A DDPG-based Energy Efficient Federated Learning Algorithm with SWIPT and MC-NOMA

Manh Cuong Ho[a], Anh Tien Tran[a], Donghyun Lee[a], Jeongyeup Paek[a], Wonjong Noh[b*], Sungrae Cho[a*]

[a]School of Computer Science and Engineering, Chung-Ang University, Seoul 06974, South Korea.

[b]School of Software, Hallym University, Chuncheon 24252, South Korea

Abstract

Federated learning (FL) has emerged as a promising distributed machine learning technique. It has the potential to play a key role in future Internet of Things (IoT) networks by ensuring the security and privacy of user data combined with efficient utilization of communication resources. This paper addresses the challenge of maximizing energy efficiency in FL systems. We employed simultaneous wireless information and power transfer (SWIPT) and multi-carrier non-orthogonal multiple access (MC-NOMA) techniques. Also, we jointly optimized power allocation and central processing unit (CPU) resource allocation to minimize latency-constrained energy consumption. We formulated an optimization problem using a Markov decision process (MDP) and utilized a deep deterministic policy gradient (DDPG) reinforcement learning algorithm to solve our MDP problem. We tested the proposed algorithm through extensive simulations and confirmed it converges in a stable manner and provides enhanced energy efficiency compared to conventional schemes.

Keywords: Deep reinforcement learning, federated learning, multi-carrier non-orthogonal multiple access, SWIPT

1. Introduction

The development and future evolution of wireless systems indicate a preference towards decentralization. This is due to privacy concerns and limited communication resources for data transmission, which makes it infeasible for all wireless devices to send their gathered data to a central data center. Centralized machine learning algorithms and analysis rely on data being readily available. Therefore, distributed learning frameworks are gaining demand and would allow wireless devices to collaboratively construct a shared learning model by training their local data. The emerging federated learning (FL) framework contains some of the most promising distributed learning algorithms and is expected to be adopted in future Internet of Things (IoT) systems.

In federated learning, wireless devices collectively perform a learning task by simply uploading local learning models to a base station without needing to share all their training data. For successful implementation of FL in wireless networks, local training updates from wireless devices must be transmitted over wireless links [1].

However, the limited availability of wireless resources, including bandwidth, transmit power, and delay time, can adversely affect FL performance. The real world constraints of wireless devices pose a significant challenge for FL deployment. We must prioritize energy efficiency and delay reduction in optimizing FL implementation to overcome the limitations posed by wireless networks.

1.1. Related Works and Contributions

Recently, many research efforts have focused on reducing energy consumption and latency for FL in wireless networks [2, 3, 4, 5, 6, 7]. In [2], Zhu et al. decreased latency by employing a broadband analog aggregation multi-access scheme that leveraged the waveform superposition property of a multi-access channel. In [3], Zeng et al. focused on reducing energy consumption while maintaining adequate learning performance through bandwidth allocation and scheduling optimization. Next, Tran et al. [4] minimized both local computation and wireless transmission energy by optimizing adjustments between latency and energy consumption. Bouzinis et al. In [5], Wu et al. minimized the total energy consumption of a base station (BS) under simultaneous wireless information and power transfer (SWIPT)-assisted FL networks with non-orthogonal multiple access (NOMA). Li et al. [6] minimized long-term energy consumption while ensuring FL convergence in a SWIPT-assisted FL network through dynamic resource allocation and UE scheduling. Finally, Chen et al. [7] delved into the complexities associated with minimizing the FL loss function. Their investigation considered not only the intricacies of local computation but also factored in the energy expended during the transmission process.

However, the proposed solutions in [2, 3, 4, 5, 6, 7, 8, 9] require synchronous uploading of learning models from all users.

*Corresponding author

Email addresses: hmcuong@uclab.re.kr (Manh Cuong Ho[a]), attran@uclab.re.kr (Anh Tien Tran[a]), dhlee@uclab.re.kr (Donghyun Lee[a]), jpaek@cau.ac.kr (Jeongyeup Paek[a]), wonjong.noh@hallym.ac.kr (Wonjong Noh[b]), srcho@cau.ac.kr (Sungrae Cho[a])

Real world limitations surrounding this issue significantly degrade convergence of their proposed FL systems. To address this issue, recent works have investigated asynchronous uploading from all users [10, 11, 12, 13]. [10] minimized the total delay in an FL round by jointly optimizing computational and communication-related resources. In [11], Yang et al. aimed to minimize the overall energy consumption of the system by considering local computing energy and wireless transmission energy while also satisfying a latency constraint. Next, Pham et al. [12] maximized the efficiency of unmanned aerial vehicle (UAV) transmit power to solve user equipment (UE) battery limitations for sustainable UAV-based FL wireless networks. Finally, in [13], Pham et al. aimed to reduce energy consumption for both an airborne server and UE to promote sustainable FL in a UAV-based system with edge computing and wireless power transfer.

On the other hand, the prevailing designs of FL model have predominantly concentrated on optimizing some fundamental performance indicators, commonly referred to represent by metrics include convergence speed, latency efficiency, energy efficiency, and the accuracy of the model classified into synchronous uploading and asynchronous uploading. However, it is noteworthy that, in many instances, these indicators have been addressed separately, as outlined in table 1. This compartmentalized approach to addressing model accuracy, convergence speed, and energy efficiency has been a common trend in existing FL designs.

1.2. Contributions and Organizations

This paper investigates optimal resource allocation to enhance energy efficiency for multi-carrier non-orthogonal multiple access (MC-NOMA) and SWIPT-based FL systems that support asynchronous uploading. The main contributions of this work are as follows

- First, we formulated a joint power and central processing unit (CPU) resource allocation problem that minimizes energy consumption while satisfying the latency constraint in federated learning systems.
- Second, we transformed this joint optimization problem into a Markov decision process (MDP) formulation, which is a NP-hard problem, and proposed a deep deterministic policy gradient (DDPG)-based reinforcement learning algorithm to efficiently utilize a continuous variables action set.
- Third, we showed the algorithms' polynomial complexity and stable convergence. Also, through extensive simulations over line-of-sight (LoS) and non-line-of-sight (NLoS) channels, we proved its enhanced energy efficiency compared with other basline schemes.

2. System Model and Problem Formulation

2.1. SWIPT-Assisted MC-NOMA Network

We consider a FL system that consists of a BS as shown in Fig. 1. The BS is equipped with a FL server and

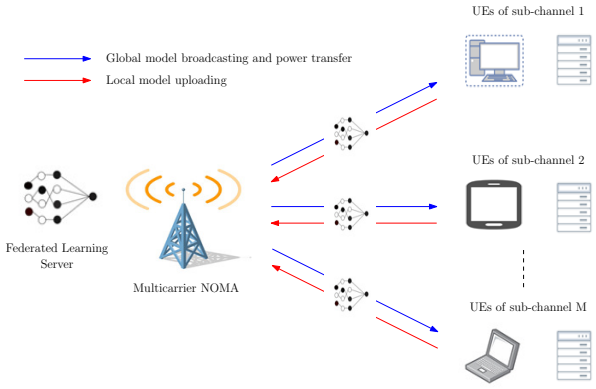


Figure 1. The system model.

$\mathcal{N} = \{1, 2, ..., N\}$ rechargeable battery-powered IoT user devices (UEs). For downlink transmission, the FL server broadcasts the global learning model to all UEs. Meanwhile, in the uplink, we employ MC-NOMA transmission with $\mathcal{M} = \{1, 2, ..., M\}$ sub-channels. The operation period of the system is divided into T time slots which are denoted by the index $t \in \mathcal{T} = \{1, 2, ..., T\}$. We assume that: (i) the communication between the FL server and each UE is primarily characterized by a NLoS link [14, 12, 15], and (ii) all channel conditions between the BS and the UEs remain relatively static and unchanged during the FL iterations. The channel gain of UE $n \in \mathcal{N}$ at time slot $t \in \mathcal{T}$ is calculated as $g_{n,t} = \iota\hat{\beta}_0(d_{n,t})^{-\alpha}$, where ι, α, $\hat{\beta}_0$, and $d_{n,t}$ are the additional attenuation factor, the path loss exponent, the reference channel gain at $d_0 = 1m$, and the distance between UE n and the BS, respectively.

For low-energy UE nodes, we employ energy harvesting by using SWIPT. During the downlink duration, each UE receives data and harvests energy through power splitting technology. The ratio between energy harvesting and data reception is determined by the power splitting ratio κ. $P\kappa_n$ denotes receiving data while $P(1-\kappa_n)$ denotes harvesting energy with $\kappa \in [0, 1]$ [5]. Next, we denote the harvested energy of UE n at time slot t by $E^{ha}_{n,t}$. Following [5], the harvested energy from the FL server of UE n can be expressed as

$$E^{ha}_{n,t} = \eta_0 \tau^{down}_t P\,(1-\kappa_n)\, g_{n,t}, \tag{1}$$

where P, $\eta_0 \in (0, 1]$, and $g_{n,t}$ are the transmit power of the BS, the energy conversion efficiency of UE n, and the channel gain between the UE n and the BS, respectively. In addition, we assumed that each UE n has a battery level $\psi_{n,t}$ (Joule) at the beginning of timeslot t. Then, the battery level of the UE in the following timeslot can be expressed as:

$$\psi_{n,t+1} = \min\left(\max(\psi_{n,t} - E^{cp}_{n,t} - E^{cm}_{n,t} + E^{ha}_{n,t}, 0), \psi^{max}\right), \tag{2}$$

where ψ^{max}, $E^{cp}_{n,t}$ and $E^{cm}_{n,t}$ denotes the maximum battery capacity of a UE, the computation energy consumption, and the communication energy consumption of UE n, respectively.

2.2. Federated Learning Process

Each UE has a local dataset $\mathcal{D}$ with D data samples expressed in bits. All UEs are equipped with an on-board computing pro-

Table 1
Comparison of proposed approach and existing studies.

Reference/ Indicators	Uploading	Delay efficiency	Accuracy	Convergence speed	Energy efficiency
[2]	Synchronous	O			
[3, 4, 5, 6]	Synchronous				O
[10]	Asynchronous	O			
[11, 12, 13]	Asynchronous			O	O
Proposed	Asynchronous	O	O	O	O

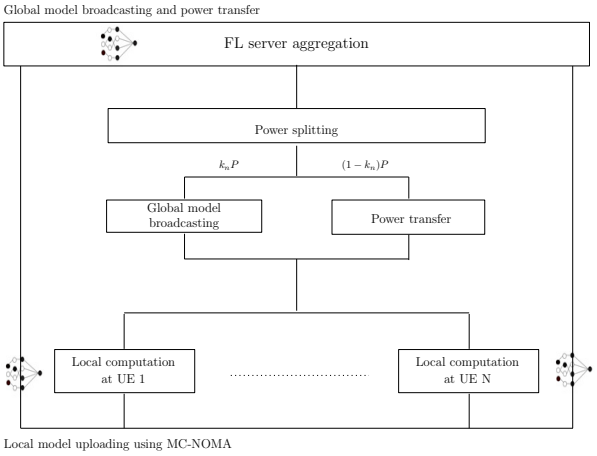


Figure 2. A system diagram illustrates how FL, SWIPT, and MC-NOMA are leveraged.

cessor responsible for training a local model. Additionally, each UE is equipped with an energy circuit to harness energy transmitted from the FL server. Each UE can efficiently handle multiple tasks concurrently, including harvesting energy from the FL server, performing FL tasks, and communicating with the FL server [15, 12, 13]. Without loss of generality, we also assume that the computational power of the FL server is sufficiently fast compared to the UEs. This ensures that the time required for aggregating all the local models at the FL server can be neglected when compared to the downlink time, the computing time at the UEs, and the transmission uplink time [11, 12, 16].

One FL round with SWIPT and MC-NOMA consists of a four step process as shown in Fig. 2 with each time duration, where τ_t^{down}, τ_t^{cp}, τ_t^{cm} and τ_t^{agg} are the model broadcast time, the energy harvesting time, the model upload time, and the time to build a global model, respectively. The detailed process is as follows:

- Step 1: At the beginning of t, the FL server broadcasts the global parameter ω_t and transfers energy to all UEs with τ_t^{down}.
- Step 2: Each UE receives the harvested energy from the FL server and utilizes this energy to train its local model based on its dataset within τ_t^{cp}.
- Step 3: The UEs transmit their local parameters $\omega_{n,t+1}$ to the FL server via uplink MC-NOMA with τ_t^{cm}.
- Step 4: Upon receiving the local models, the FL server aggregates all the local models to construct a global model within τ_t^{agg}. Subsequently, the server broadcasts the global parameter ω_{t+1} to all UEs, initiating the next time slot.

This process is repeated until the global model converges.

2.2.1. *Local computation*

Let $f_{n,t}$ represent the computational capacity of UE n at time slot t, which is measured by the number of CPU cycles per second. Then, the computation time required by UE n for data processing is

$$\tau_{n,t}^{cp} = \frac{\delta_{n,t}^{Local} C_{n,t} D_{n,t}}{f_{n,t}}, \tag{3}$$

where $C_{n,t}$, $D_{n,t}$, and $\delta_{n,t}^{Local}$ are the number of CPU cycles required for computing one bit of data, the data samples, and the number of local iterations of UE n, respectively.

2.2.2. *Transmission*

Regarding MC-NOMA in the uplink, $\mathcal{N}_{m,t}$ and $|\mathcal{N}_{m,t}|$ denote the set of UEs and the number of UEs assigned to the m^{th} sub-channel, respectively. Meanwhile, each UE can only associate with one sub-channel. Each sub-channel can accommodate multiple UEs simultaneously. Without loss of generality, UEs are assigned to one sub-channel based on their respective channel conditions, i.e. $\mathcal{N}_{m,t} \cap \mathcal{N}_{m',t} = \emptyset$ where $m' \in \mathcal{M}$ and $m' \neq m$. In this work, we omitted the optimization of selecting user-subcarrier association and rather applied a fixed rule. We will consider the optimization of association variables in our future work.

At the BS, for each sub-channel, the FL server uses successive interference cancellation (SIC) to decode the uploaded local model parameters of the UEs. In this sub-section, to avoid possible confusion, we use k-th index to indicate the users within the group of users associated to m-th sub-channel. Let $\pi_{k,m,t}$ be the decoding order of UE $k \in \mathcal{N}_{m,t}$. We followed a widely adopted assumption where the decoding order of the UEs at the BS is determined using the ascending order of their channel gains. The achievable rate of UE k over sub-channel m at time slot t can be expressed as [17]:

$$r_{k,m,t} = B_t \log_2 \left(1 + \frac{p_{k,m,t} g_{k,m,t}}{\sum_{k'=1}^{k-1} p_{k',m,t} g_{k',m,t} + B_t \sigma^2} \right), k, k' \in \mathcal{N}_{m,t}, \tag{4}$$

where B_t, $p_{k,m,t}$, $g_{k,m,t}$, and σ^2 are the bandwidth allocated to each sub-channel, the transmit power of UE k, the channel gain between BS and UE k, and the power spectral density of the Gaussian noise at time slot t, respectively. Also, $\sum_{k'=1}^{k-1} p_{k',m,t} g_{k',m,t}$ is the intra-interference that user k suffers from other users k' sharing the same sub-channel with decoding order less than $\pi_{k,m,t}$. We assume that all UEs have the same model size Z, which must be transmitted to the FL server within τ_t [15]. Then, the transmission time of UE k is

$$\tau^{cm}_{k,m,t} = \frac{Z}{r_{k,m,t}}. \tag{5}$$

2.2.3. Energy consumption Model

According to [11], the computation energy consumption of UE n at time slot t is expressed as

$$E^{cp}_{n,t} = \zeta \delta^{Local}_{n,t} C_{n,t} D_{n,t} f^2_{n,t}, \tag{6}$$

where ζ is a coefficient that depends on the hardware and chip architecture [11]. Meanwhile, the communication energy consumption of UE n is defined as

$$E^{cm}_{n,t} = p_{n,t} \tau^{cm}_{n,t}, \tag{7}$$

where $p_{n,t}$ and $\tau^{cm}_{n,t}$ are the transmit power and the transmission time of UE n, respectively. Finally, the total energy consumption of UE n can be expressed as

$$E_{n,t} = E^{cp}_{n,t} + E^{cm}_{n,t}. \tag{8}$$

2.3. Problem Formulation

We aimed to develop a FL framework that minimizes the total energy consumption of all UEs under a latency constraint and various resource constraints. The minimization problem can be expressed as

$$\min_{\mathbf{p},\mathbf{f}} \quad \sum_{t=1}^{T} \left(\sum_{n=1}^{N} E_{n,t} \right) \tag{9}$$

$$\text{s.t.} \quad 0 \le p_{n,t} \le p^{max}_n, \forall n \in \mathcal{N}, \forall t \in \mathcal{T} \tag{10}$$

$$f^{min}_n \le f_{n,t} \le f^{max}_n, \forall n \in \mathcal{N}, \forall t \in \mathcal{T} \tag{11}$$

$$\tau^{down}_{n,t} + \tau^{cp}_{n,t} + \tau^{cm}_{n,t} \le \tau_t, \forall t \in \mathcal{T} \tag{12}$$

$$E^{cp}_{n,t} + E^{cm}_{n,t} \le \psi_{n,t} + E^{ha}_{n,t}, \forall n \in \mathcal{N}, \forall t \in \mathcal{T}, \tag{13}$$

where $\mathbf{p} = [\mathbf{p}_1, \cdots, \mathbf{p}_T] \in \mathbb{R}^{T \times N}$ with $\mathbf{p}_t = [p_{1,t}, \cdots, p_{N,t}] \in \mathbb{R}^{1 \times N}$ and $\mathbf{f} = [\mathbf{f}_1, \cdots, \mathbf{f}_T] \in \mathbb{R}^{T \times N}$ with $\mathbf{f}_t = [f_{1,t}, \cdots, f_{N,t}] \in \mathbb{R}^{1 \times N}$. Constraint (10) denotes the transmission power allocation for the UEs, where p^{max}_n denotes the maximum transmit power of UE n. Constraint (11) denotes the CPU frequency allocation for the UEs, where f^{min}_n and f^{max}_n denote minimum and maximum frequency, respectively. Constraint (12) ensures that each global round should be completed within the time frame τ_t. Finally, constraint (13) ensures that the sum of the residual battery and the harvested energy for each UE should always be greater than or equal to the total energy consumption needed for local training and uploading.

3. Proposed Solution

Due to the real nature of control variables and the coupling relations among the variables as in (4), the original problem is a non-convex and non-deterministic polynomial time (NP)-hard problem. We reformulated the optimization problem as an MDP and developed a DDPG-based reinforcement learning algorithm to effectively solve the MDP.

3.1. Markov Decision Process

To find a dynamic solution for (9), we transformed the problem into an MDP model. This model is defined as a tuple of $(\mathcal{S}, \mathcal{A}, \mathcal{R})$, where $\mathcal{S}$ is the set that represents the actual state of the environment; $\mathcal{A}$ is the action space set; and $\mathcal{R}$: $\mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward space set that the agent receives from the environment when it takes an action a at state s. Next, $\gamma \in [0, 1)$ is denoted as the discount factor. The objective of the agent is to learn a mapping function, which is termed a policy, μ: $\mathcal{S} \to \mathcal{A}$ that maximizes the expected long-term discounted reward. We clarify the definitions of the notions of state space, action space, reward, and penalty as follows:

1. State space: At each time slot t, the agent observes the network environment. The system state is determined at the beginning of time slot t as $s_t = \{\psi_{n,t}, \tau^{cp}_{n,t-1}, \tau^{cm}_{n,t-1}\}_{n \in \mathcal{N}}$. This includes the current energy level, the computation time, and the transmission time from the previous time slot for each UE, respectively. By using the historical information from previous steps, the learning behavior would be more stable.

2. Action space: The action taken in response to the current state s_t is defined as $a_t = \{p_{n,t}, f_{n,t}\}_{n \in \mathcal{N}}$. This includes the transmit power and the CPU frequency for each UE, respectively. The action space $\mathcal{A}$ encompasses all possible continuous values of the relevant variables.

3. Reward and penalty: We define an energy consumption metric (ECM) for UE n at the end of time slot t as

$$R_{n,t} = -E_{n,t}, \tag{14}$$

where $E_{n,t} = E^{cp}_{n,t} + E^{cm}_{n,t}$. Penalties play a pivotal role in influencing the agent's behavior and can aid in achieving specific objectives or constraints during the learning process. We include two distinct penalties: an energy penalty and a time penalty.

The energy penalty of each time slot is defined based on the constraint (13), which is $pe^{energy}_t = \sum_{n \in \mathcal{N}} pe^{energy}_{n,t}$. Thus, The energy penalty of UE n is defined as

$$pe^{energy}_{n,t} = \max(E^{cp}_{n,t} + E^{cm}_{n,t} - (\psi_{n,t} + E^{ha}_{n,t}), 0). \tag{15}$$

Additionally, the system is stable when the batteries of all UEs have enough energy. We denote a battery variable $\Psi_{n,t}$ that describes the energy state of UE n, where

$$\Psi_{n,t} = \begin{cases} 1, & \text{UE satisfies (13)}, \\ 0, & \text{otherwise}. \end{cases} \tag{16}$$

Next, the time penalty of each time slot is defined based on constraint (12): $pe_t^{time} = \sum_{n\in\mathcal{N}} pe_{n,t}^{time}$, where

$$pe_{n,t}^{time} = \begin{cases} 0, & \text{UE satisfies (12)}, \\ 0.001, & \text{otherwise.} \end{cases} \tag{17}$$

The objective is to maximize the ECM for all UEs under a latency constraint and various resource constraints. The agent receives an instant reward ($r : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$) after making decision a_t when in state s_t. This is defined as

$$r_t = \sum_{n=1}^{\mathcal{N}} \left(R_{n,t} - pe_{n,t}^{energy} - pe_{n,t}^{time} \right). \tag{18}$$

3.2. DDPG-based Energy Efficient FL Framework

We have developed a DDPG-based reinforcement learning algorithm to solve our MDP problem. The deep reinforcement learning (DRL) agent is deployed on the FL server to learn the optimal decision policy. Our DDPG algorithm utilizes an extension of the actor-critic approach, where deep neural networks (DNNs) are employed to approximate policy and value functions [18]. This algorithm is adept at managing decision-making processes that entail extensive state and continuous action spaces. The extensive state space and action space contained in this work are due to the many variables and large number of UEs. We introduce a DDPG-based energy efficient FL algorithm, outlined in 1, as an effective solution to address our optimization problem. The key components of the DDPG algorithm include utilizing two primary DNNs: a policy network and a critic network. Specifically, our DDPG approach parameterizes the policy $\mu(s|\theta^{\mu})$ with action given a state as $a = \mu(s|\theta^{\mu})$ and the critic network $Q(s, a|\theta^{Q})$ with the weight parameter sets θ^{μ} and θ^{Q}, respectively. We aim to determine the optimal policy, defined as

$$\mu^*(s|\theta^{\mu}) = \arg\max_{a\in\mathcal{A}} \left\{ Q\left(s_t, a_t|\theta^{Q}\right) \right\}, \tag{19}$$

where $Q\left(s_t, a_t|\theta^{Q}\right)$ is the value of chosen action a_t at state s_t. Thus, we find the optimal joint action a^* that maximizes the expected cumulative Q-value $Q^*\left(s_t, a_t|\theta^{Q}\right)$. We achieve this by solving the following Bellman equation:

$$Q^*\left(s_t, a_t|\theta^{Q}\right) = \mathbb{E}\left[\max_{a\in\mathcal{A}} \left[r_t + \gamma Q^*\left(s_{t+1}, a_{t+1}|\theta^{Q}\right) \right] \right]. \tag{20}$$

At time slot t, after observing the system state s_t, the agent applies action a_t to the environment. Subsequently, the agent receives an instant reward r_t at the end of the time slot. The system then transits to state s_{t+1}. Each experience tuple (s_t, a_t, r_t, s_{t+1}) is stored in a replay buffer $\mathcal{H}$ with a limited size. Additionally, DDPG incorporates a target actor network, denoted as $\mu'(s|\theta^{\mu})$, and a target critic network, denoted as $Q'(s, a|\theta^{Q})$, to improve the stability of network training.

The agent is equipped with an actor network, responsible for learning the policy. This network takes the current state as input and generates the corresponding action as an output. The actor network is updated using a policy gradient, which is defined as:

$$\nabla_{\theta^{\mu}} J(\theta^{\mu}) \approx \mathbb{E}_{H\in\mathcal{H}} \left[\nabla_{\theta^{\mu}} \mu(s_t|\theta^{\mu}) \nabla_{a_t} Q(s_t, a_t|\theta^{Q}) | a_t = \mu(s_t|\theta^{\mu}) \right], \tag{21}$$

where H is the batch sample data.

At the same time, the critic network takes both the state and the action as inputs and produces the Q-value as its output. The loss function of the critic network is defined as:

$$L(\theta^{\mathbf{Q}}) = \mathbb{E}_{H\in\mathcal{H}} \left[(y_t - Q(s_t, a_t|\theta^{Q}))^2 \right], \tag{22}$$

where y_t is expressed as

$$y_t = r_t + \gamma Q'(s_{t+1}, \mu'(s_{t+1}|\theta^{\mu})|\theta^{Q}). \tag{23}$$

To maintain the stability of y_t during the training process, the parameters of the target networks are slowly updated with a small coefficient, $\xi \in [0, 1]$, as $\theta^{\mu'} = \xi\theta^{\mu} + (1 - \xi)\theta^{\mu'}$ and $\theta^{\mathbf{Q}'} = \xi\theta^{\mathbf{Q}} + (1 - \xi)\theta^{\mathbf{Q}'}$.

Next, to improve the exploration during the training process, the continuous policy is modified as follows:

$$a_t^* = \mu\left(s_t|\theta^{\mu}\right) + \chi_0, \tag{24}$$

where χ_0 is added noise to ensure adequate exploration of the current policy. We use the Ornstein Uhlenbeck process to generate this noise [19]. Next, we employ sigmoid activation to scale the actor's output within the range of [0,1]. Algorithm 1 outlines our DDPG-based energy efficient FL framework.

Algorithm 1 DDPG-based Energy Efficient FL Algorithm

Initialize hyperparameters: γ, the critic learning rate lra, the actor learning rate lrc, H, $\mathcal{H}$, χ.
Initialize the actor network, $\mu(s|\theta^{\mu})$, and the critic network, $Q(s, a|\theta^{Q})$ with weights θ^{μ} and θ^{Q}.
Initialize the target actor network $\mu'(s|\theta^{\mu'})$ and the target critic network $Q'(s, a|\theta^{Q'})$ with weights $\theta^{\mu'} \leftarrow \theta^{\mu}$ and $\theta^{Q'} \leftarrow \theta^{Q}$.
Initialize a replay buffer $\mathcal{H}$.
for each episode $e = 1, 2, ..., e_{max}$ **do**
  Initialize system state s_1 and random action nose χ_0.
  for $t = 1, 2, , , , , T$ **do**
    Initialize the terminal variable Ψ_t.
    Observe s_t.
    Execute overall action a_t^* according to (24).
    Observe reward r_t with penalties (15), (17) and next state s_{t+1}.
    Store experience in the buffer $\mathcal{H}$ and uniform sample a batch of H to train the DNNs.
    Update the actor network and the critic network.
    Update the target actor network and the target critic network.
    Update the battery variables $\Psi_{n,t}$ according to (16).
    Update Ψ_t according to variables $\Psi_{n,t}$.
    if $\Psi_t = 0$ **do**
      break.
  end for
end for

In addition, the learning rate in the DDPG algorithm controls the step size of the parameter updates during the optimization process. Particularly, it influences how quickly or slowly the algorithm converges to a solution and the stability and robustness of the training process. In 'Simulation Results' section, we showed the effects of the learning rate on the convergence of the proposed algorithm in Fig. 3.

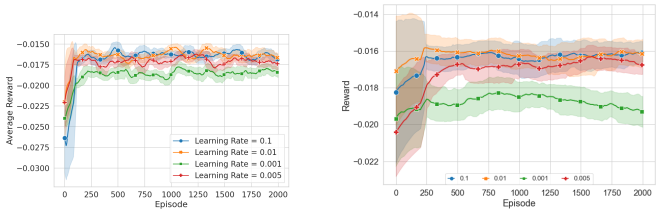
3.3. Complexity Analysis

During the training phase, the agent employs its actor and critic networks to conduct forward propagation and backpropagation for weight updates. The time complexity of system is evaluated as $O(n)$ with n as the size of the matrices. Besides, DDPG algorithm uses experience replay, a technique wherein experiences are stored in a replay buffer and sampled randomly during the training process. The time complexity of sampling from the replay buffer is $O(1)$. Consequently, the time complexity for a single episode during the training phase can be expressed as $O\Big(H(IN + (D-2)N^2 + JN)\Big)$, where H represents the batch sample data, $D \geq 2$ is the number of layer, N is the hidden layer size, and I, J denote the dimensions of the input and output layers, respectively. In the specific context of the problem, $I = 5M$, where M is the number of UEs, and 5 is the sum of the observation and action space dimensions. Moreover, J is the output Q-value, equal to 1. In the execution phase, only forward propagation is necessary for the actor network, resulting in a computational complexity of $O\Big((D-2)N^2 + 3N\Big)$. Notably, while the computational complexity is affected by the number of UEs, it demonstrates that it remains unaffected by the number of sub-channels during the training phase.

4. Simulation Results

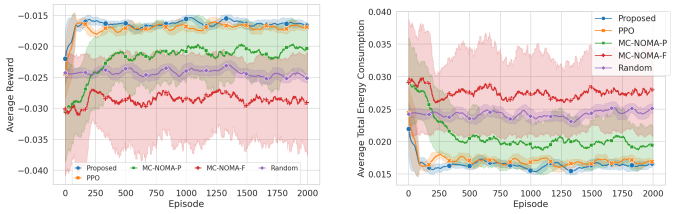
To evaluate our algorithm, we deployed a network that consisted of one BS and 10 UEs. Here, the UEs are randomly distributed within a circular disc with a radius of 50m from the BS. The system has three sub-channels, and each sub-channel has a 1 MHz bandwidth. For our energy harvest model, the BS transmit power is $P = 40$ W, the energy harvesting efficiency is $\eta_0 = 0.8$ [5], the power splitting ratio is $\kappa = 0.001$ [5], and the model broadcast time is $\tau_t^{down} = 0.01$ s. Regarding local UE computation, the maximum CPU frequency for each UE is 1 GHz [15] and the coefficient for the chip is $\zeta = 10^{-28}$ [11]. Finally, the number of local iterations, the amount of local data, and the workload are randomly selected from the ranges [1,4] [12], [10,20] Mb [15] , and [10,15] cycles per bit [15], respectively. For wireless communication, the maximum power for each UE is 10 dB [11], the additional attenuation factor $\iota = 0.2$ [14], and the data size $Z = 100$ Kb [12]. Regarding DDPG hyper-parameters, the discount factor γ, steps per episode, the critic learning rate lrc, the actor learning actor lra, the small coefficient ξ, the batch size H, and Buffer capacity $\mathcal{H}$ are 0.99 [5], 150, 1e-2. 1e-3, 1e-3, 1024, and 75,000, respectively. Finally, the duration of each time slot is defined as $\tau_t = 0.5$ s, the power spectral density of the Gaussian noise σ^2 is -174 dBm/Hz [11], the reference channel gain at $d_0 = 1$ m is -30 dB [12], and the maximum battery capacity of a UE is assumed as 100 mJ.

In these simulations, we compared our proposed algorithm with competing learning-based schemes: 1) Proximal Policy Optimization (PPO) scheme: this approach employs the actor-critic framework to train a stochastic policy in an on-policy manner; 2) user power transmit fixed scheme (MC-NOMA-P): in this scheme, the transmit power of all UEs is fixed, while we optimize the CPU resource allocation with an MC-NOMA uplink; 3) CPU frequency fixed scheme (MC-NOMA-F): in this scheme, the CPU frequency of all UEs is fixed, while we optimize the power allocation with an MC-NOMA uplink; and 4) random scheme: the agent's actions in this scheme are selected randomly.



(a) LoS. (b) NLoS.

Figure 3. Convergence of the proposed algorithm under LoS and NLoS assumption.

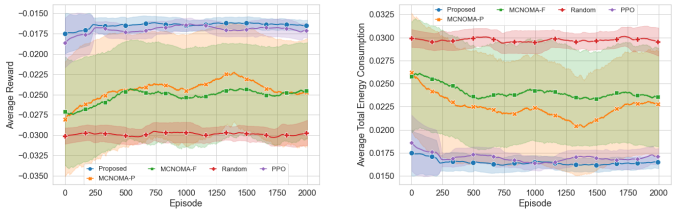


(a) System reward. (b) Energy consumption.

Figure 4. Comparison of system reward and energy consumption under LoS assumption.

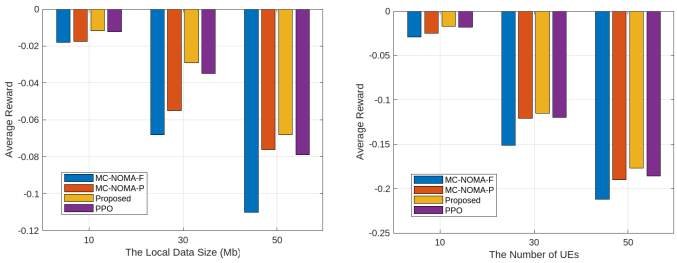
Fig. 3 illustrates the convergence behavior of our proposed algorithm for both LoS and NLoS assumptions under different learning rates. We can see that the average rewards initially increase for all cases, but they ultimately converge in a stable manner. It should be noted that the average rewards with learning rates of 0.1 and 0.01 are quite similar. However, a smaller learning rate can lead to slower convergence. Consequently, the optimal reward occurs with a learning rate of 0.01. We adopted a learning rate of 0.01 for our simulations.

Fig. 4a and Fig. 5a compare the system rewards for different schemes. The proposed algorithm outperforms the other methods. In comparison to the MC-NOMA-P scheme and the MC-NOMA-F scheme, the proposed algorithm jointly optimizes the transmit power and CPU frequency, while the transmit power is fixed for MC-NOMA-P, and the UE CPU frequency is fixed for



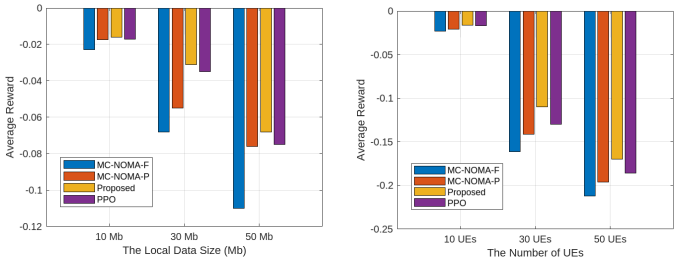
(a) System reward. (b) Energy consumption.

Figure 5. Comparison of system reward and energy consumption under NLoS assumption.



(a) Effect of the local data size. (b) Effect of the number of UEs.

Figure 6. Effect of the local data size and the number of UEs under LoS assumption.



(a) Effect of the local data size. (b) Effect of the number of UEs.

Figure 7. Effect of the local data size and the number of UEs under NLoS assumption.

MC-NOMA-F. Next, when compared to the PPO scheme, the proposed algorithm provides slightly better performance. The DDPG algorithm appears to be more suitable in this scheme due to uniform random sampling experiences from experience buffers instead of sampling the transitions of experience as in PPO. This would break the temporal relations between the consecutive experiences and help DDPG become easier to escape local minima than PPO. Fig. 4b and Fig. 5b compare the average energy consumption. The results indicate that the energy consumptions of the proposed algorithm and the PPO scheme are lower compared to other schemes. Additionally, the energy efficiency of the proposed algorithm is slightly better than the PPO scheme.

Fig. 6a and Fig. 7a show the impact of local data size on the performance. As the local data size increases from 10 Mb to 50 Mb, the average reward for all schemes decreases over the total duration. This decrease can be attributed to the fact that the UEs require more energy to train their local models given the larger local data sizes. Consequently, there is a significant increase in energy consumption that leads to decreased performance. The proposed solution consistently outperforms the competing schemes over different local data size.

Fig. 6b and Fig. 7b show the impact of the number of UEs on performance. As the number of UEs increases, the average reward for all schemes decreases. This decline can be attributed to the growing interference in each sub-channel as the number of UEs increases. As a result, the UEs require more energy to transmit their local models to the FL server. This causes a decrease in performance across all schemes. However, it is noteworthy that the proposed solution consistently outperforms the competing schemes over different number of UEs.

5. Conclusion

This paper outlined an energy efficient FL system that includes SWIPT and MC-NOMA. First, we formulated a joint power and CPU resource allocation problem to minimize energy consumption under a latency constraint. Next, we transformed this joint optimization problem into an MDP formulation, and proposed a DDPG-based reinforcement learning algorithm. Finally, the simulations confirmed that the proposed algorithm stably converges under different learning rates and provides enhanced energy efficiency compared with competing schemes. As a future work, the communication and energy efficient of federated learning system under satellite and IRS-assisted network will be investigated.

Acknowledgments

This work was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2022R1A4A5034130) and in part by Chung-Ang University Young Scientist Scholarship in 2022.

References

[1] G. Zhu, D. Liu, Y. Du, C. You, J. Zhang, and K. Huang, "Toward an intelligent edge: Wireless communication meets machine learning," *IEEE communications magazine*, vol. 58, no. 1, pp. 19–25, 2020.

[2] G. Zhu, Y. Wang, and K. Huang, "Broadband analog aggregation for low-latency federated edge learning (extended version)," *arXiv preprint arXiv:1812.11494*, 2018.

[3] Q. Zeng, Y. Du, K. Huang, and K. K. Leung, "Energy-efficient radio resource allocation for federated edge learning," in *2020 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 2020, pp. 1–6.

[4] N. H. Tran, W. Bao, A. Zomaya, M. N. Nguyen, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *IEEE INFOCOM 2019-IEEE conference on computer communications*. IEEE, 2019, pp. 1387–1395.

[5] Y. Wu, Y. Song, T. Wang, M. Dai, and T. Q. Quek, "Simultaneous wireless information and power transfer assisted federated learning via nonorthogonal multiple access," *IEEE Transactions on Green Communications and Networking*, vol. 6, no. 3, pp. 1846–1861, 2022.

[6] Y. Li, Y. Wu, Y. Song, L. Qian, and W. Jia, "Dynamic user-scheduling and power allocation for swipt aided federated learning: A deep learning approach," *IEEE Transactions on Mobile Computing*, 2022.

[7] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 1, pp. 269–283, 2020.

[8] A. Abutuleb, S. Sorour, and H. S. Hassanein, "Joint task and resource allocation for mobile edge learning," in *GLOBECOM 2020-2020 IEEE Global Communications Conference*. IEEE, 2020, pp. 1–6.

[9] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[10] P. S. Bouzinis, P. D. Diamantoulakis, and G. K. Karagiannidis, "Wireless federated learning (wfl) for 6g networks—part ii: The compute-then-transmit noma paradigm," *IEEE Communications Letters*, vol. 26, no. 1, pp. 8–12, 2021.

[11] Z. Yang, M. Chen, W. Saad, C. S. Hong, and M. Shikh-Bahaei, "Energy efficient federated learning over wireless communication networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 3, pp. 1935–1949, 2020.

[12] Q.-V. Pham, M. Zeng, R. Ruby, T. Huynh-The, and W.-J. Hwang, "Uav communications for sustainable federated learning," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 4, pp. 3944–3948, 2021.

[13] Q.-V. Pham, M. Le, T. Huynh-The, Z. Han, and W.-J. Hwang, "Energy-efficient federated learning over uav-enabled wireless powered communications," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 5, pp. 4977–4990, 2022.

[14] Y. Zeng, J. Xu, and R. Zhang, "Energy minimization for wireless communication with rotary-wing uav," *IEEE Transactions on Wireless Communications*, vol. 18, no. 4, pp. 2329–2345, 2019.

[15] Q. V. Do, Q.-V. Pham, and W.-J. Hwang, "Deep reinforcement learning for energy-efficient federated learning in uav-enabled wireless powered networks," *IEEE Communications Letters*, vol. 26, no. 1, pp. 99–103, 2021.

[16] Z. Yang, M. Chen, W. Saad, C. S. Hong, M. Shikh-Bahaei, H. V. Poor, and S. Cui, "Delay minimization for federated learning over wireless communication networks," *arXiv preprint arXiv:2007.03462*, 2020.

[17] W. Li, T. Lv, Y. Cao, W. Ni, and M. Peng, "Multi-carrier noma-empowered wireless federated learning with optimal power and bandwidth allocation," *IEEE Transactions on Wireless Communications*, 2023.

[18] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[19] D. S. Lakew, N.-N. Dao, S. Cho *et al.*, "Adaptive partial offloading and resource harmonization in wireless edge computing-assisted ioe networks," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 5, pp. 3028–3044, 2022.