Algorithm 1 Pseudeocode of the Proposed Method

Establish a pair of DQNs (a trained DQN with weights θ and a target DQN with weights $\boldsymbol{\theta}^-$) and an empty experience pool M for all CUs;
Initialize the trained DQN wih random weights; set $\boldsymbol{\theta}^- = \boldsymbol{\theta}$;
In time slot $t(M \leq M_b)$, CU takes action randomly and stores the corresponding experience $\langle s, a, r, s' \rangle$ in its experience pool.
repeat
    CU k observe its state s_k in time slot t, $\forall k \in K$;
    In time slot $t(t > M_b)$ CU k chooses an action a_k according to ϵ-greedy policy; agnet k chooses an action $a_k = \arg\max_{a \in A} q(s_k, a; \boldsymbol{\theta}_k)$ with probability $(1 - \epsilon)$, or randomly chooses an action $a_k \in A$ with probability ϵ, $\forall k \in K$;
    CU k executes the chosen action a_k, then gets an immediate reward $r_k = R(s_k, a_k)$, $\forall k \in K$;
    CU k observes a new state s'_k in time slot $t + 1$, $\forall k \in K$;
    CU k saves its new experience $\langle s_k, a_k, r_k, s'_k \rangle$ into experience pool;
    Samples a mini-batch consisting of M_b experiences from experience pool M;
    Updates the weights $\boldsymbol{\theta}$ of trained DQN according to BP base on difference between reward in experience pool and output Q-value of target DQN;
    Updates $\boldsymbol{\theta}^-$ with $\boldsymbol{\theta}$ every T_{step} time slots;
until $R(t) - R(t-1) < \gamma$