed method of the SETL-DQN algorithm scheme, which is divided into the SETL algorithm and SETL-DQN algorithm. The fifth section will explain the performance evaluation. The last section will discuss the conclusion.

2. Related Works

The applications of RL for Wi-Fi networks have currently been applied in various scenarios to enhance network performance [15], [17]. For instance, *Wu et al.* [15] proposed a protocol that can store the data in VANETs by transferring data to a new carrier (vehicle), where adopts the RL-based algorithm to consider long-term efficiency. *Sun et al.* [17] proposed an autonomous energy management framework using cell activation techniques and designed a *Q*-learning model with reduced state space size to consider varying resource demand and user population. Further, some studies [19], [20] adopted DRL to optimize various Wi-Fi parameters in highly dynamic and complex environments. *Balakrishnan et al.* [19] adopted DRL to the problem of allocating time and frequency resources in OFDMA wireless systems. *Bast et al.* [20] adopted DRL model that can dynamically optimize the slice configuration of unplanned Wi-Fi networks without expert knowledge.

The contention window (CW) is an integer with the range where has been determined by the PHY characteristic between a minimum value CW_{min} and a maximum value CW_{max}. The CW optimization can also use the *Q*-learning algorithm to increase the network's performance [16], [21-23]. *Pressas et al.* [16] proposed MAC protocol features a *Q*-learning-based algorithm to adjust the CW size from probabilistic rebroadcasts by randomly picking an action via exploration or exploitation to achieve the highest *Q*-value for its current state. *Han et al.* [21] proposed to adopt the *Q*-learning method on the cooperative learning algorithm and the non-cooperative learning algorithm by intelligently tuning the CW size for both LTE-LAA and Wi-Fi nodes. *Zerguine et al.* [22] proposed a mechanism based on *Q*-learning (MISQ) to optimize MAC protocols performance in MANETs, where each station selects the appropriate CW based on the transmitted number of data packets and the occurred collisions. *Cho* [23] proposed the RL agent based on *Q*-learning to control the data transmission rates in CSMA/CA wireless networks, where the agent observes the timeout event of packets and select appropriate modulation and coding schemes (MCS) to control the data transmission rates in order to make the use of available bandwidth effectively.

DQN is based on *Q*-learning and a value function-based DRL algorithm. Compared to simple *Q*-learning, DQN is an additional DNN to enable more effective reward extrapolation with yet unseen states. DQN algorithm is also used for CW optimization in more complex network scenarios to maximize a network-level utility used in references [24], [25]. *Kumar et*

al. [24] designed an intelligent node that can dynamically adapt its minimum CW (MCW) parameter to maximize a network-level utility without knowing the MCW values in other nodes. In another study [25], the authors proposed a self-adaptive MAC layer algorithm employing DQN with a novel contention information-based state representation to improve the performance of the V2V safety packet broadcast.

In [26], the authors proposed applying centralized contention window optimization with DRL (CCOD) method to the task saturation throughput 802.11 networks optimization by correctly predicting the best CW values and using DQN to increase the network throughput. However, the legacy binary backoff algorithm is used by CCOD with DQN for predicting the best CW values. Therefore, the BEB algorithm makes a longer waiting time before sending the packet transmission because the CW value will choose the maximum CW value at the heavy network load.

Related work [27] proposed the SETL backoff algorithm that has already proven better performance than the legacy binary backoff algorithm because of the use of the CW threshold value. The result shows that the CW threshold value chose the exponential result to quickly adapt with a light network load and fewer competing stations. Conversely, to avoid dramatically increasing the CW threshold value, it chose the linear result with the heavy load network and bigger competing stations. As a result, it will reduce the needed time for sending out the successful transmission packet. The SETL algorithm will be applied to different network scenarios. However, the CW threshold value is not easy to define. This paper proposes adopting DQN on the SETL backoff algorithm to stipulate the CW threshold value via exploration in the learning phase with more or less competing stations in the WLANs environment.

3. Applying DRL Tools to 802.11 Wireless Networks

3.1 Deep Reinforcement Learning (DRL)

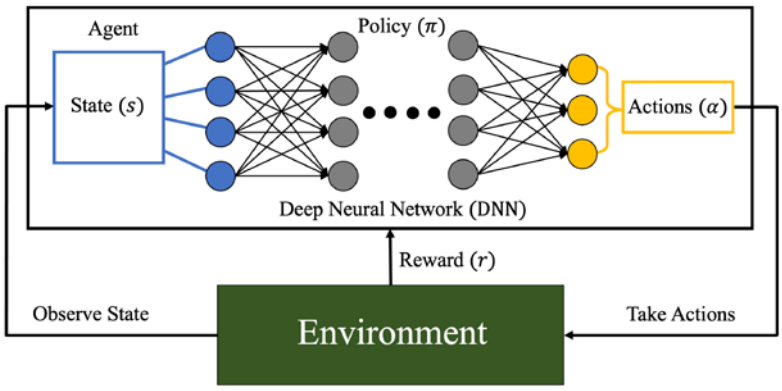

Fig. 1. Deep reinforcement learning (DRL) scheme.

RL has been used successfully for many works because its ability advantage of self-learning agent interacts with the surrounding environment based on the experience. Hence, the agent's primary purpose is to explore by observing the state to estimate the function of the environment and learning the right strategy for always performing the best action, and exploiting the agents to maximize the accumulated reward. The interaction way of the RL parameters is between the information exchange by the agent action and the environment from every state. The reward is given from the training process as the output by applying the strategy for enhancing

more best policy. Then, the agent will be able to determine the policy by DNN to estimate a future reward for yet unseen states. **Fig. 1** explains the interaction between the RL agent with the environment: (i) an agent evaluates the current state of the environment as the input of DNN, (ii) performs the action based on the policy (strategy) by DNN as the output, (iii) the agent optimizes its decision-making policy through a training phase until it discovers the correct decision for each state of the environment will be visited, (iv) the rewards were given from the environment by the appropriate behavior from the action has been taken.

3.2 Partially Observable Markov Decision Process (POMDP) Definition

Markov decision process (MDP) is the required setup framing problem optimization which consists of agent, action, state, and reward. It will be explained in more detail by partially observable Markov decision process (POMDP) [26], the tuple formally $(S, A, T, R, \Omega, O, \gamma)$ will describe for each element, as follow:

The agent applied in this paper is located in the access point (AP). The AP has a global view of observing all network environments and determining appropriate $CW_{Threshold}$ value. The AP can put the $CW_{Threshold}$ value in the beacon frames. Accordingly, the stations can set the $CW_{Threshold}$ value.

The *state* $(s \in S)$ is the current situation of the exact status for all connected devices to the wireless network.

The *action* $(a \in A)$ is carried out by the agents to set the new $CW_{Threshold}$ value updated through exploration by DQN in the learning phase, and the adjustment of action value is an integer a between 0 and 7. The definition of the new $CW_{Threshold}$ value is shown in (1), which is the range of the new $CW_{Threshold}$ value is from 128 to 1024.

$$CW_{Threshold} = 2^7 \times (1 + a) \tag{1}$$

When the action is determined, the current state will change to the next state $(s' \in S)$ according to the *transition probabilities* $T(s'|s, a)$

The network normalized throughput is the number of successfully delivered bits per second. It defines the *reward function* $r \in [0,1]$ in SETL-DQN. The normalized throughput is defined as the ratio of the obtained throughput to the channel bit rate. The estimated normalized throughput should be a real number between 0 and 1.

The *observation* $o \in \Omega$ is the collision probability history. The *current collision probability* P_{col} is defined as the unsuccessful transmission probability. P_{col} is calculated as shown in (2), where the number of transmitted frames is N_t and the number of correctly received frames is N_r. When the station transmitted the packet, piggyback can notify AP, and AP can know the number of N_t, and the number of N_r can be calculated by the AP node itself. The *previous state* and *current state* will be taken as the *history of recently observed collision probabilities* $H_{(P_{col})}$. By comparing the values between the two observations, we can know whether the packet collision rate increases or decreases at eachtime point. If the collision rate increases, the parameters can be adjusted and optimized again.

$$P_{col} = \frac{N_t - N_r}{N_t} \tag{2}$$

The *discount coefficient* (γ), corresponds to the long-term rewards over the comparison of immediate rewards. If γ is close to one, the agent will determine the importance of future rewards. Conversely, if γ is close to zero, the agent will be worthless in the future rewards.

4. Proposed SETL-DQN Algorithm

4.1 Smart Exponential-Threshold-Linear (SETL) Algorithm

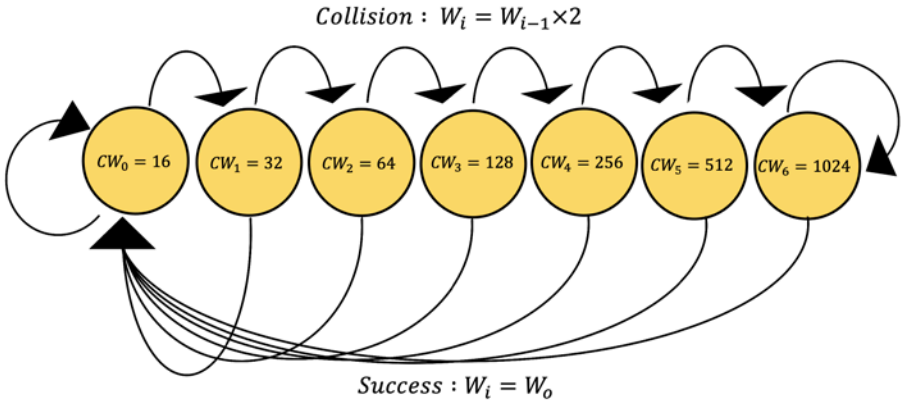

Fig. 2. Binary exponential backoff (BEB) scheme.

The CCOD-DQN algorithm proposed by [26] is succeeded in finding the best CW value, minimizing the collision rate, and maximizing the saturation throughput of IEEE 802.11 wireless networks. **Fig. 2** shows the CW settings for the binary exponential backoff scheme. The CW value is between minimum CW value CW_{min} and maximum CW value CW_{max}, with $CW_{min} = 16$, $CW_{max} = 1024$, and $m = 7$. The CW value will be doubled if the packet transmission fails. The CW value will be doubled continuously until reaching CW_{max} if transmissions keep failing. The successful packet transmission will reset the CW value into CW_{min}. When CW_{max} was chosen at the heavy network load and the bigger contending stations condition, a successful transmission will make the CW jump from a large value, i.e., CW_{max}, to a small value, i.e., CW_{min}. This phenomenon will make the station wait for a long time to send out the packet successfully. Because under a heavily loaded network, the smaller values of CW will cause the stations to get collided again. The stations need time to make CW value get bigger. Therefore, the transmission efficiency will be reduced.

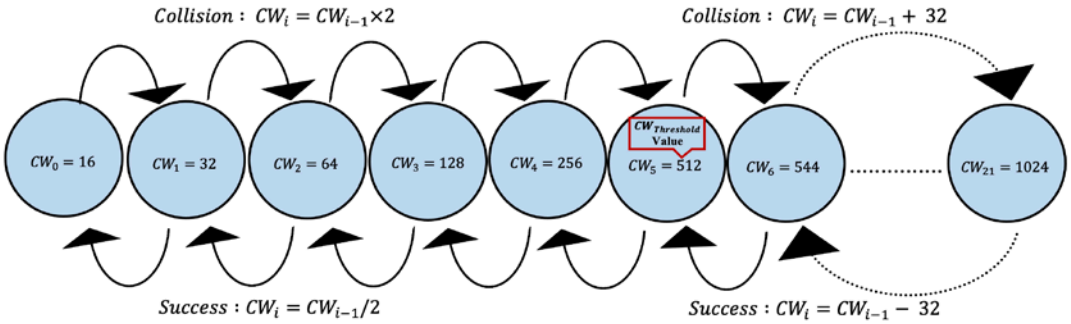

Fig. 3. Smart exponential-threshold-linear (SETL) backoff scheme.

Therefore, to solve these problems, the SETL backoff algorithm proposed by research [27] has been proven to minimize the collision rate, maximize the throughput and reduce the needed time (slightly idle time) for sending out the successful transmission packet. SETL algorithm takes advantage of EIED and LILD backoff algorithm to improve network performance in light and heavy network load conditions. **Fig. 3** is shown the SETL backoff algorithm. The

$CW_{Threshold}$ is set to determine whether there are fewer or more contending stations in the WLANs. If the CW size is smaller than the $CW_{Threshold}$ value, the CW size of the competing station will adjust exponentially because the CW value is in the lower region, which implies the light network load, where the number of collisions that occurred is lower. Conversely, if the CW size is bigger than $CW_{Threshold}$ value, the CW size of the competing station will adjust linearly to adapt the network because the CW value is in the higher region in which the heavy network load is detected, where the competing station has retransmitted many times due to the collisions. However, after each successful packet transmission, the SETL backoff algorithm will directly reduce the CW value for the next transmission either exponentially or linearly.

On the SETL backoff algorithm scheme. The CW will increase, and its increment will be decided by the $CW_{Threshold}$ value. When the CW value is smaller than $CW_{Threshold}$ value. The current CW value at unsuccessful packet transmission will be doubled, and the current CW value at successful packet transmission will be halved from the original CW value. In addition, when the CW value is bigger than $CW_{Threshold}$ value. The current CW value at unsuccessful packet transmission will be added 32 each time, and the current CW value at the successful packet transmission will be subtracted 32 each time. Furthermore, the $CW_{Threshold}$ is a fixed value, $CW_{Threshold} = 512$. Nevertheless, the scenario in this paper will be setting different values of $CW_{Threshold}$ for different network environment scenarios for optimizing packet transmission efficiency. However, the $CW_{Threshold}$ value is not easy to define. Therefore, to solve the problem, SETL-DQN will be proposed to train the agent via exploration at the learning phase and determine the appropriate $CW_{Threshold}$ value.

4.2 SETL-DQN Algorithm

This paper will adopt the DQN algorithm to the SETL algorithm, called SETL-DQN as the proposed algorithm to define the backoff threshold value via exploration at the learning phase. The self-learning agent is located at the AP, which interacts with the wireless network IEEE 802.11 (environment) to generate a new $CW_{Threshold}$ value (action) at the current state and then transit to the next state in the form of $T(s'|s,a)$. Moreover, the agent observes the collision probability P_{col}. The collision rates of the *previous state* and the *current state* are observed as the history of recently observed collision probabilities $H_{(P_{col})}$. In addition, the stations will execute a backoff algorithm. The new $CW_{Threshold}$ values will send to the stations via Beacon. Finally, the agents will get the reward (normalized throughput) as a successfully delivered packet. We define discount coefficient γ for future rewards corresponding to the long-term reward over immediate reward.

The detailed simulation model of the SETL-DQN algorithm can be shown by following **Algorithm 1**. The SETL-DQN algorithm operates in two phases. Firstly, the learning phase, which includes the pre-processing and the running process. Secondly, the evaluation phase will be entered after the model is trained. The first stage is the *learning phase*. The pre-processing of the DQN algorithm necessitates the setting of a few key parameters. The performance is determined by discount coefficient γ, which means the long-term rewards are more important than short-term rewards. Second, the various new hyperparameters added deep learning (DL) into RL algorithms, requiring each neural network to provide a learning rate α as an updated new value over an old value. Third, the optimization algorithm is used to optimize neural networks. Finally, DQN algorithms also used a replay buffer *B*, which provides the basis for mini-batch sampling by recording all interactions for the agent at the previous and current state. The interactions are stored in the replay buffer as (s,a,r,s') tuples.

Algorithm 1. Pseudo code for SETL-DQN algorithm

Notation

$H_{(P_{col})}$: history of recently observed collision probabilities
$load$: data sent since the last interaction
a : previous action
s : state
r : reward
$CW_{Threshold}$: the new contention window threshold

procedure LEARNING $(H_{(P_{col})}, load, a)$
  $s \leftarrow$ preprocess $H_{(P_{col})}$
  $r \leftarrow$ normalize $(load)$
  $agent.step\ (s, a, r)$ → Training a neural network
  $a' \leftarrow agent.act(s) + noise$
  $CW_{Threshold} \leftarrow 128 \times (1 + a')$
  return $CW_{Threshold}$
end procedure
procedure EVALUATION $(H_{(P_{col})})$
  $s \leftarrow$ preprocess $H_{(P_{col})}$
  $a \leftarrow agent.act(s)$ → Send to the neural network
  $CW_{Threshold} \leftarrow 128 \times (1 + a)$
  return $CW_{Threshold}$
end procedure

The running process algorithm was started by defining the model and forwarding the network structure of the *Q*-function. The history of collisions probability is the input, and the highest *Q*-value of all state-action pairs $[Q(s, a_1), Q(s, a_2), Q(s, a_3), \ldots, Q(s, a_n)]$ is the output. To model the *Q*-function, the dimension of action space used DQN to update the prediction value of the maximum *Q*-value network by converting to $onehot$ vector. Adam is selected as the optimization algorithm. An ε-greedy policy was implemented for the agent as the attenuation parameters. The model parameters are copied to the model target at a fixed number of training times. Every action has a selected random number probability, named exploration. A noise factor influences each action. A large value is usually used to allow for more exploration so the algorithm can transit to a different (s, a) pair and gain experience to get the best reward. In the beginning, we set the value of $CW_{Threshold}$ value to 128. The stations use the legacy CSMA/CA protocol to transmit data packets. Then during the training period, the agent will generate different actions through explorations by observing the collision probability to adjust the $CW_{Threshold}$ value and expect a better cumulative reward. In addition, as the training gradually converges, the degree of exploration slowly decreases to allow for more exploitation to select the largest subscript of Q-value for its current state.

The experience is added from the experience replay pool to store and execute the agent acts for the next state. Each observation caused the experience in the Wi-Fi network as the current collision probability (the transmission failure probability), which was calculated based on the number of transmitted frames and correctly received frames. Collision probability measurements are performed at predetermined intervals throughout interactions and reflect the appropriateness of the current $CW_{Threshold}$ value. In practice, the agent does not have immediate access to the current collision probability. However, all stations are sending the packets to the AP. The amount of data transmitted to AP from stations can be piggybacked to inform the agent located at the AP. The AP sends out the acknowledgment packets back to the stations. Then, the agent can get the number of correctly received data packets. The agent can

get the collision probability with the number of packets sent by stations and the number of packets correctly received at the AP.

After the model has been trained, the *evaluation phase* as the second stage has begun. During this stage, it only needs to observe the state, get the action through the trained model, and choose the appropriate $CW_{Threshold}$. Also, the agent will always choose the optimal action when the noise factor is zero. No more rewards are necessary because the agent is already considered completely trained and has stopped receiving updates. The agent is now ready for network deployment. Additionally, the algorithms smooth out the reward noise by separating the local and target neural networks. The local network determines the actions, while the learning process relies on target network predictions.

5. Performance Evaluation

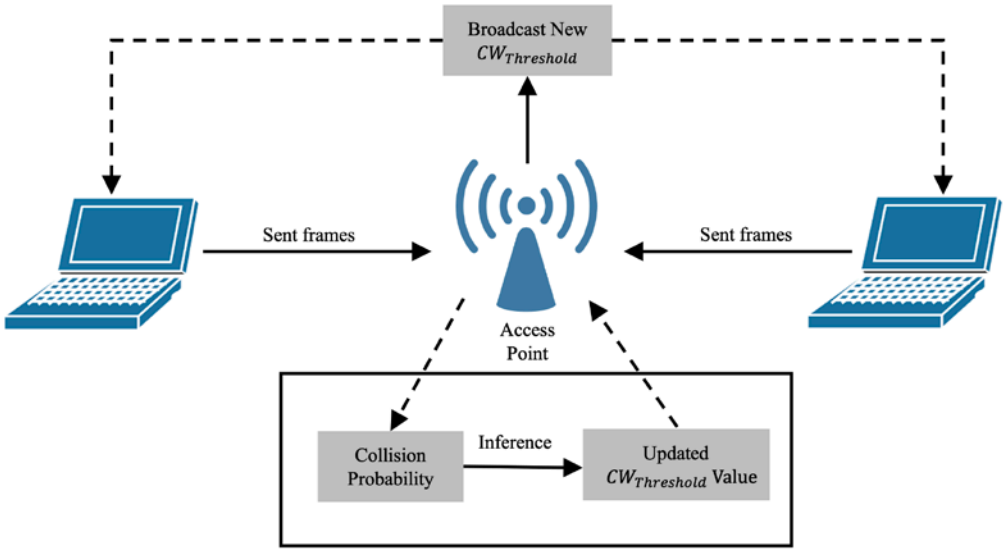

Fig. 4. The topology consideration with AP update $CW_{Threshold}$ when the station sent the data.

The medium access control (MAC) method of IEEE 802.11 has been simulated by this paper. To verify that the SETL-DQN algorithm has more advantages than the CCOD-DQN algorithm. We implemented the SETL-DQN algorithm environment using Parl and paddle-Paddle provided by Baidu [28] and python language. The SETL algorithm has been implemented based on [27], which the results show better performance than Legacy CSMA/CA, EIED, and LILD algorithms. To evaluate the performance of our proposed method, we will compare SETL-DQN with the legacy CSMA/CA, SETL [27], and CCOD-DQN [26].

5.1 Simulation Setup

The simulation topology is shown in **Fig. 4**. The following settings set the simulation experiment parameters: IEEE 802.11ac wave one, 24 non-overlapping unlicensed national information infrastructure (UNII) channels in 5 GHz frequency band, single-user transmissions. The simulations were implemented on a server with a CPU 2 GHz Quad-Core Intel Core i5. Then, we further assumed (i) perfect and immediate state information flow to the agent (i.e., the current values of N_t and N_r are known at the AP) and (ii) the immediate

setup of $CW_{Threshold}$ at each station. In practice, the $CW_{Threshold}$ AP should inform value in the periodic beacon. In the future study, we will implement this mechanism into a more realistic topology. The experiment parameters of IEEE 802.11ac are: 8184 bits of the packet payload, 272 bits of MAC header, 128 bits of PHY header, ACK is 112 bits + PHY header, 867 Mbps of channel bit rate, 9 μs of the slot time (σ), DIFS is 34 μs, SIFS is 16 μs, propagation delay is 1 μs, with $CW_{min} = 16$, and $CW_{max} = 1024$. In this scenario, we idealize the simulation settings to use the base performance of SETL-DQN before moving to a more realistic topology.

Table 1 shows the SETL-DQN algorithm parameters. The replay memory M is used to store the experience during the training process. The data amount will be given to the agent each time to utilize the iteration for sample batch size. The batch size value is 32 samples to estimate the error gradient before the training dataset updates the model weights. The optimization parameters also control how quickly the model adapts to define the appropriate $CW_{Threshold}$ trained by neural networks. The proposed algorithm uses a network structure consisting of a DNN with three hidden layers fully connected networks with 128 output dimensions. The third hidden layer is Q-value and rectified linear unit (ReLU) is used as a layer activation function. Randomness was incorporated into agent behavior and network simulation. Each experiment was run for 5,000 second interaction periods.

Table 1. SETL-DQN parameters for the training process

Parameters	Value
Learn frequency	5
Memory size	20.000
Memory warmup size	200
Batch size	32
Learning rate α	0.001
Discount factor γ	0.99
Epsilon greedy ε	0.1
Epsilon greedy ε decrement	1e-6

5.2 Simulation Result

At the simulation test, we assume that the environment is an error-free median and there is no hidden node. The node always has enough packets for transmission. The number of nodes has been set from 10 stations to 150 stations with 20 intervals. The learning phase has been run at 20 rounds to converge to the stable values. The first experiment evaluated the contention window value to compare SETL-DQN and CCOD-DQN algorithms. The result showed in **Fig. 5**. For the CCOD-DQN, the CW value is doubled for the number of competing stations is from 10 stations to 50 stations. Furthermore, the CW value turned into the maximum of CW in the heavy network loads, where $CW_{max} = 1024$. The large value of CW will cause the station to need a large waiting time to send out the packets. Conversely, the SETL-DQN backoff scheme has guaranteed the expected result by optimizing the $CW_{Threshold}$ value through adopting the DQN algorithm to the SETL algorithm to define threshold value. After training many times, $CW_{Threshold}$ value is determined by the exploration during the learning phase and succeeded in quickly adapting when the light or heavy network loads at more or less competing stations. In the starting from the 10 stations until 150 stations, the current $CW_{Threshold}$ is turned up linearly to prevent too much waiting time before packet transmission.

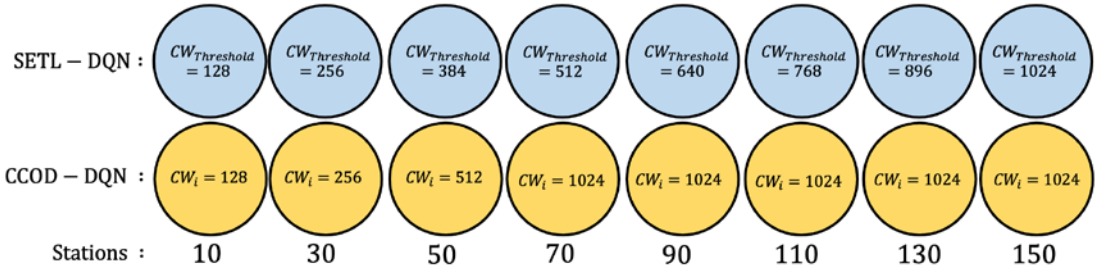

Fig. 5. SETL-DQN and CCOD-DQN contention window value comparison.

The static scenario was designed to compare the collision rate between Legacy CSMA/CA, SETL, CCOD-DQN, and SETL-DQN. In the second experiment, the higher number of competing stations, the higher the collision rate. In **Fig. 6**, the SETL algorithm is better than Legacy CSMA/CA. In comparison, the CCOD-DQN and SETL-DQN algorithms are more efficient in reducing the collision rate drastically than Legacy CSMA/CA and SETL algorithms. If we compare the collision rate of the SETL-DQN algorithm with the CCOD-DQN algorithm, we can find that the collision rate of SETL-DQN is slightly higher than CCOD-DQN. The main reason is that the CCOD-DQN will set the CW to the fixed CW_{max} to reduce the collision rate at a large number of contending stations. In our proposed method of the SETL-DQN algorithm, when the packet transmission is successful, the $CW_{Threshold}$ will be lowered to reduce the waiting time for packet transmission. As a consequence, at the same time, the lower of $CW_{Threshold}$ value at the heavy network loads will bring a higher collision rate than the COOD-DQN algorithm. However, the collision rate results between SETL-DQN and CCOD-DQN do not differ significantly. Therefore, SETL-DQN can still effectively reduce the collision rate.

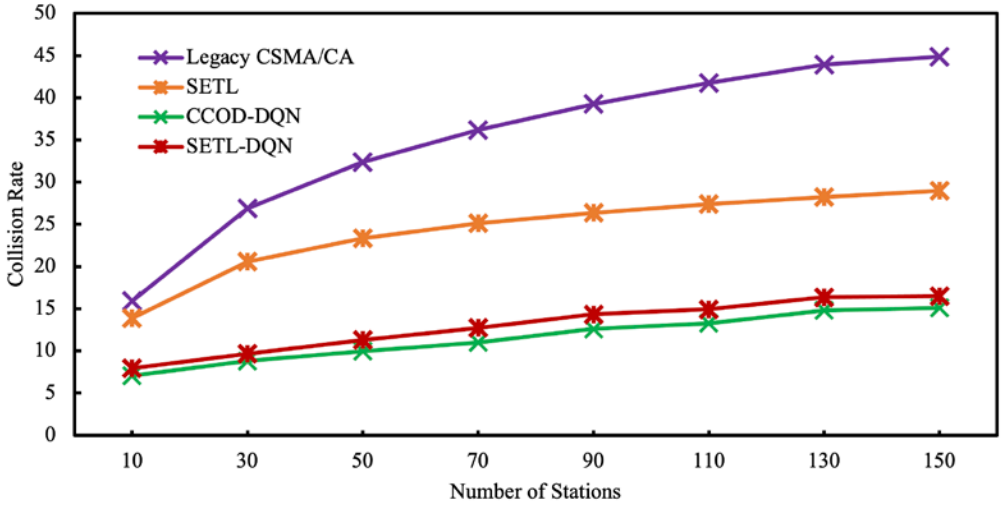

Fig. 6. Collision rate comparison for different schemes.

The third experiment of the normalized throughput comparison shows in **Fig. 7**. The SETL algorithm has better performance than Legacy CSMA/CA. The CCOD-DQN and SETL-DQN perform better than Legacy CSMA/CA and SETL algorithms. The CCOD-DQN scheme shows similar normalized throughput with the SETL-DQN algorithm in light network load. When the competing stations increase, the CW value of the CCOD-DQN algorithm cannot adapt to the heavy network load. Conversely, the SETL-DQN algorithm shows

outperformance over CCOD-DQN. At the competing 10 stations, the normalized throughput for SETL-DQN gets 0.545, and CCOD-DQN gets 0.548. The performance of SETL-DQN is only 3% lower than CCOD-DQN. Conversely, the proposed algorithm can show 55% better than CCOD-DQN for 150 stations scenario, i.e., the SETL-DQN gets 0.5, and the CCOD-DQN only gets 0.445. From the simulation results, we can see clearly that the proposed SETL-DQN can efficiently reduce the collision rate and improve the normalized throughput no matter under light or heavy network loads.

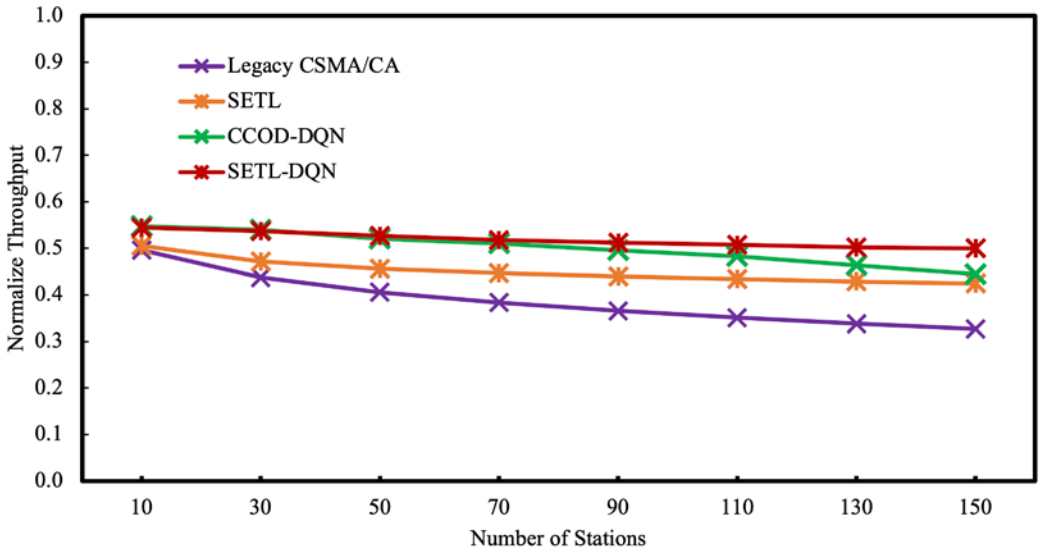

Fig. 7. Normalized throughput comparison for different schemes.

6. Conclusion

We have presented the SETL-DQN scheme which adopts deep *Q*-learning network (DQN) principles to smart exponential-threshold-linear (SETL) backoff algorithm to learn the correct $CW_{Threshold}$ value settings and improve the efficiency at IEEE 802.11ac environment as the experiment learning process. It has been effectively used to the CW optimization problem using the CW threshold value, $CW_{Threshold}$, SETL-DQN offers an efficient algorithm that we confirm (1) the $CW_{Threshold}$ value optimization can avoid excessive transmission delays before sending out the packet under light or heavy network loads, (2) the collision rate comparisons make SETL-DQN adapt quickly to the heavy network loads, (3) the normalized throughput of SETL-DQN shows the outperformance over the related backoff algorithms. The learning process has resulted in SETL-DQN obtaining a trained agent which can be directly applicable universally in any IEEE 802.11 access point (AP). Future studies should focus on applying SETL-DQN on multiagent and dynamic scenarios for removing or adding the stations with more realistic network conditions.

References

[1] Reportal Data, “Digital around the world,” *Datareportal*, April. 2021. [Online]. Available: https://datareportal.com/global-digital-overview

[2] T. A. Myrvoll, J. E. Håkegård, T. Matsui and F. Septier, “Counting public transport passenger using WiFi signatures of mobile devices,” in *Proc. of the IEEE 20th Int. Conf. on Intelligent Transportation Systems (ITSC)*, Yokohama, Japan, pp. 1-6, Oct. 2017. Article (CrossRef Link)

[3] E. Khorov, A. Kiryanov and A. Lyakhov and G. Bianchi, “A tutorial on IEEE 802.11 ax high efficiency WLANs,” *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 197-216, 2019. Article (CrossRef Link)

[4] K. Jacob, “Wi-Fi now has version numbers, and Wi-Fi 6 comes out next year,” *the verge*, Oct. 3, 2018. [Online]. Available: https://www.theverge.com/2018/10/3/17926212/wifi-6-version-numbers-announced. Accessed on: May 2, 2019.

[5] N. O. Song, B. J. Kwak, J. Song and M. E. Miller, “Enhancement of IEEE 802.11 distributed coordination function with exponential increase exponential decrease backoff algorithm,” in *Proc. of the 57th IEEE Semiannual Vehicular Technology Conf.*, Jeju, Korea (South), vol. 4, pp. 2775-2778, April, 2003. Article (CrossRef Link)

[6] S. Jang, J. G. Choi and S. Yoon, "Statistical estimation of the number of contending stations and its application to a multi-round contention resolution scheme," *KSII Transactions on Internet and Information Systems*, vol. 10, no. 9, pp. 4259-4271, 2016. Article (CrossRef Link)

[7] V. K. Garg, “Chapter 21 - Wireless local area networks,” in *Wireless Communications and Networking*, Ist ed. Isbn. Burlington, Massachussetts, US: Morgan Kaufmann Series in Networking, 2007, pp. 713-776. Article (CrossRef Link)

[8] I. Syed and B. H. Roh, “Adaptive backoff algorithm for contention window for dense IEEE 802.11 WLANs,” *Mobile Information Systems*, 2016. Article (CrossRef Link)

[9] L. Gannoune and S. Robert, “Dynamic tuning of the contention window minimum (CW/sub min/) for enhanced service differentiation in IEEE 802.11 wireless ad-hoc networks,” in *Proc. of IEEE 15th Int. Symposium on Personal, Indoor and Mobile Radio Communications*, Barcelona, Spain, vol. 1, pp. 311-317, Sep. 2004. Article (CrossRef Link)

[10] A. Ksentini, A. Nafaa, A. Gueroui and M. Naimi, “Determinist contention window algorithm for IEEE 802.11,” in *Proc. of IEEE 16th Int. Symposium on Personal, Indoor and Mobile Radio Communications*, Berlin, Germany, vol. 4, pp. 2712-2716, Sep. 2005. Article (CrossRef Link)

[11] M. Karaca, S. Bastani and B. Landfeldt, “Modifying backoff freezing mechanism to optimize dense IEEE 802.11 networks,” *IEEE Transactions on Vehicular Technology*, vol. 66. no. 10, pp. 9470-9482, 2017. Article (CrossRef Link)

[12] B. G. Choi, J. Y. Lee and M. Y. Chung, "Adaptive Binary Negative-Exponential Backoff Algorithm Based on Contention Window Optimization in IEEE 802.11 WLAN," *KSII Transactions on Internet and Information Systems*, vol. 4, no. 5, pp. 896-909, 2010. Article (CrossRef Link)

[13] F. Wilhelmi, S. B. Munoz, B. Bellalta, C. Cano, A. Jonsson and V. Ram, “A flexible machine-learning-aware architecture for future WLANs,” *IEEE Communications Magazine*, vol. 58, no. 3, pp. 25-31, 2020. Article (CrossRef Link)

[14] R. S. Sutton and A. G. Barto, “Reinforcement learning: an introduction,” *MIT Press*, Cambridge, MA, Feb. 1998. Article (CrossRef Link)

[15] C. Wu, T. Yoshinaga, Y. Ji, T. Murase and Y. Zhang, “A reinforcement learning-based data storage scheme for vehicular ad hoc networks,” *IEEE Transactions on Vehicular Technology*, vol. 66, no. 7, pp. 6336-6348, 2017. Article (CrossRef Link)

[16] A. Pressas, Z. Sheng, F. Ali, D. Tian and M. Nekovee, “Contention-based learning MAC protocol for broadcast vehicle-to-vehicle communication,” in *Proc. of IEEE Vehicular Networking Conference (VNC)*, Turin, Italy, pp. 263-270. Nov. 2017. Article (CrossRef Link)

[17] G. Sun, G. O. Boateng, H. Huang and W. Jiang, "A Reinforcement Learning Framework for Autonomous Cell Activation and Customized Energy-Efficient Resource Allocation in C-RANs," *KSII Transactions on Internet and Information Systems*, vol. 13, no. 8, pp. 3821-3841, 2019. Article (CrossRef Link)

[18] C. Zhang, P. Patras and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2224–2287, 2019. Article (CrossRef Link)

[19] R. Balakrishnan, K. Sankhe, V. S. Somayazulu, R. Vannithamby and J. Sydir, "Deep reinforcement learning based traffic-and channel-aware OFDMA resource allocation," in *Proc. of IEEE Global Communications Conference (GLOBECOM)*, Waikola, HI, USA, pp. 1-6, Dec. 2019. Article (CrossRef Link)

[20] S. D. Bast, R. T. Duran, A. Chiumento, S. Pollin and H. Gacanin, "Deep reinforcement learning for dynamic network slicing in IEEE 802.11 networks," in *Proc. of IEEE Conf. on Computer Communications Workshops (INFOCOM WKSHPS)*, Paris, France, pp. 264-269, April, 2019. Article (CrossRef Link)

[21] M. Han, S. Khairy, L. X. Cai, Y. Cheng and R. Zhang, "Reinforcement learning for efficient and fair coexistence between LTE-LAA and Wi-Fi," *IEEE Transactions on Vehicular Technology*, vol. 69, no.8, pp.8764-8776, 2020. Article (CrossRef Link)

[22] N. Zerguine, M. Mostefai, Z. Aliouat, Y. Slimani, "Intelligent CW selection mechanism based on *Q*-learning (MISQ)," *International Information and Engineering Technology Association*, vol. 25, no. 6, pp. 803-811, 2020. Article (CrossRef Link)

[23] S. Cho, "Rate adaptation with *Q*-learning in CSMA/CA wireless networks," *Journal of Information processing systems*, vol. 16, no. 5, pp. 1048-1063, 2020. Article (CrossRef Link)

[24] A. Kumar, G. Verma, C. Rao, A. Swami and S. Segarra, "Adaptive Contention Window Design Using Deep *Q*-Learning," in *Proc. of ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Toronto, ON, Canada, pp. 4950-4954, June, 2021. Article (CrossRef Link)

[25] C. Choe, J. Choi, J. Ahn, D. Park and S. Ahn, "Multiple channel access using deep reinforcement learning for congested vehicular networks," in *Proc. of IEEE 91st Vehicular Technology Conf.*, Antwerp, Belgium, pp. 1-6, May, 2020. Article (CrossRef Link)

[26] W. Wydmański and S. Szott, "Contention window optimization in IEEE 802.11 ax networks with deep reinforcement learning," in *Proc. of IEEE Wireless Communications and Networking Conf. (WCNC)*, Nanjing, China, pp. 1-6, March, 2021. Article (CrossRef Link)

[27] C. H. Ke, C. C. Wei, K. W. Lin and J. W. Ding, "A smart exponential-threshold-linear backoff mechanism for IEEE 802.11 WLANs," *International Journal of Communication Systems*, vol. 24. no. 8, pp. 1033-1048, 2011. Article (CrossRef Link)

[28] M. Sun, B. Jiang, H. Xiong, Z. He, H. Wu and H. Wang, "Baidu neural machine translation systems for WMT19," in *Proc. of the 4th Conf. on Machine Translation*, Florence, Italy, vol. 2, pp. 374-381, August, 2019. Article (CrossRef Link)

Chih-Heng Ke received his B.S. and Ph.D. degrees in electrical engineering from National Cheng-Kung University, in 1999 and 2007. He is an associate professor at the Department of Computer Science and Information Engineering in National Quemoy University, Kinmen, Taiwan. His current research interests include multimedia communications, wireless networks, and software defined networks.

Lia Astuti received her B.Ed. degree in International Program on Science Education from Indonesian University of Education, Bandung, Indonesia, in 2019, respectively. She is currently a master's student at Information Technology and Application at National Quemoy University, Kinmen, Taiwan. Her current research interests include teaching media for science education, reinforcement learning, and wireless networks.