A Deep Reinforcement Learning Scheme for SCMA-Based Edge Computing in IoT Networks

Pengtao Liu[1], Jing Lei[1], Wei Liu[1]
[1]College of Electronic Science and Technology, National University of Defense Technology, Changsha, 410073, China
E-mail: liupengtao15@nudt.edu.cn, leijing@nudt.edu.cn, wliu_nudt@nudt.edu.cn

GLOBECOM 2022 - 2022 IEEE Global Communications Conference | 978-1-6654-3540-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/GLOBECOM48099.2022.10001088

***Abstract*—The application of sparse code multiple access (SCMA) to multi-access edge computing (MEC) networks can provide massive connections as well as timely and efficient computation services for resource-constrained Internet of Things (IoT) devices. This paper investigates the maximization of computation rate in SCMA-MEC networks under a dynamic environment. We first formulate an initial optimization problem to maximize the long-term computation rate of IoT devices under task delay constraints. Then, a joint computation offloading and SCMA resource allocation algorithm based on long short-term memory (LSTM) network and dueling deep Q network (DQN) is proposed. Specifically, each IoT device acts as an agent in the algorithm. Since each device can only observe part of the environment state, the LSTM network is used to predict the states of other devices. The computation rate of devices is taken as a reward to conduct action exploration in dueling DQN, and then the near-optimal computation offloading decision, SCMA codebook allocation, and power distribution of IoT users are obtained after training. Numerical simulation results demonstrate that the proposed algorithm can achieve higher computation rate compared with other baseline schemes.**

***Index Terms*—Sparse Code Multiple Access (SCMA), Multi-Access Edge Computing (MEC), Deep Reinforcement Learning (DRL), computation offloading, resource allocation.**

I. Introduction

With the advent of beyond 5G (B5G) era and the rapid development of the Internet of Things (IoT), plenty of resource-constrained applications and latency-sensitive tasks are increasing in terminals. However, the limited computing capability of IoT devices can hardly meet the latency requirement of tasks. As a new computing paradigm, multi-access edge computing (MEC) can provide timely and effective computing services for IoT users [1].

Non-orthogonal multiple access (NOMA) technology allows multiple IoT devices to share the same time slot and frequency resources by dividing the transmission power or codebooks [2]. Sparse code multiple access (SCMA) technology is a code-domain NOMA (CD-NOMA) scheme, which obtains a series of non-orthogonal sequences through sparse spread spectrum then distributes to multiple users [3]. Compared with the power-domain NOMA (PD-NOMA) scheme, SCMA has additional advantages, such as the coding and shaping gains [4], [5], providing improved throughput and connectivity [6].

This work is supported by the National Natural Science Foundation of China (No. 61502518 and 61702536). (Corresponding author: Jing Lei)

In recent years, the combination of NOMA and MEC in IoT scenarios has been regarded as a promising approach to providing efficient transmission and timely computation services for massive IoT devices. Through theoretical analysis, the authors in [7] proved that NOMA can effectively reduce the delay and energy consumption during the process of computation offloading. The minimization of offloading latency in NOMA-MEC networks was studied in [8]. In [9], a NOMA-MEC optimization framework was constructed to minimize the energy consumption of IoT devices by optimizing user clustering, resource allocation, and transmission power. An SCMA-based MEC scheme was investigated in [10] to improve the connectivity and throughput in IoT scenarios. In [11], [12], a joint resource allocation and computation offloading optimization algorithm in SCMA-MEC networks was proposed to achieve the tradeoff between task completion time and energy consumption of IoT devices.

While the aforementioned studies focused on static channel environments, the channel states and task requests of IoT devices change dynamically over time in actual IoT scenarios. Classical optimization methods usually require high computational complexity to find a optimal solution, and it is difficult to consider tasks in a dynamic time dimension. Fortunately, deep reinforcement learning (DRL) can be applied to IoT networks for real-time decision-making tasks such as dynamic resource allocation and computation offloading policy. In [13], the task offloading and subcarrier assignment problems in NOMA-MEC networks were studied with the goal of maximizing the sum computation rate, and a DRL-based algorithm was adopted to solve this problem. An online computation offloading algorithm based on DRL was proposed to minimize the total energy consumption of IoT devices for a time-varying dynamic channel scenario in the NOMA-MEC system [14]. To the authors' knowledge, DRL-based design and optimization scheme for SCMA-MEC networks in the dynamic environment has not been provided before, which is the motivation of this paper.

The contribution of this paper is shown below. Considering time-varying channel and random task generation in a dynamic environment, we jointly optimize computation offloading decision, SCMA codebook allocation, and power distribution to maximize the long-term computation rate of IoT devices. Due to the dynamic and non-convex nature of the optimization problem, we propose a joint optimization algorithm based on long short-term memory (LSTM) network and dueling deep Q

network (DQN), which utilizes the LSTM network to predict the states of other IoT devices and uses dueling DQN to explore and train actions to obtain the near-optimal policy.

II. System Model

We consider an SCMA-based MEC system model in a dynamic environment, where the base station (BS) is equipped with a MEC server located in the center of the cell. There are U randomly moving IoT users, denoted as $\mathcal{U} = \{1, 2, \ldots, U\}$. The time range of the dynamic environment can be expressed as $t \in \mathcal{T} = \{1, 2, ..., T\}$. During each time slot t, every IoT device is considered static and generates a computation task with a probability of ρ. Every generated task $T_u(t)$ is characterized by three parameters, $(d_u(t), \phi_u(t), t_u^{\max}(t))$, where $d_u(t)$ [bit] represents the amount of raw data, $\phi_u(t)$ [CPU cycles/bit] denotes the computational density, and $t_u^{\max}(t)$ [s] specifies the maximum task delay. The local computing speed is defined as $f_u^l(t)$ [CPU cycles/s] of user u at time slot t. In this model, computation tasks are regarded as indivisible, i.e., binary offloading is adopted. Define IoT devices' offloading policy as $\boldsymbol{x} = \{x_u(t) | u \in \mathcal{U}, t \in \mathcal{T}\}$, users can choose to perform local computing ($x_u(t) = 0$) or offload the task to the MEC server ($x_u(t) = 1$). When the task excutes locally, the computation rate can be calculated as $f_u^l(t)/\phi(t)$ [bit/s]. If multiple IoT devices request computing services, they can offload computing tasks to MEC servers simultaneously through SCMA.

A. Computation Offloading through SCMA

SCMA is a NOMA system that combines the mapping part of orthogonal amplitude modulation (QAM) with a low-density signature spreader [15]. We denote the set of codebooks and subcarriers in the SCMA system as $\mathcal{C} = \{1, 2, \ldots, C\}$ and $\mathcal{K} = \{1, 2, \ldots, K\}$, respectively. The relation between subcarriers and codebooks in SCMA can be expressed by the indicator matrix $\boldsymbol{F}_{K \times C}$, where the entries in the matrix are defined as $f_{k,c}$. If $f_{k,c} = 1$, it represents codebook c occupies the kth subcarrier. An SCMA encoder can be defined as a mapping from $log_2(M)$ bits to a K-dimensional complex codeword of size M [16]. An SCMA codeword is a sparse vector with d_c non-zero values, which means $\|\boldsymbol{F}\|_1 = d_c$ and each codebook occupies d_c subcarriers. Each subcarrier is used by the SCMA codebook at most d_s times, i.e., $\|\boldsymbol{F}\|_\infty = d_s$. Let $p_{u,c}$ represent the transmission power of IoT user u on SCMA codebook c, which is allocated to subcarrier k in proportion to $\eta_{c,k}(t) \in [0, 1]$, satisfying $\sum_{\forall k \in c} \eta_{c,k}(t) = 1$ [6]. In the SCMA system, multiple access is achieved by assigning different codebooks to multiple devices. The indicator of codebook assignment is denoted by $\delta_{u,c}(t) \in \{0, 1\}$. If codebook c is allocated to IoT user u at time slot t, $\delta_{u,c}(t) = 1$, else $\delta_{u,c}(t) = 0$. In order to reduce interference between different users, it is assumed that a codebook is assigned to at most one IoT user, i.e., $\sum_{c \in \mathcal{C}} \delta_{u,c}(t) \leqslant 1$. Define the codebook assignment and power allocation policy at time slot t as $\boldsymbol{\delta} = \{\delta_{u,c}(t) | u \in \mathcal{U}, c \in \mathcal{C}, t \in \mathcal{T}\}$ and $\boldsymbol{\eta} = \{\eta_{c,k}(t) | c \in \mathcal{C}, k \in \mathcal{K}, t \in \mathcal{T}\}$, respectively. The signal to interference plus noise ratio (SINR) of IoT device u on the SCMA codebook c at time t is given by

$$\gamma_{u,c}(t) = \sum_{k \in c} \frac{p_{u,c}(t) \eta_{c,k}(t) \left| h_{u,k}(t) \right|^2}{\sigma_{u,k}^2(t) + I_{u,k}(t)}, \tag{1}$$

where $h_{u,k}(t)$ is defined as the channel gain between the IoT device u and BS on subcarrier k at time t. We assume that $h_{u,k}(t)$ is constant in each time slot t and changes dynamically in different time slots. $\sigma_{u,k}^2(t)$ denotes the power of the background noise. Let $\boldsymbol{S}_k(t)$ denote the users occupying the same subcarrier k. $I_{u,k}(t)$ represents the interference of user u on subcarrier k [17], given by

$$I_{u,k}(t) = \sum_{i \in \{\boldsymbol{S}_k(t) || h_{i,k}(t)|^2 > |h_{u,k}(t)|^2\}} p_{i,c}(t) \eta_{c,k}(t) \left| h_{i,k}(t) \right|^2. \tag{2}$$

With normalized bandwidth, the transmission rate of IoT device u at time slot t can be calculated as

$$\begin{aligned} R_u(t) &= \sum_{c \in \mathcal{C}} \delta_{u,c}(t) \log_2 \left(1 + \gamma_{u,c}(t)\right) \\ &= \sum_{c \in \mathcal{C}} \delta_{u,c}(t) \log_2 \left(1 + \sum_{k \in c} \frac{p_{u,c}(t) \eta_{c,k}(t) |h_{u,k}(t)|^2}{\sigma_{u,k}^2(t) + I_{u,k}(t)}\right). \end{aligned} \tag{3}$$

B. Problem Formulation

In the practical application, the computation capacity of MEC server is much stronger than IoT devices and the amount of result data that needs to be returned is very small. Compared with the offloading time, there is a big gap for the sum of computation and return time, which is more than three orders of magnitude and thus can be ignored, as mentioned in [18]. The sum computation rate at time slot t in SCMA-MEC networks is calculated as

$$R_t(\boldsymbol{x}, \boldsymbol{\delta}, \boldsymbol{\eta}) = \sum_{u \in \mathcal{U}} \left[(1 - x_u(t))(f_u^l(t)/\phi_u(t)) + x_u(t) R_u(t)\right]. \tag{4}$$

The optimization objective of this paper is to maximize the long-term computation rate of SCMA-MEC system in the dynamic environment and the optimization problem is expressed as follows,

$$\max_{\boldsymbol{x}, \boldsymbol{\delta}, \boldsymbol{\eta}} \quad \lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} R_t(\boldsymbol{x}, \boldsymbol{\delta}, \boldsymbol{\eta}) \tag{5a}$$

$$s.t. \quad (1 - x_u(t)) \frac{f_u^l(t)}{\phi_u(t)} + x_u(t) R_u(t) \geqslant \frac{d_u(t)}{t_u^{\max}(t)} \tag{5b}$$

$$x_u(t) \in \{0, 1\}, \forall u \in \mathcal{U}, t \in \mathcal{T} \tag{5c}$$

$$\delta_{u,c}(t) \in \{0, 1\}, \forall u \in \mathcal{U}, c \in \mathcal{C}, t \in \mathcal{T} \tag{5d}$$

$$\eta_{c,k}(t) \in [0, 1], \forall c \in \mathcal{C}, k \in \mathcal{K}, t \in \mathcal{T} \tag{5e}$$

$$\sum_{c \in \mathcal{C}} \delta_{u,c}(t) \leqslant 1, \forall u \in \mathcal{U}, t \in \mathcal{T} \tag{5f}$$

$$\sum_{\forall k \in c} \eta_{c,k}(t) = 1, \forall c \in \mathcal{C}, t \in \mathcal{T}. \tag{5g}$$

The constraint conditions in the above problem can be interpreted as follows: Constraint (5b) notes the computation time of local and MEC cannot exceed the task deadline $t_u^{\max}(t)$. Constraint (5c) indicates the binary variable of computation offloading decision. Constraint (5d) represents the variable that assigns SCMA codebooks to the IoT devices. Constraint (5e) shows the proportion of power allocated to the occupied subcarriers. Constraint (5f) implies that a codebook is assigned to at most one IoT user. Constraint (5g) means that the sum of the power distribution ratios of all subcarriers occupied by the codebook c equals 1.

III. DRL-based Dynamic Computation Offloading and SCMA Resource Allocation Scheme

Since Problem (5) is a non-convex optimization problem in a dynamic environment, it is challenging to solve it with traditional optimization algorithms. On one hand, the optimization objective of this problem is to maximize the long-term computation rate. Although the problem can be solved by traditional optimization algorithms for each time slot, it leads to extremely high computational complexity. On the other hand, traditional non-convex optimization algorithms usually need to obtain the global information of the environment, which is difficult to obtain in a highly dynamic environment for IoT devices. Therefore, we propose an algorithm based on DRL, which can use the agent to learn the approximate optimal task offloading and SCMA resource allocation strategies from the dynamic environment with partial channel state information. The proposed online learning algorithm avoids solving the non-convex optimization problem of the dynamic channel in each time slot, thus greatly reducing the complexity.

A. Proposed DRL framework

In this part, a DRL-based framework is proposed to solve the Problem (5). Based on the original long-term optimization problem, we firstly design the state space, action space, and reward function as follows.

State space: Let $\boldsymbol{s}_{u,t}$ denote as the state of environment observed by IoT user u at time slot t. The first part is the observed channel gain on different subcarriers, which is represented by $\boldsymbol{h}_u(t) = \{h_{u,k}(t), k \in \mathcal{K}\}$. The second part is the task information $T_u(t)$ generated at time slot t. Finally, the channel states, tasks and actions of other devices at the previous moment $t-1$ observed by the device u from BS broadcast are represented by $\boldsymbol{H}(t-1) = \{h_{u,k}(t-1), u \in \mathcal{U}, k \in \mathcal{K}\}$ $\boldsymbol{T}(t-1) = \{T_u(t-1), u \in \mathcal{U}\}$, and $\boldsymbol{A}(t-1) = \{\boldsymbol{a}_{u,t-1}, u \in \mathcal{U}\}$, respectively. Therefore, the state can be expressed as $\boldsymbol{s}_{u,t} = \{\boldsymbol{h}_u(t), T_u(t), \boldsymbol{H}(t-1), \boldsymbol{T}(t-1), \boldsymbol{A}(t-1)\}$ with size $(k+3) \times (U+1) + (d_c+2) \times U$.

Action space:The action space of IoT devices includes the decision of whether to carry out computation offloading ($x_u(t)$), the selection of an SCMA codebook ($\delta_{u,c}(t)$), and the allocation of the power factor on the occupied subcarriers ($\eta_{c,k}(t)$). The action of agent u at time slot t can be expressed as $\boldsymbol{a}_{u,t} = \{x_u(t), \delta_{u,c}(t), \eta_{c,k}(t)\}$. In practice, each IoT user only needs to select one SCMA codebook, and each codebook occupies d_c subcarriers. Hence, there are $d_c + 2$ elements in the action $\boldsymbol{a}_{u,t}$. We can discretize the ratio of power on different subcarriers into 9 levels, i.e., $\eta_{c,k}(t) \in \{0.1, 0.2, ..., 0.8, 0.9\}$. The dimension of the action $\boldsymbol{a}_{u,t}$ can be calculated as $(d_c - 1) \times 9 \times C + 1$. For the most common SCMA system with factor matrix $\boldsymbol{F}_{4\times 6}$, the dimension of the action space is 55.

Reward function: $r_{u,t}$ is defined as the reward after the action $\boldsymbol{a}_{u,t}$ is taken at time slot t. The reward function $r_{u,t}$ for guiding learning should be consistent with the optimization objective in Problem (5), so the immediate reward function of this framework is the task computation rate of IoT device u, expressed as

$$r_{u,t} = \begin{cases} (1 - x_u(t))\dfrac{f_u^l(t)}{\phi_u(t)} + x_u(t)R_u(t), & r_{u,t} \geqslant \dfrac{d_u(t)}{t_u^{\max}(t)} \\ 0, & otherwise. \end{cases} \tag{6}$$

By setting the reward function as shown in (6), the IoT device can get more rewards when choosing the strategy of computation offloading and SCMA resource allocation with a higher computation rate. When the computation rate can not meet the delay requirement of the task, the reward is set to 0, i.e., the agent gets punished.

Every IoT device u collects information through interaction with the SCMA-MEC environment and updates its action strategy through the feedback of rewards and punishments. At each time slot t, the IoT device u, as an agent, observes a state $\boldsymbol{s}_{u,t}$ and selects an action $\boldsymbol{a}_{u,t}$ from the action space according to Q value, expressed as

$$\begin{aligned} Q(\boldsymbol{s}_{u,t}, \boldsymbol{a}_{u,t}) &= \mathbb{E}\left[R_{u,t}\right] = \mathbb{E}\left[\sum_{i=0}^{\infty} \gamma^i r_{u,t+i}\right] \\ &= \mathbb{E}\left[r_{u,t} + \gamma Q\left(\boldsymbol{s}_{u,t+1}, \boldsymbol{a}_{u,t+1}\right)\right], \end{aligned} \tag{7}$$

where $R_{u,t}$ denotes the accumulative discount reward, and $0 \leqslant \gamma \leqslant 1$ is the discount factor.

When the dimension of state-action space is huge, the corresponding relationship can also be approximated by deep neural networks (DNNs) with weights $\boldsymbol{\theta}$. Dueling DQN improves the convergence of training through experience replay and fixed target network. Experience replay method is used to randomize data and eliminate correlations in the observation data series. A group of transitions $(\boldsymbol{s}_{u,t}, \boldsymbol{a}_{u,t}, r_{u,t}, \boldsymbol{s}_{u,t+1})$ are randomly selected from the experience pool as a training batch. The target network has the same structure as the train network, but the weights are different. The parameters remain fixed for a certain number of steps and then update towards the train network. The fixed target network reduces the correlation between the target and the estimated Q value. The input of the DNNs is the state $s_{u,t}$, and the output is Q value of each action $Q(\boldsymbol{s}_{u,t}, \boldsymbol{a}_{u,t}|\boldsymbol{\theta})$. Given the state $s_{u,t}$ and action $a_{u,t}$, the output is only determined by the weights $\boldsymbol{\theta}$, which is updated by back propagation during the training phase. In the training process of dueling DQN, the loss function $L(\boldsymbol{\theta})$ can be written as

$$L(\boldsymbol{\theta}) = \mathbb{E}\left[\left(y - Q(\boldsymbol{s}_{u,t}, \boldsymbol{a}_{u,t}|\boldsymbol{\theta})\right)^2\right], \tag{8}$$

where $y = r_{u,t} + \gamma \max_{\boldsymbol{a}_{u,t+1}} Q\left(\boldsymbol{s}_{u,t+1}, \boldsymbol{a}_{u,t+1}|\boldsymbol{\theta}^{-}\right)$ and $\boldsymbol{\theta}^{-}$ means that the weights of target network. After the action $\boldsymbol{a}_{u,t}$ taken by agent u, the environment changes to a new state, and the agent gets a reward $r_{u,t}$, i.e., the computation rate.

B. Proposed DRL Network Architecture

In the distributed scheme, each IoT user acts as an agent and learns to make computation offloading decisions as well as SCMA codebook and power allocation strategies. IoT devices can only observe part of the environment state and do not know the channel and task status of other users. However, SCMA resource allocation depends on other users' computation offloading decisions. Therefore, LSTM network is introduced into dueling DQN network to predict channel and task states of other devices. The proposed structure of DNNs mapping states to actions in dueling DQN is introduced as follows.

(1) *Input layer*: The inputs are the states $\boldsymbol{s}_{u,t}$ observed by IoT devices. The historical channel states $\boldsymbol{H}(t-1)$, task states $\boldsymbol{T}(t-1)$ and actions $\boldsymbol{A}(t-1)$ are transmitted to the LSTM layer.

(2) *LSTM layer*: This layer is used to learn the dynamics of channel states and tasks to predict current states of other devices, which is achieved through an LSTM network, using a time-dependent approach to predict future changes in time series.

(3) *Fully connected layer*: The fully connected (FC) layer with rectified linear unit (ReLu) activation function learns to map the observed states and the predicted states to Q values.

(4) *Dueling layer*: The dueling layer first learns state value $V(\boldsymbol{s}_{u,t}|\boldsymbol{\theta})$ and action advantage value $A(\boldsymbol{s}_{u,t}, \boldsymbol{a}_{u,t}|\boldsymbol{\theta})$ respectively. Then, V and A are used to form the Q value $Q(\boldsymbol{s}_{u,t}, \boldsymbol{a}_{u,t}|\boldsymbol{\theta}) = V(\boldsymbol{s}_{u,t}|\boldsymbol{\theta}) + A(\boldsymbol{s}_{u,t}, \boldsymbol{a}_{u,t}|\boldsymbol{\theta})$ of the state-action pairs. This method can improve learning efficiency and enhance convergence by separating the contribution of state and action to Q value.

(5) *Output layer*: The output layer determines the Q value of each action, and the dimension is $(d_c - 1) \times 9 \times C + 1$.

C. Proposed DRL Algorithm

The specific steps of the training algorithm for joint computation offloading and SCMA resource allocation are shown in Algorithm 1. In the algorithm, initialization is performed first, followed by the training of dueling DQN. In the initialization process, some SCMA-MEC environment parameters and dueling DQN training parameters are taken as input. The parameters of SCMA-MEC network environment include including SCMA factor matrix F, the number of codebooks C, and the number of subcarriers K. Dueling DQN training parameters include train episodes E, experience replay memory D_u, minibatch size N, learning rate α, and the replace steps of target network's weights x. The weights of the training network and target network are initialized to the same random number $\boldsymbol{\theta}_0$. During the training procedure, state normalization and action selection are worth noting. In each episode, the environment parameters of SCMA-MEC networks are reset. The agent u observes part of the environment states. Since the states may not be of the same order of magnitude, the normalization method Min-Max scaling is used, which can normalize the same type of state data to $[0, 1]$. In each time slot t, the normalized state $\boldsymbol{s}_{u,t}$ is taken as the input to the training network to obtain Q values for different actions. To balance the exploration and exploitation, the action $\boldsymbol{a}_{u,t}$ is determined by the ϵ-policy, expressed as

$$\boldsymbol{a}_{u,t} = \begin{cases} \text{random action}, & p > \epsilon \\ \text{argmax}_a Q(\boldsymbol{s}_{u,t}, \boldsymbol{a}|\boldsymbol{\theta}), & p \leqslant 1 - \epsilon, \end{cases} \tag{9}$$

where p is a random number uniformly distributed in $[0, 1]$.

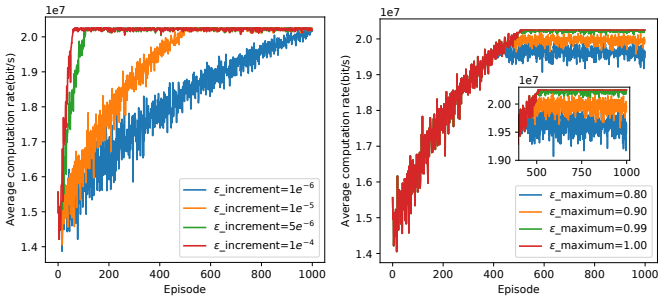
Algorithm 1: DRL-based Joint Computation Offloading and SCMA Resource Allocation Training Algorithm

Input: F, K, C, U, T, γ, E, D_u, N, α, x.
Output: The weights of dueling DQN $\boldsymbol{\theta}$.

```
Initialize: Set the weights θ of train network and the weights θ⁻ of target network with same random θ_0.
for episode=1, 2, ..., E do
    Reset the simulation parameters of SCMA-MEC networks. Initialize and normalize the state.
    for t=1, 2, ..., T do
        for u=1, 2, ..., U do
            Input s_{u,t} into the train network and obtain Q(s_{u,t}, a|θ).
            Select the action from (9).
            Observe the reward r_{u,t} and normalize the next state s_{u,t+1}.
            Store experience (s_{u,t}, a_{u,t}, r_{u,t}, s_{u,t+1}) in D_u.
            Sample N experiences as a train batch from D_u.
            Update θ by minimizing L(θ) in (8) using the train network.
            if mod (t, x) = 0 then
                θ⁻ = θ.
            end
        end
    end
end
```

IV. Simulation Results and Analysis

In this section, representative simulation results are shown to evaluate the performance of proposed DRL-based dynamic joint computation offloading and SCMA resource allocation algorithm in IoT systems. Consider a uplink SCMA-MEC network, where $U = 25$ IoT devices are randomly distributed in a $1 \times 1\text{km}^2$ region and the BS equipped with an MEC server is located in the center of the region. This system adopts the common SCMA scheme with the indicator matrix $\boldsymbol{F}_{4\times 6}$ and each codebook occupies $d_c = 2$ subcarriers. The dynamic observation range of the system is 200s with 100 time slots and 2s per time slot. At the beginning of each time slot, IoT



(a) ϵ_increment (b) ϵ_maximum

Fig. 1. The training process of the proposed algorithm under different (a) ϵ_increment and (b)ϵ_maximum values.

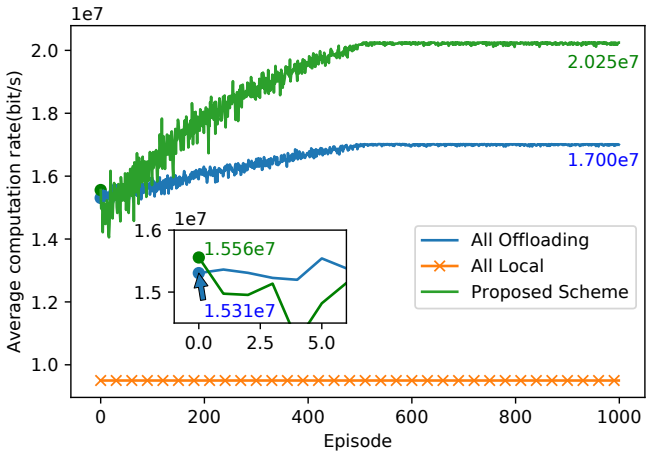


Fig. 2. Comparsion of average computation rate among three offloading schemes.

devices generate computing tasks with a probability of $\rho = 0.3$ and perform tasks locally or request computation services. The channels between the users and BS are generated by a distance dependent path loss, modeled as $PL(\text{dB}) = 140.7 + 36.7log_{10}(d)_{[km]}$ with 8dB log-normal multipath shadowing [19]. The maximum transmit power of IoT devices p_u is set as 23dBm. We set the system bandwidth B as 10MHz and the noise power $\sigma^2_{u,k}$ as -100dBm. We assume the generated task $T_u(t)$ with raw data $d_u(t)$ within $[300, 1200]$KB, the computational density $\phi_u(t) = 100$ CPU cycles/bit, and the maximum delay $t_u^{\max}(t)$ in $[0.5, 2]$s. The local computing speed $f_u^l(t)$ is random distributed within $[0.2, 1]$GHz for different IoT devices. The training parameters are set as train episodes $E = 1000$, experience replay memory $D_u = 2000$, minibath size $N = 64$, learning rate $\alpha = 0.01$, reward discount factor $\gamma = 0.8$ and the replace steps of target network's weights $x = 100$. Adam optimizer is adopted in the dueling DQN training process.

In order to encourage action exploration in the training process, we first simulated the convergence performance of the proposed algorithm under different ϵ_increment and ϵ_maximum values, as shown in Fig. 1. To ensure the fairness of different exploration parameters, we randomly generated and fixed the channel gain and task arrival of different time slots in each episode. Fig. 1(a) illustrates that the average reward, i.e., the average computation rate of IoT devices increases gradually and eventually converges as the number of episodes increases. The default value of ϵ_increment for subsequent simulation is chosen as 10^{-5}, which can balance the action exploration and the number of episodes (510) required to achieve convergence. The training process under different ϵ_maximum values $0.8, 0.9, 0.99, 1$ is shown in Fig. 1(b). In the experience accumulation and preliminary exploration stage, the initial exploration with different ϵ_maximum is consistent due to the fixed channel and task arrival. In the convergence phase, when the ϵ_maximum is smaller, the fluctuation of the convergence value is greater. To ensure the dynamics of action selection, the default value of ϵ_maximum is selected as 0.99.

Fig. 2 compares the average computation rate of IoT devices when selecting all offloading, all local computing and the proposed scheme. All offloading means that all IoT devices are offloading to MEC server for computation services, and SCMA codebook allocation and power distribution based on the proposed DRL algorithm is still adopted. As can be seen from the figure, the average computation rate of the local devices remains 0.95×10^6 bit/s during the training process. At the beginning (episode $= 1$), random codebook and power are adopted. The computation rate of full offloading, i.e., the sumrate of SCMA, is 1.531×10^7 bit/s. Through learning and training, the final sumrate of SCMA is 1.700×10^7 bit/s with an increase of 11%. In the proposed scheme, the computation rate is 1.556×10^7 bit/s when the random policy is adopted. In the initial episodes, the computation rate is lower than that of full offloading because some IoT devices randomly adopt local computation. However, with continuous training, the proposed scheme converges to 2.025×10^7 bit/s, which is 19% higher than that of full offloading. This can be interpreted as more interference with multiple devices simultaneously requesting computation services in the process of full offloading.

TABLE I
SAMPLE TEST RESULTS USING A TRAINED DUELING DQN NETWORK

Policys \ Users	1	2	3	4	5	6	7	8
Offloading	1	1	1	0	0	1	1	1
Codebook	1	4	5	/	/	2	3	6
Power	0.7/0.3	0.4/0.6	0.3/0.7	/	/	0.6/0.4	0.7/0.3	0.3/0.7
Rate	2.01e7	1.96e7	2.24e7	9.51e6	8.92e6	2.11e7	1.88e7	2.31e7

TABLE I shows some test results of the proposed algorithm using the trained dueling DQN network. Since the test is conducted over a period of time, we selected the test results at a certain time slot. At the time slot, 8 users out of 25 devices generated computation tasks. The policys for each device include the offloading decision, SCMA codebook selection, power allocation on occupied subcarriers. As can be seen from the table, the fourth and fifth users choose local computing. The remaining 6 users choose SCMA codebook 1, 4, 5, 2, 3, 6, respectively. $0.7/0.3$ represents the subcarrier

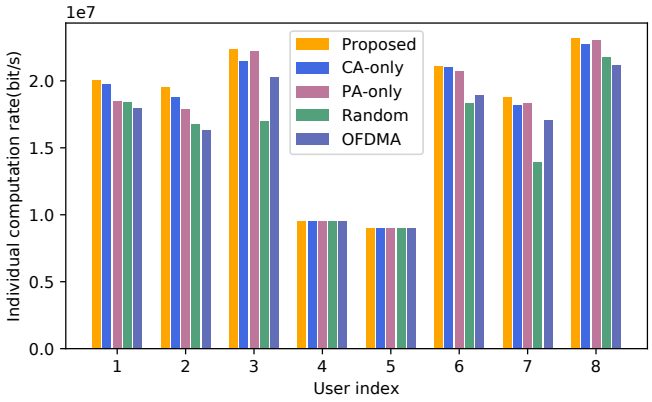


Fig. 3. Comparsion of individual computation rate for four baseline access schemes.

power distribution occupied by the SCMA codebook. When the transmit power of the IoT device is 0.2w (23dBm), the distributed power for the two subcarriers is 0.14w and 0.06w, respectively.

The following four baseline schemes are used for performance comparison. CA-only indicates SCMA codebook allocation and equal power $(0.5/0.5)$. PA-only means random codebook and proposed power distribution. Random stands for random codebook and equal power. OFDMA refers to the access scheme using orthogonal frequency division multiple access (OFDMA) instead of SCMA. Fig. 3 shows the comparison between the individual computation rate of the above four baseline schemes and the proposed scheme. It can be seen from the figure that the computation rate of the proposed scheme is superior to that of OFDMA. In addition, the computation rate of IoT devices requesting computation offloading is different due to the different channel conditions. For different users, the effect of SCMA codebook allocation and power distribution is different. For instance, SCMA codebook allocation plays a vital role for IoT devices 1,2 and 6, while the advantage of power allocation is more pronounced among users 3, 7 and 8. Compared with random resource allocation, the propsed SCMA resource optimization scheme of user 3 and user 7 is greatly improved, with 31.8% improvement from 1.70×10^7 bit/s to 2.24×10^7 bit/s and an increase of 35.3% from 1.39×10^7 bit/s to 1.88×10^7 bit/s, respectively.

V. Conclusions

In this paper, we investigated the long-term computation rate maximization problem in SCMA-MEC networks under a dynamic environment. Based on LSTM and dueling DQN network architecture, a joint computation offloading and SCMA resource allocation algorithm was proposed. Each IoT device acted as an agent, which predicted the states of other devices through the LSTM layer, and decided its offloading decision, SCMA codebook selection, and power factor distribution by dueling DQN. Simulation results demonstrated that the proposed algorithm has great advantages over the benchmark algorithms. The computation rate of the proposed scheme was 19% higher than that of all offloading scheme, and nearly 30% better than that of OFDMA and random SCMA resource allocation scheme.

References

[1] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 2017.

[2] X. Wang, Y. Zhang, R. Shen, Y. Xu, and F.-C. Zheng, "DRL-based energy-efficient resource allocation frameworks for uplink NOMA systems," *IEEE Internet Thing J.*, vol. 7, no. 8, pp. 7279–7294, 2020.

[3] H. Nikopour and H. Baligh, "Sparse code multiple access," in *Proc. IEEE 24th Annu. Int. Symp. Pers. Indoor Mobile Radio Commun. (PIMRC)*. IEEE, 2013, pp. 332–336.

[4] L. Yu, Z. Liu, M. Wen, D. Cai, S. Dang, Y. Wang, and P. Xiao, "Sparse code multiple access for 6G wireless communication networks: Recent advances and future directions," *IEEE Commun. Stand. Mag.*, vol. 5, no. 2, pp. 92–99, 2021.

[5] J. V. C. Evangelista, Z. Sattar, G. Kaddoum, and A. Chaaban, "Fairness and sum-rate maximization via joint subcarrier and power allocation in uplink SCMA transmission," *IEEE Trans. Wireless Commun.*, vol. 18, no. 12, pp. 5855–5867, 2019.

[6] M. Moltafet, N. M. Yamchi, M. R. Javan, and P. Azmi, "Comparison study between PD-NOMA and SCMA," *IEEE Trans. Veh. Technol.*, vol. 67, no. 2, pp. 1830–1834, 2018.

[7] Z. Ding, P. Fan, and H. V. Poor, "Impact of non-orthogonal multiple access on the offloading of mobile edge computing," *IEEE Trans. Commun.*, vol. 67, no. 1, pp. 375–390, 2019.

[8] Z. Ding, D. W. K. Ng, R. Schober, and H. V. Poor, "Delay minimization for NOMA-MEC offloading," *IEEE Signal Processing Lett.*, vol. 25, no. 12, pp. 1875–1879, 2018.

[9] A. Kiani and N. Ansari, "Edge computing aware NOMA for 5G networks," *IEEE Internet Thing J.*, vol. 5, no. 2, pp. 1299–1306, 2018.

[10] A. Alnoman, S. Erkucuk, and A. Anpalagan, "Sparse code multiple access-based edge computing for IoT systems," *IEEE Internet Thing J.*, vol. 6, no. 4, pp. 7152–7161, 2019.

[11] P. Liu, J. Lei, and W. Liu, "An optimization scheme for SCMA-based multi-access edge computing," in *Proc. IEEE 93rd Veh. Technol. Conf.(VTC-Spring)*, 2021, pp. 1–6.

[12] P. Liu, K. An, J. Lei, G. Zheng, Y. Sun, and W. Liu, "SCMA-based multiaccess edge computing in IoT systems: An energy-efficiency and latency tradeoff," *IEEE Internet Thing J.*, vol. 9, no. 7, pp. 4849–4862, 2022.

[13] M. Nduwayezu, Q.-V. Pham, and W.-J. Hwang, "Online computation offloading in NOMA-based multi-access edge computing: A deep reinforcement learning approach," *IEEE Access*, vol. 8, pp. 99 098–99 109, 2020.

[14] L. Qian, Y. Wu, F. Jiang, N. Yu, W. Lu, and B. Lin, "NOMA assisted multi-task multi-access mobile edge computing via deep reinforcement learning for industrial internet of things," *IEEE Trans Ind. Informat.*, vol. 17, no. 8, pp. 5688–5698, 2021.

[15] L. Yu, P. Fan, D. Cai, and Z. Ma, "Design and analysis of SCMA codebook based on star-QAM signaling constellations," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 10 543–10 553, 2018.

[16] X. Li, Z. Gao, Y. Gui, Z. Liu, P. Xiao, and L. Yu, "Design of power-imbalanced SCMA codebook," *IEEE Trans. Veh. Technol.*, vol. 71, no. 2, pp. 2140–2145, 2022.

[17] B. Di, L. Song, and Y. Li, "Radio resource allocation for uplink sparse code multiple access (SCMA) networks using matching game," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2016, pp. 1–6.

[18] S. Bi and Y. J. Zhang, "Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading," *IEEE Trans. Wireless Commun.*, vol. 17, no. 6, pp. 4177–4190, 2018.

[19] T. X. Tran and D. Pompili, "Joint task offloading and resource allocation for multi-server mobile-edge computing networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 856–868, 2019.