Robust DNN Partitioning and Task Offloading in Multi-UAV-Assisted MEC Against UAV Failures

Shuman Meng[1], **Bin Li**[1]

[1]Nanjing University of Information Science & Technology, Nanjing, China
{202412200705, bin.li}@nuist.edu.cn

Abstract

With the rapid development of artificial intelligence, deep neural network (DNN)-based tasks have become increasingly prevalent, yet their intensive computation demands often exceed the capability of a single mobile device. Unmanned aerial vehicle (UAV)-assisted mobile edge computing (MEC) is able to provide the flexibility of task offloading through aerial edge servers. However, UAV may fail owing to battery depletion, hardware faults, or harsh environments, thus threatening task continuity and service reliability. To enhance fault tolerance and ensure continuous DNN inference, we develop a robust multi-UAV-assisted MEC framework that integrates adaptive DNN partitioning and failure-aware task migration mechanisms. The proposed framework aims to minimize the weighted energy consumption by jointly optimizing user association, task migration, DNN partitioning, UAV trajectories, computing resource allocation, and transmission power, while imposing the practical constraints on DNN-based tasks. To solve this problem, we propose a soft actor-critic with prioritized experience replay algorithm, incorporating convex optimization to determine the optimal transmission power. Finally, simulation results demonstrate that the proposed framework outperforms benchmark schemes in both performance and robustness, validating its effectiveness in dynamic and failure-prone environments.

Introduction

Driven by recent breakthroughs in artificial intelligence, deep neural networks (DNNs) have become crucial in intelligent mobile applications such as autonomous driving, augmented reality, and real-time monitoring (Zhang et al. 2019). To address the intensive computational demands of these applications, DNN partitioning has emerged as an effective technique that transfers the output data of the last processed layer from local devices instead of raw data, thereby reducing computation latency and mitigating privacy leakage risks. For instance, Yan *et al.* (Yan et al. 2025) designed the DNN partitioning and task offloading strategies in the vehicular edge-cloud collaboration environment to minimize processing delay and energy consumption. In (Wen et al. 2024), a multi-device edge inference system with DNN splitting was investigated to enable the low-latency edge intelligent services. However, the high computational demand of DNN inference still exceed the limited processing capability of resource-constrained mobile devices.

To bridge this gap between computation demand and device capability, mobile edge computing (MEC) has emerged as a promising paradigm to overcome this limitation by offloading computation-intensive tasks from resource-constrained devices to nearby edge servers, thus reducing processing latency and energy consumption (Mach and Becvar 2017). Recently, unmanned aerial vehicles (UAVs)-assisted MEC has attracted growing attention due to their flexible deployment, wide coverage, and rapid response capability. Compared with static edge servers, UAVs provide greater adaptability in dynamic environments. For example, He *et al.* (He et al. 2023) designed a three-dimensional (3D) dynamic multi-UAV-assisted MEC system which optimizes communication, computation, and flight energy consumption among UAVs based on fairness. In (Li et al. 2024), Li *et al.* proposed a multi-UAV MEC framework that uses knowledge-assisted reinforcement learning approach to optimize task offloading and UAV trajectory planning, aiming to improve processing success rates and fairness. Moreover, multi-UAV-assisted MEC frameworks significantly enhance system capacity through the coordination of multiple UAVs, which enables efficient task offloading and distributed DNN partitioning.

Despite their advantages, multi-UAV-assisted MEC systems are vulnerable to unexpected UAV failures due to limited battery life, hardware faults, or harsh environments. Such failures can interrupt computing and communication tasks, severely degrading DNN inference and even causing service outages. This is unacceptable for latency-sensitive applications like autonomous driving, disaster response, and real-time surveillance. Thus, robustness against UAV failures is not optional but essential for practical deployment. Ensuring task continuity and reliable service delivery under failures is critical to futher leverage the potential of UAV-assisted edge intelligence. To ensure the robustness of DNN inference against UAV failures, we investigate a multi-UAV-assisted MEC architecture. The contributions of this work are outlined as follows:

- To enhance fault tolerance and ensure continuous DNN inference in dynamic and failure-prone environments, we propose a robust multi-UAV-assisted MEC framework that integrates adaptive DNN partitioning and failure-

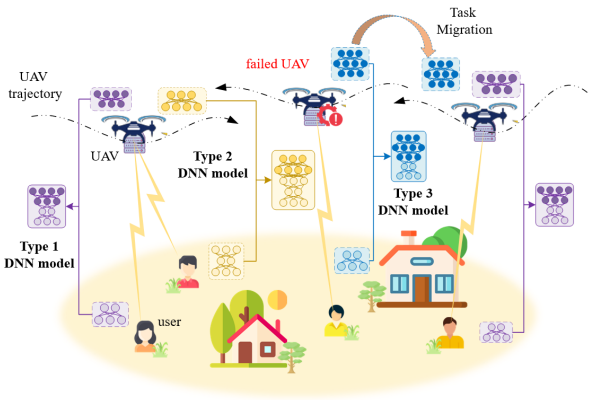

Figure 1: The system model of multi-UAV-assisted MEC system.

aware task migration mechanisms. Taking into account the latency limitations of DNN-based tasks, we establish a formulation for the problem of minimizing weighted energy consumption by collaboratively designing user association, task migration, DNN partitioning, UAV trajectories, computing resource allocation, and transmission power.

- To tackle the high complexity of the real-time problem, we model it as a Markov decision process (MDP) and integrated soft actor-critic with prioritized experience replay (SAC-PER) with convex optimization techniques for transmitting power, aiming to boost the convergence of its learning process.
- Simulation results demonstrate that the SAC-PER outperforms representative DRL algorithms in both convergence and performance, while ensuring the robustness. Additionally, the proposed partitioning scheme significantly reduces energy consumption compared to other schemes.

System Model And Problem Formulation

A multi-UAV-assisted MEC system where J UAVs equipped with MEC servers serve I mobile users (MUs) is considered, as shown in Figure 1. The set of UAVs and MUs are denoted by $\mathcal{J} = \{1, ..., j, ..., J\}$ and set $\mathcal{I} = \{1, ..., i...., I\}$, respectively. To make the problem more tractable and scalable, the flight period N is divided into T time slots with equal size δ, i.e., $N = T\delta$. The set of time slots is denoted as $\mathcal{T} = \{1, ..., t, ..., T\}$. We assume that each MU i generates a DNN-based task b_i at the beginning of time slot t. It is worth noting that both MUs and UAVs in this MEC scenario use pre-trained DNN models to compute their tasks (Gao et al. 2023). Since the limited computation resources of MUs, their DNN-based tasks need to be offloaded to the associated UAV. We assume that each MU is associated with only a single UAV per time slot. Let $\alpha_{i,j}(t) \in \{0, 1\}$ denote the MU association indicator, where $\alpha_{i,j}(t) = 1$ if MU i is associated with UAV j; otherwise $\alpha_{i,j}(t) = 0$. Thus, we have

$$\sum_{j=1}^{J} \alpha_{i,j}(t) = 1, \forall i \in \mathcal{I}, t \in \mathcal{T}. \tag{1}$$

Subsequently, we adopt the UAV's propulsion energy model introduced by (Zeng, Xu, and Zhang 2019), the flying energy of UAV j is then expressed as follows:

$$e_j^{\text{fly}}(t) = \left[\frac{1}{2} d_0 \rho_0 g A \left\|\mathbf{v}_j(t)\right\|^3 + P_1 \left(1 + \frac{3\left\|\mathbf{v}_j(t)\right\|^3}{U_{\text{tip}}^2}\right) + P_2 \left(\sqrt{1 + \frac{\left\|\mathbf{v}_j(t)\right\|^4}{4v_0^4}} - \frac{\left\|\mathbf{v}_j(t)\right\|^2}{2v_0^2}\right)^{\frac{1}{2}}\right] \delta, \tag{2}$$

where P_1 and P_2 are the blade profile power and the induced power, respectively. v_0 signifies the average rotor s peed, and U_{tip} represents the rotor blade's tip speed. Additionally, d_0, ρ_0, g, and A are the fuselage drag ratio, air density, rotor solidity and rotor disc area, respectively.

Mobility Model

The 3D coordinate is applied to denote the trajectories of the UAVs and MUs. For time slot t, the location of UAV j and MU i can be expressed as $\mathbf{q}_j(t) = (x_j(t), y_j(t), H_j(t))^T$ and $\mathbf{w}_i(t) = (x_i(t), y_i(t), 0)^T$, respectively. Note that all MUs and UAVs are initially located randomly, and all MUs navigate following the Gaussian-Markov model (Liu et al. 2020). Within time slot t, the direction $\theta_i(t)$ and the velocity $v_i(t)$ of MU i are determined by the subsequent equations

$$\theta_i(t) = \mu_1 \theta_i(t-1) + (1-\mu_1)\bar{\theta} + \sqrt{1-\mu_1^2}\Psi_i, \tag{3a}$$

$$v_i(t) = \mu_2 v_i(t-1) + (1-\mu_2)\bar{v} + \sqrt{1-\mu_2^2}\Phi_i, \tag{3b}$$

where $\bar{\theta}$ and $\bar{v}$ represent the average direction and velocity of all MUs, respectively. $0 \leq \mu_1, \mu_2 \leq 1$ are utilized to adjust the influence of the previous state. Additionally, Ψ_i and Φ_i follow two independent Gaussian distributions with different mean-variance pairs $(\bar{\xi}_{\theta_i}, \varsigma_{\theta_i}^2)$ and $(\bar{\xi}_{v_i}, \varsigma_{v_i}^2)$ for MU i. According to (3a) and (3b), the location of MU i can be updated as

$$x_i(t) = x_i(t-1) + v_i(t-1)\cos(\theta_i(t-1))\delta, \tag{4a}$$

$$y_i(t) = y_i(t-1) + v_i(t-1)\sin(\theta_i(t-1))\delta. \tag{4b}$$

In addition, to avoid the mutual collision, a safe distance d_{safe} should be maintained between UAVs, thus we have

$$\left\|\mathbf{q}_j(t) - \mathbf{q}_k(t)\right\|^2 \geq d_{\text{safe}}^2, \forall j, k \in \mathcal{J}, t \in \mathcal{T}, j \neq k. \tag{5}$$

To ensure authenticity of UAVs flight trajectory, we also define the maximum velocity $v_{\max}$ and maximum acceleration $a_{\max}$ of each UAV. Therefore, UAVs are required to satisfy the following constraints within each time slot

$$\mathbf{q}_j(t+1) = \mathbf{q}_j(t) + \mathbf{v}_j(t)\delta + \frac{1}{2}\mathbf{a}_j(t)\delta^2, \forall j \in \mathcal{J}, t \in \mathcal{T}, \tag{6}$$

$$\left\|\mathbf{v}_j(t)\right\| \leq v_{\max}, \left\|\mathbf{a}_j(t)\right\| \leq a_{\max}, \forall j \in \mathcal{J}, t \in \mathcal{T}. \tag{7}$$

DNN Partitioning Model

It is supposed that there are K types of DNNs, which can be defined as $\mathcal{K} = \{1, ..., k, ..., K\}$. The k-th model has L_k layers, denoted by $\mathcal{L}_k = \{1, ..., l_k, ..., L_k\}$, which contains convolution (Conv) layers $Conv_k$ and fully connected (FC) layers FC_k, satisfying $FC_k \cup Conv_k = \mathcal{L}_k$ and $FC_k \cap Conv_k = \emptyset$.

For the Conv layer $l_k \in FC_k$, it is characterized with a tuple $(H_{l_k}, W_{l_k}, C_{l_k}^{\text{in}}, C_{l_k}^{\text{out}}, ker_{l_k})$, where $(H_{l_k}, W_{l_k}, C_{l_k}^{\text{in}})$ represents the input height, width and channel, and $(C_{l_k}^{\text{out}}, ker_{l_k})$ represents the output channel, and kernel size of layer l_k, respectively. For the FC layer $l_k \in Conv_k$, the configuration tuple $(U_{l_k}^{\text{in}}, U_{l_k}^{\text{out}})$ represents the sizes of the one-dimensional input and output.

Let the set of MUs with the k-th type DNN model be denoted by $\mathcal{I}_k$, where $\mathcal{I} = \bigcup_{k \in \mathcal{K}} \mathcal{I}_k$. We represent the MU i's DNN partitioning decision during time slot t by introducing an integer variable $\varphi_i(t) \in \{1, ..., L_k\}, i \in \mathcal{I}_k$. Especially, when $\varphi_i(t) = 1$, the entire DNN-based task b_i is offloaded onto the UAV for computation; when $2 \leq \varphi_i(t) \leq L_k$, the first part is processed locally and the second part is processed on the UAV.

We use the number of floating-point operations (FLOPs) to model the computation workload of processing input data in each layer (Liu et al. 2025), which can be expressed as

$$B_{l_k} = \begin{cases} 2H_{l_k}W_{l_k}\left(C_{l_k}^{\text{in}} ker_{l_k}^2\right) C_{l_k}^{\text{out}}, & l_k \in FC_k, \\ 2\left(U_{l_k}^{\text{in}} - 1\right) U_{l_k}^{\text{out}}, & l_k \in Conv_k. \end{cases} \tag{8}$$

When the DNN-baesd task is divided into two segments, the output data of the last layer in the first segment is used for transmission. Let D_{l_k} (in bits) denote the output data size of layer l_k, which is given by (Liu et al. 2023)

$$D_{l_k} = \begin{cases} H_{l_k}W_{l_k}C_{l_k}^{\text{in}}\sigma, & l_k \in FC_k, \\ U_{l_k}^{\text{in}}\sigma, & l_k \in Conv_k, \end{cases} \tag{9}$$

where σ is the memory footprint per unit of data.

UAV Failure Model

To reflect the practical risk of UAV failure during task execution, we incorporate a UAV failure model into the system. Specifically, we consider permanent failures: each UAV may experience an unexpected failure (at most once) during the task execution period, and once a failure occurs, the UAV becomes permanently unavailable in all subsequent time slots.

We define a binary variable $s_j(t) \in \{0, 1\}$ to denote whether UAV j is operational or not during time slot t. If $s_j(t) = 1$, the UAV operates normally; otherwise, it fails. Additionally, we assume that at most $F_{\max}$ UAVs may fail throughout the entire task execution. Therefore, the following constraints need to be satisfied

$$s_j(t) \leq s_j(t-1), \forall j \in \mathcal{J}, \forall t \in 2, 3, ...\mathcal{T}, \tag{10}$$

$$\sum_{j=1}^{J}\left(1 - \prod_{t=1}^{T} s_j(t)\right) \leq F_{\max}. \tag{11}$$

When UAV j fails at time slot t (i.e., $s_j(t) = 0$), the tasks offloaded to UAV j need to be migrated to other available UAV. Therefore, we define the migration decision as $\beta_{i,jk}(t) \in \{0, 1\}$, when $\beta_{i,jk}(t) = 1$, the subsequent layers of DNN-based task b_i are migrated from UAV j to UAV k for processing; otherwise, $\beta_{i,jk}(t) = 0$. To ensure the continuous execution of DNN-based tasks and the stability of the system, task migration only occurs when a UAV fails and the target UAV (to which the tasks are migrated) must be operational. Therefore, we have

$$\sum_{k=1}^{J} \beta_{i,jk}(t) \leq 1 - s_j(t), \forall j \in \mathcal{J}, t \in \mathcal{T}, i \in \mathcal{I}, \tag{12}$$

$$\beta_{i,jk}(t) \leq s_k(t), \forall j, k \in \mathcal{J}, t \in \mathcal{T}, i \in \mathcal{I}, j \neq k. \tag{13}$$

Communication Model

1) MU-UAV Communication: Due to obstacle blockages in real signal propagation environments, both the line-of-sight (LoS) and non-LoS components should be incorporated. Therefore, we use the Rician fading channel model (Pang et al. 2022), the channel between MU i and UAV j is expressed as

$$h_{i,j}(t) = \sqrt{\frac{\beta_0}{d_{i,j}(t)^{\chi}}}\left(\sqrt{\frac{\Gamma}{\Gamma+1}}\bar{h}_{i,j}(t) + \sqrt{\frac{1}{\Gamma+1}}\tilde{h}_{i,j}(t)\right), \tag{14}$$

where $\bar{h}_{ij}(t)$ and $\tilde{h}_{ij}(t)$ represent the LoS component and non-LoS component, respectively. The distance between MU i and UAV j is denoted by $d_{i,j}(t)$, Γ represents the Rician factor, χ represents the path loss exponent, and the path loss exponent is represented by β_0. Therefore, the transmission rate from MU i to UAV j can be expressed as

$$r_{i,j}(t) = B_1 \log_2\left(1 + p_i(t)|h_{i,j}(t)|^2/\sigma_1^2\right), \tag{15}$$

where B_1 and σ_1 represent the bandwidth and the additive Gaussian noise between the UAVs and MUs, respectively. In addition, $p_i(t)$ denotes the transmission power of MU i.

2) UAV-UAV Communication: Considering the high hovering altitude of UAVs, the LoS link is the dominant one in UAV-UAV communications (Li et al. 2023). The channel gain between UAVs is given as

$$h_{j,k}(t) = \frac{h_0}{\|\mathbf{q}_j(t) - \mathbf{q}_k(t)\|^2}, \tag{16}$$

where h_0 represents the path-loss at the reference distance of 1 meter. Then the transmission rate between UAVs can be expressed as

$$r_{j,k}(t) = B_2 \log_2\left(1 + p_j(t)|h_{j,k}(t)|^2/\sigma_2^2\right), \tag{17}$$

where B_2 and σ_2 represent the bandwidth and additive Gaussian noise power between UAVs, respectively. And $p_j(t)$ denotes UAV j's transmission power.

Computation Model

1) Local Processing: During time slot t, the local processing time and energy consumption of the first segment of layers $l_k = 1$ to $\varphi_i(t) - 1$ of DNN-based task b_i are given by

$$\tau_i^{\text{loc}}(t) = \frac{\sum_{l_k=1}^{\varphi_i(t)-1} B_{l_k}}{C_0 f_i^{\text{loc}}(t)}, \tag{18}$$

$$e_i^{\text{loc}}(t) = \kappa f_i^{\text{loc}}(t)^2 \sum_{l_k=1}^{\varphi_i(t)-1} B_{l_k}/C_0, \tag{19}$$

where C_0 is the number of FLOPs within a CPU cycle, $f_i^{\text{loc}}(t) \in [0, f_i^{\max}]$ denotes MU i's CPU frequency, and κ is the effective capacitance coefficient.

The transmission time and energy consumption from MU i during time slot t can be calculated as

$$\tau_i^{\text{tr}}(t) = \sum_{j=1}^{J} \alpha_{i,j}(t) \frac{D_{\varphi_i(t)}}{r_{i,j}(t)}, \tag{20}$$

$$e_i^{\text{tr}}(t) = p_i(t)\tau_i^{\text{tr}}(t). \tag{21}$$

2) Task Offloading: If a UAV failure occurs, DNN-based tasks that were offloaded to the failed UAV need to be migrated to an available UAV. Thus, the transmission delay and energy consumption from UAV j to UAV k are denoted as

$$\tau_{i,jk}^{\text{tr}}(t) = \alpha_{i,j}(t)\beta_{i,jk}(t) \frac{D_{\varphi_i(t)}}{r_{i,j}(t)}, \tag{22}$$

$$e_{i,jk}^{\text{tr}}(t) = p_j(t)\tau_{i,jk}^{\text{tr}}(t), \tag{23}$$

where $p_j(t)$ represents UAV j's transmission power.

During time slot t, the processing time and energy consumption of DNN-based task b_i from the layer $l_k = \varphi_i(t)$ to L_k at the UAV can be expressed as

$$\tau_i^{\text{off}}(t) = \sum_{j=1}^{J} \alpha_{i,j}(t) \frac{\sum_{l_k=\varphi_i(t)}^{L_k} B_{l_k}}{C_0 f_{i,j}^{\text{uav}}(t)}, \tag{24}$$

$$e_i^{\text{off}}(t) = \kappa \sum_{j=1}^{J} \alpha_{i,j}(t) f_{i,j}^{\text{uav}}(t)^2 \sum_{l_k=\varphi_i(t)}^{L_k} B_{l_k}/C_0. \tag{25}$$

Here, $f_{i,j}^{\text{uav}}(t)$ is the computing resources allocated by UAV j to MU i.

As a result, the total latency and weighted energy consumption of DNN-based task b_i are modeled as

$$\tau_i(t) = \tau_i^{\text{loc}}(t) + \tau_i^{\text{tr}}(t) + \sum_{j=1}^{J} \sum_{k=1,k\neq j}^{J} \tau_{i,jk}^{\text{tr}}(t) + \tau_i^{\text{off}}(t), \tag{26}$$

$$\begin{aligned} e_{\text{tot}}(t) = &\sum_{i=1}^{I} \left(e_i^{\text{loc}}(t) + e_i^{\text{tr}}(t)\right) + \varpi_1 \sum_{j=1}^{J} e_j^{\text{fly}}(t) \\ &+ \varpi_2 \left(\sum_{i=1}^{I} e_i^{\text{off}}(t) + \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{k=1,k\neq j}^{J} e_{i,jk}^{\text{tr}}(t) \right), \end{aligned} \tag{27}$$

where ϖ_1 and ϖ_2 are non-negative energy consumption weighting factors.

Problem Formulation

Our objective is to minimize weighted energy consumption by jointly optimizing the computational resource allocation of UAVs $\boldsymbol{f} \triangleq \left\{f_{i,j}^{\text{uav}}(t), \forall i \in \mathcal{I}, j \in \mathcal{J}, t \in \mathcal{T}\right\}$, the MU association $\boldsymbol{\alpha} \triangleq \{\alpha_{i,j}(t), \forall i \in \mathcal{I}, j \in \mathcal{J}, t \in \mathcal{T}\}$, task migration $\boldsymbol{\beta} \triangleq \{\beta_{i,jk}(t), \forall i \in \mathcal{I}, j, k \in \mathcal{J}, t \in \mathcal{T}, j \neq k\}$, DNN partitioning decision $\boldsymbol{\varphi} \triangleq \{\varphi_i(t), \forall i \in \mathcal{I}, t \in \mathcal{T}\}$, the flight trajectory of UAVs $\mathbf{q} \triangleq \{\mathbf{q}_j(t), \forall j \in \mathcal{J}, t \in \mathcal{T}\}$ and transmission power $\boldsymbol{p} \triangleq \{p_i(t), \forall i \in \mathcal{I}, t \in \mathcal{T}\}$, while satisfying constraints on latency. The formulated problem is formulated as

$$\min_{\{\boldsymbol{\alpha},\boldsymbol{f},\boldsymbol{p},\mathbf{q},\boldsymbol{\varphi},\boldsymbol{\beta}\}} \sum_{t=1}^{T} e_{\text{tot}}(t) \tag{28a}$$

$$\text{s.t.} \quad (1), (5), (7), (12), (13), \tag{28b}$$

$$H_{\min} \leq H_j(t) \leq H_{\max}, \forall t \in \mathcal{T}, j \in \mathcal{J}, \tag{28c}$$

$$\alpha_{i,j}(t) \in \{0,1\}, \forall j \in \mathcal{J}, i \in \mathcal{I}, t \in \mathcal{T}, \tag{28d}$$

$$\varphi_i(t) \in \{1, ..., L_k\}, \forall t \in \mathcal{T}, i \in \mathcal{I}_k, \tag{28e}$$

$$\beta_{i,jk}(t) \in \{0,1\}, \forall j,k \in \mathcal{J}, i \in \mathcal{I}, t \in \mathcal{T}, \tag{28f}$$

$$\sum_{j=1}^{J} \beta_{i,jk}(t) \leq \alpha_{i,k}(t), \forall k \in \mathcal{J}, t \in, \mathcal{T}, i \in \mathcal{I}, \tag{28g}$$

$$\alpha_{i,j}(t) \leq s_j(t), \forall j \in \mathcal{J}, t \in \mathcal{T}, i \in \mathcal{I}, \tag{28h}$$

$$\sum_{i=1}^{I} \alpha_{i,j}(t) f_{i,j}^{\text{uav}}(t) \leq f_j^{\text{uav}}, \forall j \in \mathcal{J}, t \in \mathcal{T}, \tag{28i}$$

$$0 \leq p_i(t) \leq p_i^{\max}, \forall i \in \mathcal{I}, t \in \mathcal{T}, \tag{28j}$$

$$\tau_i(t) \leq \delta, \forall i \in \mathcal{I}, t \in \mathcal{T}, \tag{28k}$$

where $H_{\min}$ and $H_{\max}$ are the minimum and maximum altitude levels for UAV flight, $p_i^{\max}$ is the maximum transmission power, $f_i^{\max}$ and f_j^{uav} denote the maximum computational resource of user i and UAV j, respectively. (28c) is the constraint for the altitude of UAVs. (28d) restrict the MU association factor. (28e) implies that the DNN partitioning decision is integer. (28f) defines the task migration variable as a binary variables. (28g) and (28h) ensure the continuous execution of DNN-based tasks and the stability of the system. (28i) represents the restriction on the allocation of computing resources for UAVs. (28j) is the transmission power requirements of MUs. (28k) denotes the total delay requirement.

Proposed DRL Approach: SAC-PER

In this section, we propose a framework that integrates DRL with convex optimization. The proposed framework jointly optimizes DNN partitioning, MU association, UAV trajectory, task migration and computation resource allocation based on SAC-PER algorithm. Subsequently, the transmission power is determined in the DRL environment by applying Dinkelbach's method to the actions.

MDP Model

To apply DRL, we first define MDP which serves as a fundamental framework to model and solve sequential decision-making problems in a stochastic environment. The MDP

consists of four components: state space $\mathcal{S}$, action space $\mathcal{A}$, state transition probability function $\mathcal{P}$, and reward $\mathcal{R}$.

1) State: The environment's state s_t is obtained by the agent, which is expressed as

$$s_t = \{b_i, \mathbf{v}_j(t), \mathbf{q}_j(t), r_{i,j}(t), r_{j,k}(t)\}. \tag{29}$$

2) Action: The action a_t is chosen according to the received state, which is denoted by

$$a_t = \{\varphi_i(t), \alpha_{i,j}(t), \beta_{i,jk}(t), \mathbf{a}_j(t)\}. \tag{30}$$

3) Reward: The reward function of the agent must include the expected goals and penalties related to failure to request. Therefore, the reward function of the agent is written as

$$r(s_t, a_t) = -e_{\text{tot}}(t) P_{\text{dis}}(t) P_{\text{time}}(t) P_{\text{o}}(t), \tag{31}$$

where $P_{\text{dis}}(t)$ is the collision penalty between UAVs, $P_{\text{time}}(t)$ represents the timeout penalty, and $P_{\text{o}}(t)$ denotes the UAV exceeding the boundary penalty.

Optimization for Transmission Power

By combining equations (15), (20), and (21) under the constraint $\tau_i^{\text{tr}}(t) \leq \delta$, the transmission energy consumption minimization problem is formulated as

$$\min_{\{\boldsymbol{p}\}} \quad \sum_{j=1}^{J} \frac{\alpha_{i,j}(t) p_i(t) D_{\varphi_i(t)}}{B_1 \log_2\left(1 + p_i(t)|h_{i,j}(t)|^2/\sigma_1^2\right)} \tag{32a}$$

$$\text{s.t.} \quad \hat{p}_i(t) \leq p_i(t) \leq p_i^{\max}, \forall i \in \mathcal{I}, t \in \mathcal{T}, \tag{32b}$$

where $\hat{p}_i(t)$ denotes the minimum feasible transmission power of MU i at time slot t, expressed as

$$\hat{p}_i(t) = \sigma_1^2 \sum_{j=1}^{J} |h_{i,j}(t)|^{-2} \left(2^{\alpha_{i,j}(t) D_{\varphi_i(t)} (\delta B_1)^{-1}} - 1\right). \tag{33}$$

To handle the fractional problem (32), Dinkelbach's transform (Shen and Yu 2018) is applied, reformulating the objective (32a) as

$$\min_{\{\boldsymbol{p}\}} \sum_{j=1}^{J} \left(\alpha_{i,j}(t) p_i(t) D_{\varphi_i(t)} - y_i B_1 \log_2\left(1 + \frac{p_i(t)|h_{i,j}(t)|^2}{\sigma_1^2}\right)\right) \tag{34}$$

with a new auxiliary variable y_i, iteratively updated by

$$y_i^{n+1} = \sum_{j=1}^{J} \frac{\alpha_{i.j}(t) p_i^n(t) D_{\varphi_i(t)}}{B_1 \log_2\left(1 + p_i^n(t)|h_{i,j}(t)|^2/\sigma_1^2\right)}. \tag{35}$$

Here, n denotes the iteration index. Convergence is achieved by alternatively solving $p_i[t]$ in (34) and updating y in (35). The optimal transmission power $p_i^*[t]$ is obtained after successive iterations.

SAC-based DRL Training Framework

1) SAC Framework: The SAC algorithm aims to maximize the long-term return while encouraging exploration via entropy regularization. Its objective function is formulated as

$$J(\pi) = \sum_{t=1}^{T} \mathbb{E}_{(s_t, a_t) \sim \rho_\pi} \left[\gamma^{t-1} r(s_t, a_t) + \alpha \mathcal{H}(\pi(\cdot|s_t))\right], \tag{36}$$

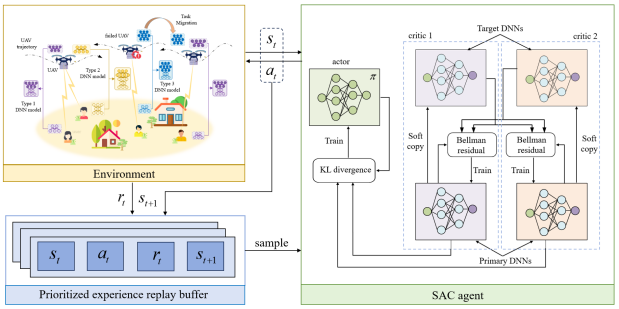

Figure 2: The training framework of SAC-PER.

where α is the temperature coefficient and $\mathcal{H}$ denotes the policy entropy. To balance exploration and exploitation, α is adaptively updated by minimizing the following loss

$$\alpha_t^* = \arg\min_{a_t} \mathbb{E}_{a_t \sim \pi_t^*} \left[-\alpha_t \log\left(\pi_t^*(a_t|s_t; s_t)\right) - \alpha_l \mathcal{H}_{\min}\right]. \tag{37}$$

SAC follows policy iteration framework. In the policy evaluation phase, the Q-function is updated by the Bellman exception equation with entropy, expressed as $Q_\pi(s_t, a_t) = r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim t_s}[v_\pi(s_{t+1})]$. In the policy improvement phase, the actor network is optimized via reparameterization, which is given by

$$L_\pi(\varphi) = \mathbb{E}_{s_t \sim \mathcal{D}} \mathbb{E}_{a_t \sim \pi_\varphi} \left[\alpha \log_2\left(\pi_\varphi(a_t|s_t)\right) - \min_{i=1,2} Q_{\omega i}(s_t, a_t)\right]. \tag{38}$$

2) PER: To enhance sample efficiency, PER assigns a priority each transition based on its temporal difference error

$$|\delta_t| = \frac{1}{2} \sum_{i=1}^{2} |r(s_t, a_t) + \gamma Q_{\hat{\omega}_i}(s_{t+1}, a_{t+1}) - Q_{\omega_i}(s_t, a_t)|. \tag{39}$$

The priority $p_i = (\delta_i + \varepsilon)^\eta$ determines the sampling probability $P(i) = p_i^{\beta_1} / \sum_k p_k^{\beta_1}$ and the importance weight for bias correction is $\omega_i = (1/WP(i))^{\beta_2}$. Figure 2 illustrates the training process of the proposed SAC-PER algorithm.

Simulation Results

In this section, numerical results are presented to demonstrate the feasibility and effectiveness of the proposed algorithm. We consider a 500 m × 500 m square MEC network area, where UAVs can fly at an altitude of 100 m to 200 m. Unless otherwise specified, we set the number of MUs $I = 10$ and the number of UAVs $J = 3$. For the DNN model, we selected the pre-trained AlexNet, VGG16 and VGG13. The simulation parameters are detailed as follows: $v_{\max} = 20$m/s, $a_{\max} = 5$ m/s^2, $d_{\text{safe}} = 5$ m, $f_i^{\max}$=1 GHz, $f_j^{\text{uav}} = 10$ GHz, $p_{\max} = 0.5$ W, $N = 60$ s, $T = 60$, $\sigma = 4$ Byte, $C_0 = 8$ FLOPs/cycle, $\varpi_1 = 0.0001$, $\varpi_2 = 0.001$, $B_1 = 10$ MHz, $B_2 = 15$ MHz, $h_0 = -30$ dB, $\kappa = 10^{-28}$.

To verify the performance of the proposed scheme, we compare it against four baseline schemes. *i)* SAC algorithm; *ii)* Randomized ensemble double Q-learning (REDQ) algorithm; *iii)* Offloading execution: all layers of the DNN-based

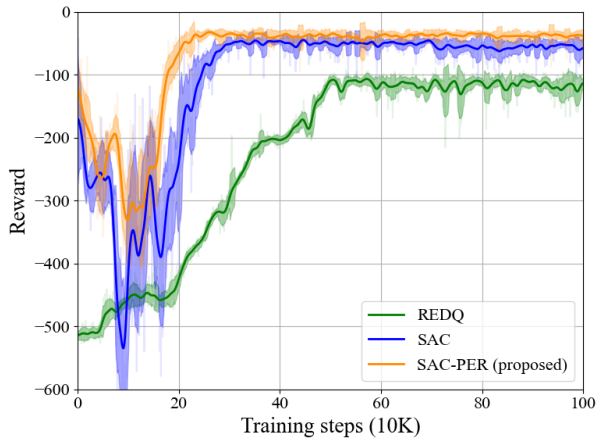

Figure 3: The convergence performance of different algorithms, where $I = 10$, $J = 3$.

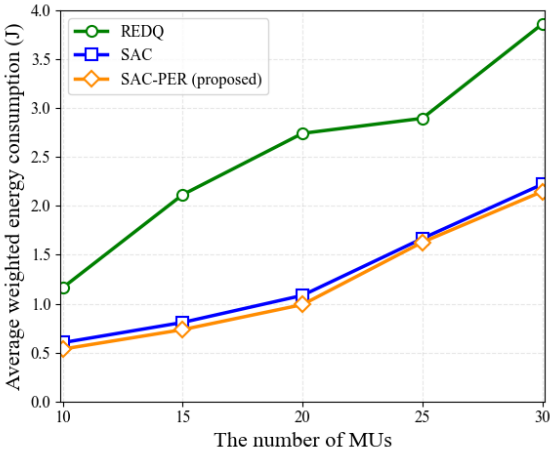

Figure 4: The performance comparison versus different numbers of users, where $J = 3$.

tasks are computed on UAVs; *iv)* Random partitioning: The DNN-based tasks are randomly partitioned and offloaded to UAVs for computation.

In figure 3, we evaluate the performance of the SAC-PER algorithm against the SAC and REDQ algorithms. SAC-PER shows clear improvements in both convergence speed and overall performance. Within about 20K training steps, its reward rapidly increases and stabilizes. In contrast, SAC converges around 30K steps and REDQ around 50K, both achieving lower rewards. During the initial training phase, SAC and SAC-PER exhibit a temporary decrease in reward due to exploration, where the agents learn optimal policies and may take suboptimal actions. As training continues, the algorithms converge to more efficient strategies, leading to higher rewards.

To verify the effectiveness of the proposed scheme in scenarios with multiple MUs, we compare the performance of

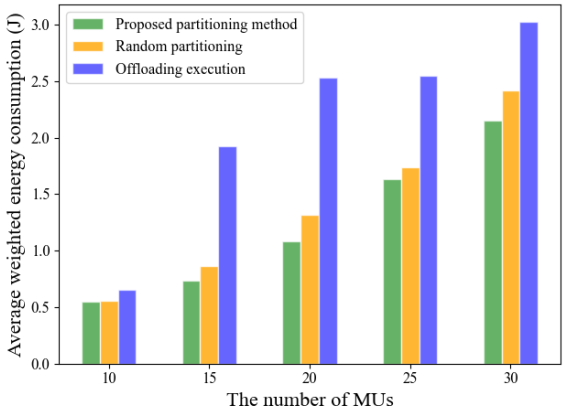

Figure 5: The performance comparison versus different partitioning methods, where $I = 10$, $J = 3$.

various schemes under different numbers of users. As shown in figure 4, the proposed SAC-PER algorithm consistently achieves the lowest average weighted energy consumption, while the REDQ algorithm shows the highest. Although the gap between SAC and SAC-PER is relatively small, SAC-PER outperforms SAC due to its enhanced learning efficiency. Both algorithms can adapt effectively to multi-user environments, but SAC-PER further optimizes energy consumption through more refined power control strategies and prioritized experience replay, enabling it to focus on critical experiences and achieve superior energy efficiency.

Figure 5 illustrates the performance comparison among three partitioning strategies. The proposed partitioning method consistently achieves the lowest average weighted energy consumption, demonstrating its superior efficiency in balancing computational and communication loads. In contrast, the random partitioning method exhibits moderate energy consumption, while the offloading execution method incurs the highest energy cost, particularly as the number of UAVs increases. This comparison further highlights the effectiveness and necessity of the proposed DNN partitioning method in achieving energy-efficient in multi-UAV-assisted MEC system.

Conclusion

In this paper, we have proposed a robust multi-UAV-assisted MEC framework to ensure continuous DNN inference and reliable task execution under UAV failures. The framework integrated adaptive DNN partitioning and failure-aware task migration, allowing incomplete DNN layers on failed UAVs to be reassigned to operational UAVs. We have formulated a joint optimization problem to minimize weighted energy consumption under DNN latency constraints by jointly optimizing MU association, UAV trajectories, task migration, DNN partitioning, computation resource allocation, and transmission power. To solve this high-dimensional problem efficiently, we have designed a hybrid algorithm combining SAC-PER and convex optimization. Simulation results showed that the proposed framework achieves notable

improvements in energy efficiency, convergence speed, and robustness compared with benchmark schemes.

References

Gao, M.; Shen, R.; Shi, L.; Qi, W.; Li, J.; and Li, Y. 2023. Task partitioning and offloading in DNN-task enabled mobile edge computing networks. *IEEE Transactions on Mobile Computing*, 22(4): 2435–2445.

He, Y.; Gan, Y.; Cui, H.; and Guizani, M. 2023. Fairness-based 3-D multi-UAV trajectory optimization in multi-UAV-assisted MEC system. *IEEE Internet of Things Journal*, 10(13): 11383–11395.

Li, B.; Liu, W.; Xie, W.; Zhang, N.; and Zhang, Y. 2023. Adaptive digital twin for UAV-assisted integrated sensing, communication, and computation networks. *IEEE Transactions on Green Communications and Networking*, 7(4): 1996–2009.

Li, X.; Qin, Y.; Huo, J.; and Huangfu, W. 2024. Computation offloading and trajectory planning of multi-UAV-enabled MEC: A knowledge-assisted multiagent reinforcement learning approach. *IEEE Transactions on Vehicular Technology*, 73(5): 7077–7088.

Liu, K.; Liu, C.; Yan, G.; Lee, V. C. S.; and Cao, J. 2023. Accelerating DNN inference with reliability guarantee in vehicular edge computing. *IEEE/ACM Transactions on Networking*, 31(6): 3238–3253.

Liu, Q.; Shi, L.; Sun, L.; Li, J.; Ding, M.; and Shu, F. 2020. Path planning for UAV-mounted mobile edge computing with deep reinforcement learning. *IEEE Transactions on Vehicular Technology*, 69(5): 5723–5728.

Liu, Z.; Du, H.; Lin, J.; Gao, Z.; Huang, L.; Hosseinalipour, S.; and Niyato, D. 2025. DNN partitioning, task offloading, and resource allocation in dynamic vehicular networks: A Lyapunov-guided diffusion-based reinforcement learning approach. *IEEE Transactions on Mobile Computing*, 24(3): 1945–1962.

Mach, P.; and Becvar, Z. 2017. Mobile edge computing: A survey on architecture and computation offloading. *IEEE Communications Surveys & Tutorials*, 19(3): 1628–1656.

Pang, X.; Zhao, N.; Tang, J.; Wu, C.; Niyato, D.; and Wong, K.-K. 2022. IRS-assisted secure UAV transmission via joint trajectory and beamforming design. *IEEE Transactions on Communications*, 70(2): 1140–1152.

Shen, K.; and Yu, W. 2018. Fractional programming for communication systems—Part I: Power control and beamforming. *IEEE Transactions on Signal Processing*, 66(10): 2616–2630.

Wen, D.; Liu, P.; Zhu, G.; Shi, Y.; Xu, J.; Eldar, Y. C.; and Cui, S. 2024. Task-oriented sensing, computation, and communication integration for multi-device edge AI. *IEEE Transactions on Wireless Communications*, 23(3): 2486–2502.

Yan, H.; Gu, Y.; He, H.; Ning, X.; Wang, Q.; and Cheng, L. 2025. DNN-based task partitioning and offloading in edge-cloud collaboration within electric vehicles. *IEEE Transactions on Consumer Electronics*, 71(2): 4100–4109.

Zeng, Y.; Xu, J.; and Zhang, R. 2019. Energy minimization for wireless communication with rotary-wing UAV. *IEEE Transactions on Wireless Communications*, 18(4): 2329–2345.

Zhang, Z.; Li, Y.; Huang, C.; Guo, Q.; Yuen, C.; and Guan, Y. L. 2019. DNN-aided block sparse bayesian learning for user activity detection and channel estimation in grant-free non-orthogonal random access. *IEEE Transactions on Vehicular Technology*, 68(12): 12000–12012.