Deep-Graph-Based Reinforcement Learning for Joint Cruise Control and Task Offloading for Aerial Edge Internet of Things (EdgeIoT)

Kai Li*
Wei Ni
Xin Yuan
Alam Noor*
Abbas Jamalipour

*CISTER Research Centre

CISTER-TR-230704

2022/06/10

Deep-Graph-Based Reinforcement Learning for Joint Cruise Control and Task Offloading for Aerial Edge Internet of Things (EdgeIoT)

Kai Li*, Wei Ni, Xin Yuan, Alam Noor*, Abbas Jamalipour

*CISTER Research Centre

Polytechnic Institute of Porto (ISEP P.Porto)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8321159

E-mail: kai@isep.ipp.pt, Wei.Ni@data61.csiro.au, xin.yuan@data61.csiro.au, alamn@isep.ipp.pt

https://www.cister-labs.pt

Abstract

This article puts forth an aerial edge Internet of Things (EdgeIoT) system, where an unmanned aerial vehicle (UAV) is employed as a mobile-edge server to process mission-critical computation tasks of ground Internet of Things (IoT) devices. When the UAV schedules an IoT device to offload its computation task, the tasks buffered at the other unselected devices could be outdated and have to be canceled. We investigate a new joint optimization of UAV cruise control and task offloading allocation, which maximizes tasks offloaded to the UAV, subject to the IoT device 19s computation capacity and battery budget, and the UAV 19s speed limit. Since the optimization contains a large solution space while the instantaneous network states are unknown to the UAV, we propose a new deep-graph-based reinforcement learning framework. An advantage actor 13critic (A2C) structure is developed to train the real-time continuous actions of the UAV in terms of the flight speed, heading, and the offloading schedule of the IoT device. By exploring hidden representations resulting from the network feature correlation, our framework takes advantage of graph neural networks (GNNs) to supervise the training of UAV 19s actions in A2C. The proposed graph neural network-enabled A2C (GNN-A2C) framework is implemented with Google Tensorflow. The performance analysis shows that GNN-A2C achieves fast convergence and reduces considerably the task missing rate in aerial EdgeIoT.

Deep Graph-based Reinforcement Learning for Joint Cruise Control and Task Offloading for Aerial Edge Internet-of-Things (EdgeIoT)

Kai Li, *Senior Member, IEEE,* Wei Ni, *Senior Member, IEEE,* Xin Yuan, *Member, IEEE,* Alam Noor, *Student Member, IEEE,* and Abbas Jamalipour, *Fellow, IEEE*

***Abstract*—This paper puts forth an aerial edge Internet-of-Things (EdgeIoT) system, where an unmanned aerial vehicle (UAV) is employed as a mobile edge server to process mission-critical computation tasks of ground Internet-of-Things (IoT) devices. When the UAV schedules an IoT device to offload its computation task, the tasks buffered at the other unselected devices could be outdated and have to be cancelled. We investigate a new joint optimization of UAV cruise control and task offloading allocation, which maximizes tasks offloaded to the UAV, subject to the IoT device's computation capacity and battery budget, and the UAV's speed limit. Since the optimization contains a large solution space while the instantaneous network states are unknown to the UAV, we propose a new deep graph-based reinforcement learning framework. An advantage actor-critic (A2C) structure is developed to train the real-time continuous actions of the UAV in terms of the flight speed, heading, and the offloading schedule of the IoT device. By exploring hidden representations resulting from the network feature correlation, our framework takes advantage of graph neural networks (GNN) to supervise the training of UAV's actions in A2C. The proposed GNN-A2C framework is implemented with Google Tensorflow. The performance analysis shows that GNN-A2C achieves fast convergence and reduces considerably the task missing rate in aerial EdgeIoT.**

***Index Terms*—Unmanned aerial vehicle, Aerial EdgeIoT, Graph neural network, Deep reinforcement learning, Cruise control, Task offloading**

I. Introduction

In recent years, mobile edge computing (MEC) has provided a promising solution to enabling cloud computing services in edge Internet-of-Things (EdgeIoT) [1], [2]. To alleviate the storage and computing limitations and prolong the lifetime, the IoT devices can offload their local computation-intensive tasks to computationally powerful edge servers. By making use of computational resources at the edge server, MEC improves the computation capacity of the IoT devices with computation-hungry applications, such as augmented reality and virtual reality [3], [4].

With the proliferation of IoT devices, EdgeIoT applications operate increasingly in human-unfriendly, unattended areas, such as remote farmlands, rural vineyards or highways, to carry out computation-intensive operations, e.g., monitoring crops and water-efficient irrigation control under complex environmental conditions [5]. In these harsh scenarios, the IoT devices are equipped with energy harvesting modules, e.g., solar panels, to harvest renewable energy from ambient resources to recharge its battery [6]. Since conventional terrestrial communication networks are distributed sparsely, reliable connections for the IoT devices are difficult to be guaranteed [7].

Figure 1 studies an aerial EdgeIoT, which explores the excellent maneuverability of an unmanned aerial vehicle (UAV) to extend the coverage of EdgeIoT [8]. With a limited communication range, an IoT device buffers its computation tasks in its local task queue awaiting to be offloaded to the UAV [9]. Moreover, the UAV with a lightweight edge server can physically approach the geo-distributed IoT devices to process their computation tasks. A line-of-sight (LoS) communication link between the UAV and the IoT device improves the channel quality and enables a high-speed task offloading rate at the IoT device [10].

When the UAV schedules an IoT device for task offloading, the computation tasks in the buffer of the other unselected devices may be outdated and have to be cancelled in the case that the task buffers are full. This gives rise to a large number of dropped computation tasks. Moreover, offloading a task from a scheduled IoT device can experience a poor channel quality since the UAV's movement results in time-varying channel dynamics. The offloading errors, in turn, overflow the task buffer of the unscheduled IoT devices and raise the task missing rate (TMR). In practice, the instantaneous knowledge of the task buffer and channel qualities of the IoT devices is unknown to the UAV. Hence, scheduling the task offloading online in the presence of the joint cruise control (i.e., speed and heading) is crucial to minimize the TMR, resulting from the computation task cancellation and the offloading errors in the aerial EdgeIoT.

In this paper, we optimize online the joint UAV cruise control and task offloading schedule, where the task queue

K. Li, and A. Noor are with Real-Time and Embedded Computing Systems Research Centre (CISTER), 4249–015 Porto, Portugal (E-mail:{kai,alamn}@isep.ipp.pt).

W. Ni and X. Yuan with the Digital Productivity and Services Flagship, Commonwealth Scientific and Industrial Research Organization (CSIRO), Sydney, Australia (E-mail: {wei.ni,xin.yuan}@data61.csiro.au).

A. Jamalipour is with the School of Electrical and Information Engineering, The University of Sydney, Australia (E-mail: a.jamalipour@ieee.org).

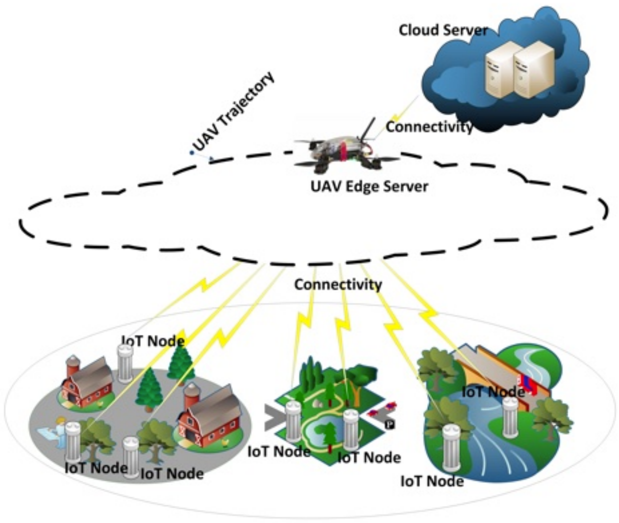


Fig. 1: Illustration of the aerial EdgeIoT. The UAV that carries a lightweight edge server controls the trajectory and patrols over the area of interest. The IoT devices can offload their computation tasks to the UAV for processing or forwarding to the remote cloud server.

lengths, battery statuses, and channel quality of the IoT devices are unknown to the UAV. An optimization model is formulated to maximize the number of computation tasks offloaded to the UAV, subject to the computation capacity and battery budget of the IoT device, and the speed limit of the UAV. Given unknown network states, an advantage actor-critic (A2C) structure is developed to train real-time continuous actions of the UAV in terms of the flight speed, heading, and the offloading schedule of the IoT device.

Since the IoT devices are deployed in the same target field, the energy harvesting on the solar or wind power is affected by the same weather conditions. The devices' battery status and the number of generated tasks can be correlated. Similarly, the channel quality variation is also correlated. By exploring hidden representations resulting from the network feature correlation, we propose a new deep graph-based reinforcement learning framework, named graph neural network-enabled A2C (GNN-A2C), where the GNN is developed to supervise the training of UAV's actions in A2C.

The GNN constructs a graph consisting of vertexes (i.e., the network states) linked by edges (representing the correlations) [11]. In the GNN, the hidden representations of vertexes are iteratively passed between neighboring vertexes. Those hidden representations are propagated throughout the graph using multiple iterations until a fixed vertex is found. The final hidden representation is then used for predicting future network states about the vertexes. Furthermore, a long short-term memory (LSTM) model is developed in the proposed GNN-A2C to predict the time-varying UAV-ground channels and task generation at the IoT devices based on the extracted features in GNN.

The key contributions of this paper are summarized as follows:

1) To our knowledge, this work is the first attempt to exploit the task offloading scheduling optimization online in the presence of the joint UAV cruise control (i.e., speed and heading) to minimize the TMR, resulting from the computation task cancellation and the offloading errors in the aerial EdgeIoT.

2) Since the optimization contains a large solution space while the instantaneous network states are unknown to the UAV, the new combinatorial learning architecture of GNN-A2C is proposed. The new GNN-A2C framework develops GNN to supervise the training of UAV's real-time continuous actions in A2C, which sufficiently explores the hidden representations based on the network feature correlation.

3) To validate this new deep graph-based reinforcement learning framework, the proposed GNN-A2C is implemented in Python 3.9 with Google Tensorflow, setting up on a Linux workstation with 64-bit Ubuntu 18.04. Numerical studies show that the GNN-A2C significantly reduces the TMR in aerial EdgeIoT while effectively controlling the flight cruise to process the computation tasks.

This paper is organized as follows. Section II presents the literature on the task offloading and trajectory design in UAV-assisted MEC. In Section III, the system model of the aerial EdgeIoT is formulated. In Section IV, we propose the joint optimization of the cruise control and task offloading to maximize the computation tasks offloaded from the IoT devices to the UAV. We investigate the proposed GNN-A2C architecture in Section V. The implementation and performance of the GNN-A2C are presented in Section VI. Finally, Section VII concludes the paper.

II. Related Work

This section reviews the literature on the task offloading and trajectory design in UAV-assisted MEC.

In [12], an edge server is mounted to the UAV, which is employed to provide computation services for the ground IoT devices. Given several hot-spot areas of the computation tasks, a UAV allocation scheme is developed to determine the hover locations, aiming to increase the number of served tasks. In [13], a resource allocation strategy is studied, where one portion of bits at the ground device is computed locally, and the remaining portion is transmitted to UAV for computing. The strategy controls the bit allocation, transmit power of the ground device and the UAV, and the UAV's trajectory to reduce the energy consumption of the ground device. A UAV-aided MEC is presented in [14], where the UAV's trajectory, the offloading tasks, and the transmission scheduling are designed to reduce the total delay of all the ground devices. The authors study the UAV's response delay and computation resource allocation with a Poisson point process model in MEC [15]. Given the successful transmission probability, computation resources of the UAV are estimated to meet the time constraint of latency-sensitive tasks.

In [16], a UAV is served by cellular ground base stations for offloading tasks. Given the maximum speed of the UAV and the computation capacity of the ground stations, the UAV's mission completion time is reduced by designing the flight trajectory with the computation offloading scheduling. Computation offloading scheduling can be developed with the UAV's trajectory to reduce the energy consumption and mission completion time of the UAV [17]. Since the completion time minimization is non-convex, a path discretization approximation model is utilized to decouple the problem into two subproblems addressed with a successive convex approximation (SCA). The energy consumption of the UAV on communication and computation can be reduced by scheduling transmit power, time and offloading tasks [18]. Due to the non-convexity, a problem decomposition method is used, where the problem is decomposed into two subproblems which are solved by the existing Lagrangian duality and SCA. According to the required communication and computation latency, in [19], the UAV's trajectory is designed with the task offloading and resource allocation to reduce the energy consumption.

In [20], the transmit power and task offloading allocation is developed with the UAV's trajectory to improve the energy efficiency of the UAV. Due to limited knowledge of the ground device, a mobility estimation model is used to design the UAV's trajectory. An allocation of the UAV's discrete waypoints, communication and computing resource is studied in [21] to reduce the service delay of the IoT devices. The authors in [22] present a task data allocation and trajectory planning scheme to ensure a fair energy consumption of the ground devices.

Given an unknown MEC environment, deep reinforcement learning (DRL) is utilized in [23] for designing the UAV's trajectory to serve the ground devices. The energy efficiency of the UAV can be improved subject to the quality-of-service requirement of the ground device. In [24], deep Q-network (DQN) and DRL are applied to learn the offloading task ratio as well as the UAV trajectory to reduce the energy consumption of the UAV and the ground devices. In wireless powered sensor networks [25], [26], deep deterministic policy gradient (DDPG) or DQN-based strategies are studied to schedule the UAV's trajectory and resource allocation to minimize the buffer overflow of the ground sensors.

The existing literature focuses on the trajectory planning and resource allocation to shorten the mission completion time or improve the service coverage and energy efficiency, where the correlation of the network state is not considered. In contrast, this paper investigates the challenge that poor cruise control of the UAV results in an unsuccessful task offloading of the selected IoT device while the tasks in the buffer at unselected devices are outdated and cancelled. In addition, a new GNN-A2C structure is developed, where the GNN extracts the correlated features of the network states to supervise the training of the UAV's actions.

Particularly, GNN is a neural network utilizing graph structures to capture the dependence of graphs via message passing between the vertexes of graphs [27]. For message passing, the vertexes are connected by edges, where the hidden representations of the vertexes are iteratively passed between neighboring vertexes through the edges. The hidden representations are propagated throughout the graph iteratively until a fixed point is found. The final hidden representation is used to predict the features of the vertexes. Although GNN is inspired by a convolutional neural network (CNN) which uses convolutional filters to reduce the number of parameters in fully connected deep neural networks, the convolution operations in the CNN are only applicable to grid-like image data in the Euclidean space. GNN redefines the convolution operation of graphs, generalizing the convolution operations performed in CNN to convolutions performed on arbitrarily structured data.

TABLE I: Notation and definition

Notation	Definition
N	total number of IoT devices
$\mathcal{G}(\mathcal{V},\mathcal{E};\mathbf{w})$	a graph for modeling the network states
$\mathcal{V}$, $\mathcal{E}$	a vertex or an edge of the graph $\mathcal{G}(\mathcal{V},\mathcal{E};\mathbf{w})$
$\mathbf{f}$	feature representation in GNN
K	total number of layers that are composed in the graph
v_t	instantaneous speed of the UAV
b_t^{UAV}	battery level of the UAV
(x_t, y_t, z)	location of the UAV
$\Lambda_i(t)$	Offloaded tasks of device i to the UAV
S_α, S_β	network states
a_{S_α}	the action taken by the UAV at state S_α
i_{S_α}	the selected ground device at state S_α
$\mathcal{Q}$	maximum number of tasks of the ground device
E	battery capacity of the ground device
$b_i(t)$	battery of device i
$q_i(t)$	number of tasks generated at device i
$H_i(t)$	SNR received at the UAV from device i
ep	number of training episodes
t_{learning}	training time of each episode
μ	discount value of the action-value function
$\nabla\Gamma(\vartheta)$	policy gradient in A2C

III. System Model

In this section, the system model of aerial EdgeIoT is presented. Table I lists the notations used in this paper.

The studied aerial EdgeIoT consists of N ground devices, which offload computing tasks to a remote cloud server via the UAV, as depicted in Figure 1. We consider that each ground device i can offload a part of its computation load, i.e., a computation task, to the UAV equipped with a portable edge server through a wireless link. The UAV patrols over the target field, while scheduling the ground devices for the task offloading. The battery level of device i $(\in [1, N])$ at time t is $b_i(t)$. Since the device's battery capacity is limited, $b_i(t) \leq E$ with E (in Joules) being the battery capacity of the ground device. It is assumed that the ground devices undergo a random task generation following the Poisson distribution. The tasks are buffered at the radio transmitter of the ground device and wait to be offloaded to the edge UAV, where the offloading of the buffered tasks follows the first-come-first-serve discipline. Particularly, the number of tasks generated by device i at t is $q_i(t) \leq \mathcal{Q}$, where $\mathcal{Q}$ denotes the maximum number of tasks that can be buffered at the ground device. The buffer size of the device is finite. As a result, the newly generated tasks have to be dropped if $q_i(t) > \mathcal{Q}$, i.e., the buffer overflows.

Let (x_t, y_t, z) denote the position of the UAV at time t, and the UAV flies at a constant altitude of z meters [28]. With safety consideration, the instantaneous flight speed of the UAV, denoted by v_t, has to be no larger than the maximum speed V_{max}, which depends on the maximum available thrust of the motors, the weight of the UAV, and the structural strength. Thus, we have $v_t \leq V_{max}$. Δt is the time when the UAV moves from (x_t, y_t, z) to (x_{t+1}, y_{t+1}, z), respectively. The acceleration at (x_t, y_t, z) can be given by

$$\Delta v_t / \Delta t = (v_{t+1} - v_t)/\Delta t, \tag{1}$$

where

$$0 \leq \Delta v_t / \Delta t \leq V_{max}/\Delta t. \tag{2}$$

Considering aeronautics, a smooth turn mobility is applied to the UAV, where the change of the heading direction follows an arc of a circle [29]. Suppose that θ_t is the turning angle of the UAV at time t, and (x_t^c, y_t^c, z) are the coordinates of the circle centre at time t. Thus, we have

$$\theta_t = \arctan\left(\frac{y_{t+1} - y_t^c}{x_{t+1} - x_t^c}\right) - \arctan\left(\frac{y_t - y_t^c}{x_t - x_t^c}\right). \tag{3}$$

Notably, it is assumed that the UAV does not move backward, i.e., $\theta_t \neq 2\pi$.

Let b_t^{UAV} denote the battery level of the UAV at time t, which can be measured by onboard sensors in real-time. Suppose that the UAV has a solar panel to harvest energy and a rechargeable battery to store the energy. The harvested solar power of the UAV at time t can be given by [30]

$$\begin{aligned}\Delta b_t^{\text{UAV}} = \psi C_{\text{solar}} P_{\text{solar}} \exp\Big(\frac{-\varkappa}{\cos\phi_t^{\text{solar}}}(1 - 2.2556 \\ \times 10^{-5} z)^{5.2561}\Big),\end{aligned} \tag{4}$$

where $\psi \in (0,1)$ and C_{solar} denote the charging efficiency and size of the solar panel, respectively. P_{solar} is the constant power intensity of the solar beams. $\varkappa > 0$ is the sum atmospheric extinction. ϕ_t^{solar} denotes the solar zenith angle at time t.

The propulsion energy of the UAV is given by [31]

$$\begin{aligned}\Delta b_t^{\text{UAV}\prime} =& P_0\Big(1 + \frac{3v_t^2}{\omega_t^2}\Big) + P_0'\Big(\sqrt{1 + \frac{v_t^4}{4v_0^4}} - \frac{v_t^2}{2v_0^2}\Big)^{1/2} \\ &+ \frac{1}{2}\xi_{\text{drag}}\rho_{\text{air}}\xi_{\text{rotor}}\xi_{\text{rotor}}' v_t^3,\end{aligned} \tag{5}$$

where P_0 and P_0' are two constants. ω_t is the tip speed of the rotor blade. v_0 is the mean rotor induced velocity in hover. ξ_{drag} and ξ_{rotor} denote the fuselage drag ratio and rotor solidity, respectively. ρ_{air} and ξ_{rotor}' denote the air density and rotor disc area, respectively. Thus, we have

$$b_t^{\text{UAV}} + \Delta b_t^{\text{UAV}} - \Delta b_t^{\text{UAV}\prime} > b_0^{\text{UAV}}, \tag{6}$$

where b_0^{UAV} is the minimum safe battery level of the UAV.

IV. Joint Optimization of Cruise Control and Task Offloading

In this section, joint optimization of cruise control and task offloading in aerial EdgeIoT is formulated to maximize the computation tasks offloaded from the IoT devices to the UAV. Given the coordinates of the UAV (x_t, y_t, z), the flying speed v_t can be decomposed to x and y components, i.e., v_t^x and v_t^y. The location of the UAV at time $t+1$ can be given by

$$\left\{\begin{array}{l} x_{t+1} = x_t + v_t^x \cdot \Delta t \\ y_{t+1} = y_t + v_t^y \cdot \Delta t \end{array}\right., \tag{7}$$

where Δt is the flight duration from t to $t+1$ [32]. Assume that the location of ground device i is (x_i, y_i). Based on (7), the distance between device i and the UAV is

$$d_i(t) = \sqrt{(x_t - x_i)^2 + (y_t - y_i)^2 + z^2}. \tag{8}$$

Due to potential obstructions between the IoT device and the UAV, the propagation path yields a mixed LoS/non-LoS channel model [33], which contains two components

$$\left\{\begin{array}{l} \delta_i^{\text{LoS}}(t) = 20\log(f_c) + 20\log(4\pi/v_c) + \\ \qquad\qquad 20\log(d_i(t)) + \eta_{\text{LoS}} \\ \delta_i^{\text{NLoS}}(t) = 20\log(f_c) + 20\log(4\pi/v_c) + \\ \qquad\qquad 20\log(d_i(t)) + \eta_{\text{NLoS}} \end{array}\right., \tag{9}$$

where f_c and v_c denote the carrier frequency and the speed of light, respectively. η_{LoS} and η_{NLoS} stand for the excessive path loss of LoS and non-LoS, respectively. η_{LoS} and η_{NLoS} are two constants dependent on different applications [34].

Let $\text{Pr}_i^{\text{LoS}}(t)$ denote the LoS probability, as given by

$$\text{Pr}_i^{\text{LoS}}(t) = \frac{1}{1 + a\exp(-b[\varphi_i(t) - a])}, \tag{10}$$

where a and b are two Sigmoid function parameters. $\varphi_i(t)$ is the elevation angle between device i and the UAV at t. The signal-to-noise ratio (SNR) received at the UAV from device i, denoted by $H_i(t)$, is given by

$$H_i(t) = \sqrt{\frac{c_2^{-1}\text{In}(\frac{c_1}{\epsilon})(2^{\rho_i(t)} - 1)}{(H_i^{\text{LoS}}(t))^2}}, \tag{11}$$

where c_1 and c_2 are two constants, ϵ is the bit error rate (BER), and ρ_i is the modulation scheme of the IoT device. According to (9) and (10), we can obtain

$$H_i^{\text{LoS}}(t) = \text{Pr}_i^{\text{LoS}}(t) \times \delta_i^{\text{LoS}}(t) + \text{Pr}_i^{\text{NLoS}}(t) \times \delta_i^{\text{NLoS}}(t), \tag{12}$$

where $\text{Pr}_i^{\text{NLoS}}(t) = 1 - \text{Pr}_i^{\text{LoS}}(t)$.

According to (11), a regression model can map the SNR to the number of computation tasks successfully collected by the UAV edge server. The regression model is given by [35]

$$\Lambda_i(t) = (1 - \frac{1}{2}e^{\beta_1 - \beta_0 H_i(t)})^{8(2f-l)}, \tag{13}$$

where β_0 and β_1 are two constants in the regression model. β_0 controls the shape of the regression curve and β_1 induces horizontal shifts of the curve. f and l denote the frame size and preamble size of a computation task offloaded to the UAV, respectively. We have $f > l$, as a frame must be longer than the preamble.

Therefore, maximizing the number of computation tasks offloaded to the UAV can be formulated as:

$$\textbf{OPT:} \max_{i_t,(x_t,y_t,z),(v_t^x,v_t^y)} \quad i_t\Lambda_i(t) \tag{14}$$

$$s.\ t.: \quad (2),(6)$$

$$1 \leq i \leq N \tag{15}$$

$$0 \leq i_t \leq 1 \tag{16}$$

$$1 \leq t \leq T \tag{17}$$

$$q_i(t) \leq \mathcal{Q} \tag{18}$$

where $i_t = 1$ indicates that device i is selected for task offloading at time t; otherwise, $i_t = 0$. Constraint $q_i(t) \leq \mathcal{Q}$ indicates that the number of generated computation tasks at device i has to be smaller than the threshold $\mathcal{Q}$.

V. Combinatorial GNN-enabled Deep Reinforcement Learning

This section investigates the combinatorial learning architecture of GNN-enabled DRL. In the proposed structure, GNN extracts features and hidden connections of the network states for future state prediction. A2C utilizes the learning outcomes of GNN to train the action of the cruise control and task offloading.

A. Extracting feature correlation with GNN

To optimize jointly the cruise control and task offloading in practice (i.e., the **OPT** problem formulated by (14) $\sim$ (18)), the UAV edge server needs to be aware of the a-priori knowledge of all the IoT devices, e.g., $q_i(t)$ and $H_i(t)$, across the entire time horizon. Moreover, collecting the real-time information of $q_i(t)$ and $H_i(t)$ during the flight can result in large communication overheads between the UAV and the IoT devices. Therefore, the **OPT** problem could only be optimized offline. To achieve optimal cruise control and task offloading online, we put forth the new combinatorial learning architecture, where GNN predicts the network states via modeling the dependencies between the states.

A network state consists of the battery levels of the IoT devices and the UAV at time t, i.e., $b_i(t)$ and b_t^{UAV}, the number of generated tasks at the IoT device $q_i(t)$, the channel condition $H_i(t)$, the location of the UAV, and the number of tasks of device i being offloaded to the UAV till time t, i.e., $D_i(t)$. Therefore, the network state, denoted by S_α, is given by

$$S_\alpha = \{b_i, q_i, H_i, D_i, b^{\mathrm{UAV}}, (x,y,z)\}, \quad \forall i \in [1,N]. \tag{19}$$

The UAV edge server takes actions along the trajectory to control its heading and speed, while the scheduled IoT devices offload the computation tasks to the UAV. The action is given by

$$a_{S_\alpha} = \Big\{ \Big(x(S_\beta), y(S_\beta), z\Big), \Big(v_x(S_\alpha), v_y(S_\alpha)\Big), i_{S_\alpha} \Big\}, \tag{20}$$

where $(x(S_\beta), y(S_\beta), z)$ is the next location of the UAV. $(v_x(S_\alpha), v_y(S_\alpha))$ is the projection of the speed of the UAV on x- and y- axes. i_{S_α} indicates the selected IoT device at state S_α. We define an action space $\mathcal{A}$ which consists of all the actions that the UAV can take to optimize the cruise control and task offloading.

The correlation between the network states can be described by a graph $\mathcal{G}(\mathcal{V},\mathcal{E};\mathbf{w})$, where a vertex $\mathcal{V}$ in the graph represents one of the network states S_α and the presence of an edge $\mathcal{E}$ encodes the fact that the network states correlate with each other. The vertex space is denoted by $\mathcal{V}_\mathcal{G}$. $\mathbf{w}$ collects the weights of the edges in the graph.

The inputs of $\mathcal{G}(\mathcal{V},\mathcal{E};\mathbf{w})$ contain the vertexes and edges of the graph, and the vertex representation matrix. Moreover, the GNN consists of many input, output, and hidden layers. Let K denote the total number of layers that are composed in $\mathcal{G}(\mathcal{V},\mathcal{E};\mathbf{w})$. In the k-th layer, $w_\mathcal{V}^k$ is a learnable vector of the weights of $\mathcal{V}$'s edges, where $w_\mathcal{V}^k \subset \mathbf{w}$. The hidden state of the vertex $\mathcal{V}$ can be given by

$$\mathbf{f}_\mathcal{V}^k = \Phi^k\Big(\mathbf{f}_\mathcal{V}^{k-1} \oplus agt^k\Big(\{\mathbf{f}_{\mathcal{V}',\mathcal{E}}^{k-1} : (\mathcal{V},\mathcal{V}') \in \mathcal{E}^k\}_{\mathcal{E}^k \in \mathbb{R}^\mathcal{E}}\Big); w_\mathcal{V}^k\Big), \tag{21}$$

where $\oplus$ denotes the embedding summation operation; $\Phi^k(\cdot)$ is a nonlinear activation function (e.g., $\tanh(\cdot)$ or $ReLU(\cdot)$); $\mathbf{f}_\mathcal{V}$, $\mathbf{f}_\mathcal{E}$ and $\mathbf{f}_{\mathcal{V}'}$ are the representations of the vertex $\mathcal{V}$, edge $\mathcal{E}$, and neighbors of $\mathcal{V}$, respectively; $\mathcal{E}^k$ collects the edges at the k-th layer; $\mathbb{R}^\mathcal{E}$ denotes the hidden state dimension; and $agt^k(\cdot)$ is the aggregation function at the k-th layer, which maps the neighborhood information from different relations into a vector, e.g., mean aggregation and attention aggregation. $\mathbf{f}_\mathcal{V}^k$ can be initialized by $\mathbf{f}_\mathcal{V}^0 = \mathcal{V}$.

Based on (21), the graph generation loss, denoted by $\mathcal{L}_\mathcal{G}^k$, can be given by

$$\mathcal{L}_\mathcal{G}^k = \sum_{\mathcal{V}\in\mathcal{V}_\mathcal{G}} -\log(\Phi^k(\Omega(\mathbf{f}_\mathcal{V}^k))), \tag{22}$$

where Ω is a multilayer perceptron (with $\tanh(\cdot)$ in our work). The input of Ω at the k-th layer is the node embedding at the previous layer. The output is a scalar which is then fed into a nonlinear activation function $\Phi^k(\cdot)$.

B. GNN-A2C for joint cruise control and task offloading

Figure 2 demonstrates the proposed GNN-A2C combinatorial learning structure, where network states are pretrained in GNN to predict the future states for training the UAV's actions in A2C, i.e., a_{S_α} in (20).

The reward of the A2C training, denoted by $R\{S_\beta|S_\alpha^\mathcal{G}, a_{S_\alpha^\mathcal{G}}\}$, measures the number of offloaded tasks to the UAV, as given by

$$R\{S_\beta|S_\alpha^\mathcal{G}, a_{S_\alpha^\mathcal{G}}\} = \min_{\forall i_{S_\alpha^\mathcal{G}} \in [1,N]} \Lambda_{i_{S_\alpha^\mathcal{G}}}(S_\alpha^\mathcal{G}), \tag{23}$$

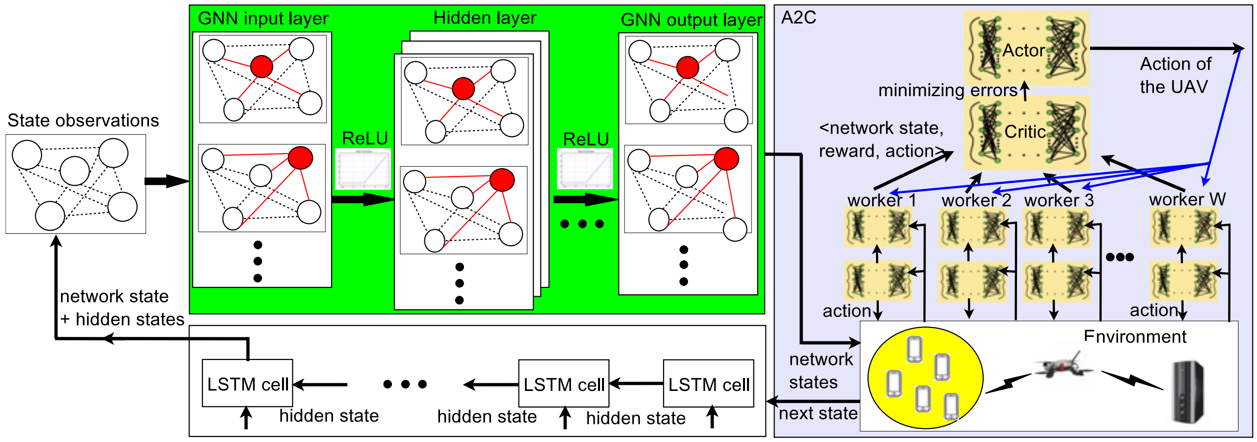


Fig. 2: The combinatorial learning architecture based on GNN and A2C for aerial EdgeIoT.

where $S_\alpha^{\mathcal{G}}$ denotes the network state trained by the GNN. The action $a_{S_\alpha^{\mathcal{G}}}$ is carried out by the UAV and the network state transits from $S_\alpha^{\mathcal{G}}$ to S_β. In particular, the channel condition $H_i(t)$ in the network state $S_\alpha^{\mathcal{G}}$ indicates the required bandwidth of the selected IoT device, which can be affected by the transmission delay and jitter. According to the spatial and temporal channel conditions, GNN-A2C trains the cruise and speed of the UAV while selecting the IoT devices to maximize $R\{S_\beta|S_\alpha^{\mathcal{G}}, a_{S_\alpha^{\mathcal{G}}}\}$.

The actor neural network in the GNN-A2C approximates cruise control and task offloading rules and prioritizes data streams at the network state. In contrast to widely adopted value-based approaches, e.g., [36], GNN-A2C is a policy-based model-free structure, which directly parameterizes the policy $\pi\{S_\beta|S_\alpha^{\mathcal{G}}, a_{S_\alpha^{\mathcal{G}}};\vartheta\}$. ϑ is updated by gradient ascents to approximate the expected reward $\mathbb{E}_\vartheta[\cdot]$. Therefore, the actor can be updated by the policy gradient, i.e.,

$$\nabla\Gamma(\vartheta) = \mathbb{E}_\vartheta[\nabla_\vartheta \log \pi\{S_\beta|S_\alpha^{\mathcal{G}}, a_{S_\alpha^{\mathcal{G}}};\vartheta\}A(S_\alpha^{\mathcal{G}}, a_{S_\alpha^{\mathcal{G}}})], \quad (24)$$

where $A(S_\alpha^{\mathcal{G}}, a_{S_\alpha^{\mathcal{G}}})$ denotes the advantage function. The advantage function measures the difference between the action-value function $Q\{S_\beta|S_\alpha^{\mathcal{G}}, a_{S_\alpha^{\mathcal{G}}}\}$ and the state-value function $V(S_\alpha^{\mathcal{G}})$ [37]. The advantage function can be estimated by the temporal difference (TD) error, as given by

$$\begin{aligned} A(S_\alpha^{\mathcal{G}}, a_{S_\alpha^{\mathcal{G}}}) &= Q\{S_\beta|S_\alpha^{\mathcal{G}}, a_{S_\alpha^{\mathcal{G}}}\} - V(S_\alpha^{\mathcal{G}}) \\ &\approx R\{S_\beta|S_\alpha^{\mathcal{G}}, a_{S_\alpha^{\mathcal{G}}}\} + \mu V(S_\beta|S_\alpha^{\mathcal{G}}, a_{S_\alpha^{\mathcal{G}}}) - V(S_\alpha^{\mathcal{G}}), \end{aligned} \quad (25)$$

where μ is a discount value ($0 < \mu < 1$).

The critic neural network in the GNN-A2C estimates the action-value function to evaluate the actions of the actor at each state. The loss function of the critic neural network, denoted by Γ_{loss}^c, defines the estimation error, as given by

$$\Gamma_{\text{loss}}^c = A(S_\alpha^{\mathcal{G}}, a_{S_\alpha^{\mathcal{G}}})^2. \quad (26)$$

At the updating step, the least squares TD is minimized to update the critic at each training step, where the state-value function is optimized by

$$a_{S_\alpha^{\mathcal{G}}}^* = \arg\min \Gamma_{\text{loss}}^c. \quad (27)$$

Algorithm 1 GNN-A2C for joint cruise control and task offloading in aerial EdgeIoT

```
1. Initialize: $\mathcal{G}(\mathcal{V}, \mathcal{E}; \mathbf{w})$, $\mathcal{V}_\mathcal{G}$, $S_\alpha$, $a_{S_\alpha}$, $t_{\text{learning}}$, $\mathcal{M}_{\text{replay}}$.
for episode 1 to $ep$ do
    Map the network states $S_\alpha$ to the vertexes $\mathcal{V}$ of the graph.
    GNN:
    for Vertex $\mathcal{V} \in \mathcal{V}_\mathcal{G}$ do
        for $k$ = 1 to $K$ do
            For each vertex, $\mathbf{f}_\mathcal{V}^k \leftarrow$ Eq. (21).
            Calculate the graph generation loss, $\mathcal{L}_\mathcal{G}^k \leftarrow$ Eq. (22).
            Optimize $w_\mathcal{V}^k \in \mathbf{w}'$ to minimize $\mathcal{L}_\mathcal{G}^k$.
        end for
    end for
    Obtain $S_\alpha^\mathcal{G}$ according to the optimized graph $\mathcal{G}'(\mathcal{V}, \mathcal{E}; \mathbf{w}')$.
    Store the experience $< S_\alpha^\mathcal{G}, S_\beta, a_{S_\alpha}, R\{S_\beta|S_\alpha^\mathcal{G}, a_{S_\alpha}\} >$ to $\mathcal{M}_{\text{replay}}$.
    A2C:
    Sample random minibatches from $\mathcal{M}_{\text{replay}}$.
    $S_\alpha^\mathcal{G} \rightarrow$ the critic neural network, and calculate the state-value function $V(S_\alpha^\mathcal{G})$.
    The critic $\leftarrow R\{S_\beta|S_\alpha^\mathcal{G}, a_{S_\alpha^\mathcal{G}}\}$.
    $A(S_\alpha^\mathcal{G}, a_{S_\alpha^\mathcal{G}})$ is updated by (25).
    $S_\alpha^\mathcal{G}$ and $S_\beta \rightarrow$ the actor neural network.
    The policy gradient $\nabla\Gamma(\vartheta)$ is updated by (24). The actor is trained while the critic evaluates (27) $\rightarrow a_{S_\alpha^\mathcal{G}}^*$.
    The UAV carries out the action $a_{S_\alpha^\mathcal{G}}^*$ in the environment.
    According to the training environment, the state observation $S_\alpha$ is renewed for GNN.
end for
```

To implement the proposed GNN-A2C structure, Algorithm 1 is developed for the new joint UAV cruise control and task offloading in aerial EdgeIoT.

VI. Implementation and Performance Evaluation

A. Implementation of GNN-A2C

We implement the proposed GNN-A2C in Python 3.9 with Google Tensorflow, set up on a Linux workstation with 64-bit Ubuntu 18.04. The hardware for training the GNN-A2C has two Nvidia's GPUs, one is GeForce GTX 1060 with 3 GB memory, and the other is GeForce RTX 2060 with 6 GB memory. The IoT devices are randomly distributed in a target area of 1000 × 1000 meters. The initial position of the UAV is at (600, 500) m. The highest flying speed is 15 m/s. The maximum transmit power of the IoT device is 100 milliwatts, and the battery energy budget has 800 Joules.

The maximum number of iterations in GNN is set to 50, and the graph $\mathcal{G}(\mathcal{V}, \mathcal{E}; \mathbf{w})$ is trained in 1000 epochs for extracting the correlated features of the network states. In each iteration, the GNN is evaluated by a validation function every 20 epochs. The validation function provides the loss value of the training and the number of iterations at the convergence. The graph model that achieves the lowest cost is considered as the best $\mathcal{G}(\mathcal{V}, \mathcal{E}; \mathbf{w})$, and is used to evaluate the testing data.

The A2C module contains two hidden layers separated from the same input tensor. Each hidden layer has 128 units. Policy gradients, weighted by advantages, define the policy loss. The actions of the UAV is trained in the A2C for 1500 steps, and the discount factor is 0.99. The loss function is calculated as the mean square error between the value estimates and returns, where the value function coefficient for the loss calculation is set to 0.5.

B. Performance analysis

1) Training of the GNN-A2C framework: Figure 3 shows that the TMR of the proposed GNN-A2C framework gradually declines with the growing number of training episodes. This is because the GNN module works with LSTM to extract the feature correlation of the network states and predict the hidden network states, enriching the training environment of the A2C module. Hence, the action of the UAV is sufficiently trained by more state observations S_α with the growing number of episodes. This leads to the fast convergence of the proposed GNN-A2C framework, where the TMR converges in about 150 episodes.

Given 200 nodes and that 10% of the IoT devices generate new tasks in each time step (i.e., $\Delta q(t) = 0.1$), the TMR drops from 20% to 5% when the number of training iterations increases from 50 to 1500. Moreover, increasing the number of the IoT devices that generate new computation tasks from 10% to 20% can slightly raise the TMR of GNN-A2C from 5% to 12%. This is due to the limited cruising speed of the UAV. Consequently, the IoT devices far away from the UAV may not get served in time.

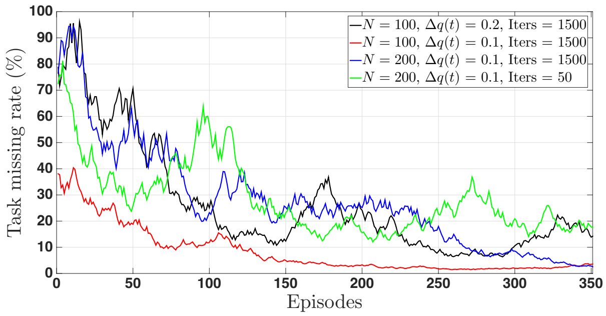


Fig. 3: TMR in terms of the training episodes.

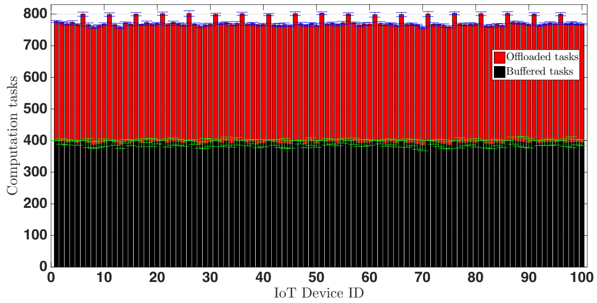


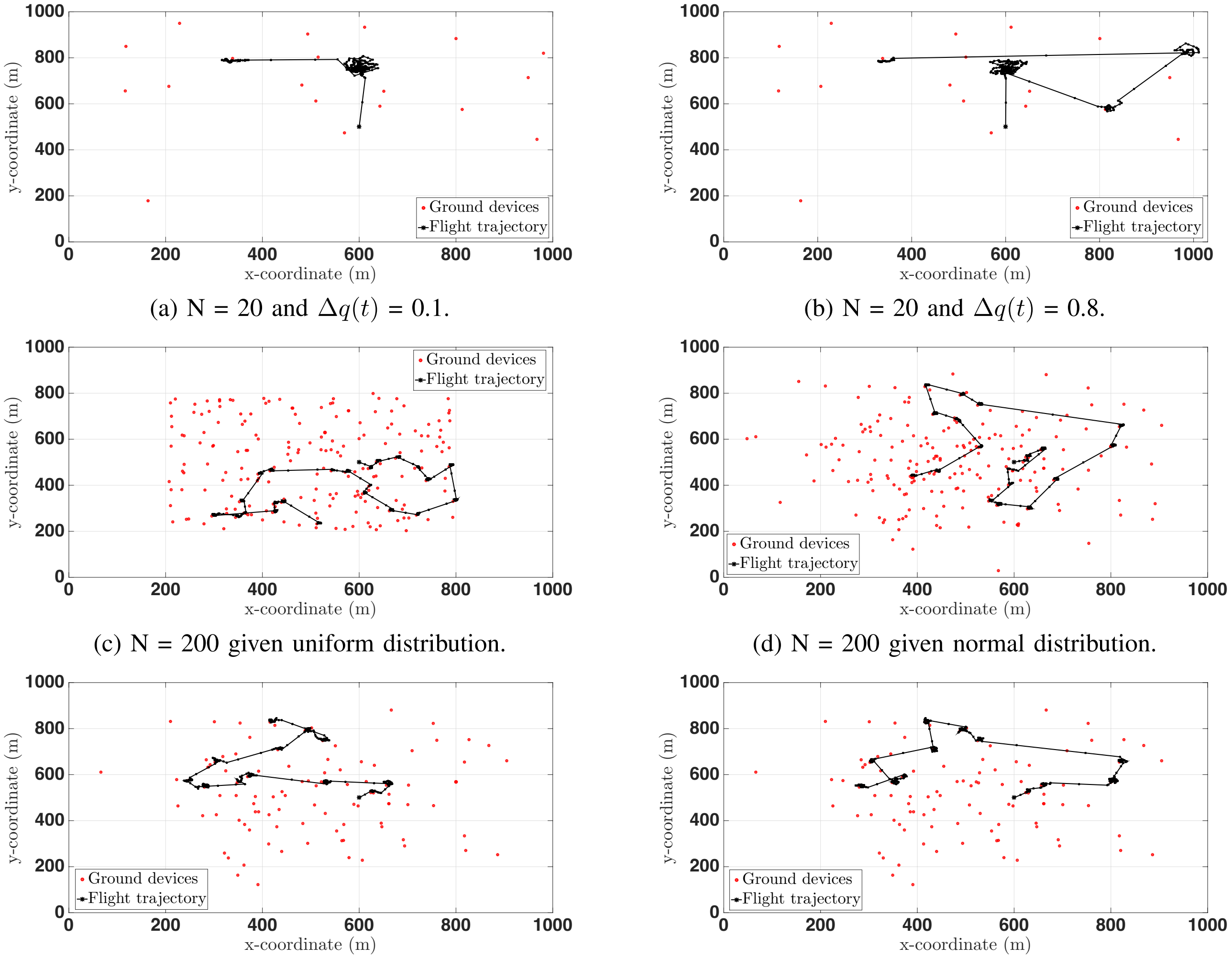
Fig. 4: Number of offloaded tasks and buffered tasks at each IoT device, where N = 100 and $\mathcal{Q}$ = 400. Each error bar provides the standard deviation of 10 experiments.

Figure 4 shows the number of offloaded tasks and buffered tasks at each IoT device, where N = 100 and $\mathcal{Q}$ = 400. Each error bar is calculated with the standard deviation over ten experiments. Thanks to the feature correlation extraction of the GNN-A2C, the UAV's flight speed and heading are trained with prior knowledge of the network state dynamics. This confirms that the sufficient training of the UAV's cruise control enables all the IoT devices to timely offload their tasks to minimize the missing computation tasks.

2) Continuous cruise control: Figure 5 presents the flight trajectories of the UAV controlled by the proposed GNN-A2C framework, given the different number of IoT devices, $\Delta q(t)$ values, distribution patterns of the IoT devices, and learning iterations. Specifically, we can observe in Figures 5(a) and 5(b) that the UAV extends its flight coverage when $\Delta q(t)$ increases from 0.1 to 0.8. This is reasonable since a higher $\Delta q(t)$ value leads to more IoT devices producing computation tasks. Thus, the UAV adapts its cruise to approach the IoT devices to minimize the TMR.

In Figure 5(c), the IoT devices are uniformly deployed in a 600 m × 600 m square area. In Figure 5(d), the deployment follows the normal distribution, where the devices are dispersed in a 1000 m × 1000 m field. The performance validates that GNN-A2C effectively trains the cruise control of the UAV and the task offloading allocation according to the density of the task generation.

Furthermore, Figures 5(e) and 5(f) show the flight trajectories of the UAV when the training iterations of the GNN-A2C increase from 1500 to 3000. Once the GNN-A2C is sufficiently trained, the UAV alters its course to serve the devices with the full task buffers or high channel quality to minimize the TMR. This also cross-validates the performance in Figure 3.



(a) N = 20 and $\Delta q(t)$ = 0.1.

(b) N = 20 and $\Delta q(t)$ = 0.8.

(c) N = 200 given uniform distribution.

(d) N = 200 given normal distribution.

(e) N = 100 and training iterations = 1500.

(f) N = 100 and training iterations = 3000.

Fig. 5: Flight trajectories of the UAV, which is controlled by the proposed GNN-A2C framework, given different number of IoT devices, $\Delta q(t)$ values, distribution patterns of the IoT devices, and training iterations.

3) Number of IoT devices and their buffer capability: Figure 6 shows the TMR in terms of different aerial EdgeIoT's scalability. We compare the proposed GNN-A2C framework with DQN- and DDPG-based trajectory planning strategies [38], [39], a pre-planned trajectory and random device selection (PTRS) strategy [40], and a channel-guided cruise control (CGC) strategy [41]. In general, the TMR grows with the number of IoT devices. When $N = 600$, GNN-A2C outperforms DDPG, DQN, CGC, and PTRS, by about 16.7%, 43.5%, 63.9%, and 64.1%, respectively. This is because PTRS applies a predetermined flight trajectory of the UAV while the IoT devices are randomly selected to offload the computation tasks. Therefore, PTRS provides the worst. Although the DDPG enables the training of the continuous actions (compared to the DQN-based strategy), the hidden states are hardly explored and predicted, resulting in insufficient action training.

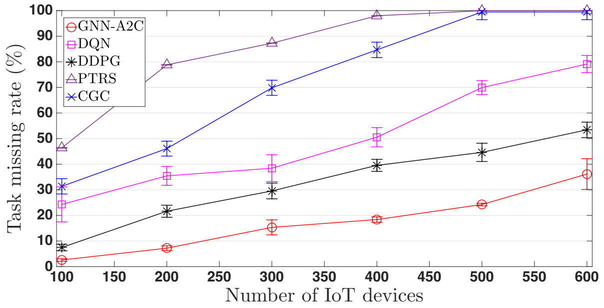


Fig. 6: TMR vs. the aerial EdgeIoT's scalability N, where $\mathcal{Q}$ is 100. Each error bar provides the standard deviation of 10 independent experiments.

Different from the existing solutions, the proposed GNN-A2C framework extracts the features of the network state observation. In contrast, the hidden states are predicted, which guides the actions in A2C to be sufficiently trained. Furthermore, GNN-A2C trains the actions of the UAV in a continuous domain, which optimally controls the real-time flight speed and heading of the UAV.

Figure 7 studies the TMR in regards to the maximum

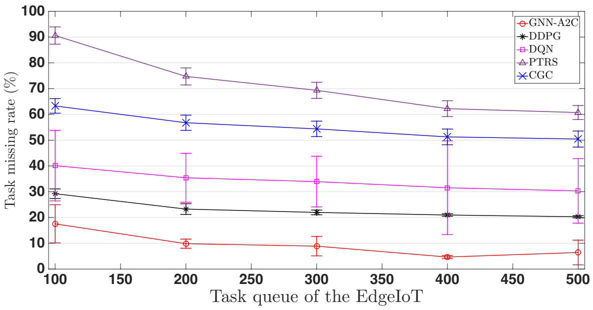


Fig. 7: TMR vs. $\mathcal{Q}$, where $N = 300$. Each error bar shows the standard deviation of 10 independent experiments.

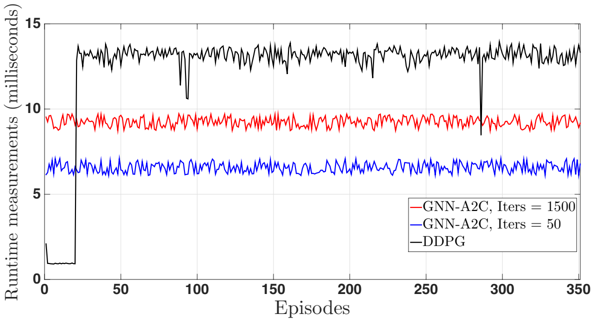


Fig. 8: Runtime measurement of the GNN-A2C, where the number of training iterations is 50 or 1500.

number of tasks that can be buffered at the IoT devices. In general, increasing the buffer sizes of the IoT devices reduces the TMR, since more generated computation tasks can be kept at an IoT device until the UAV visits the device. When $\mathcal{Q} = 500$, the TMRs of the proposed GNN-A2C are 13.2%, 23.6%, 43.3% and 54.8% lower than the strategies based on DDPG, DQN, CGC and PTRS, respectively. The reason is that the GNN enables the training of the cruise control and task offloading with observed network states and hidden states in the continuous action space.

4) Runtime of the GNN-A2C framework: In Figure 8, the runtime of the GNN-A2C is measured after 50 or 1500 training iterations in comparison to the DDPG-based strategy. Particularly, the training time of the GNN-A2C with 1500 iterations is around 9 ms in each episode. When the number of training iterations decreases to 50, the training time drops to about 7 ms. Moreover, the DDPG-based strategy consumes 23 episodes to initialize its experience replay, consuming about 1.2 ms. Once the DDPG-based strategy starts the training, the runtime grows to 13 ms at each episode, which is slower than the GNN-A2C framework for at least 4 ms. The DDPG-based strategy trains the action with a random process for the action exploration, which takes extra time. In contrast, the GNN-A2C predicts the future network states for training the UAV's actions in A2C at each episode, saving time on the random action exploration of the DDPG-based strategy.

VII. Conclusion

In this paper, a new joint optimization of UAV cruise control and task offloading allocation in aerial EdgeIoT was proposed to maximize the number of computation tasks offloaded to the UAV. Since the optimization contains a large solution space while the instantaneous network states are unknown to the UAV, a new deep graph-based DRL framework, i.e., GNN-A2C, was developed to take advantage of the GNN to supervise the training of UAV's actions in A2C, which explores the hidden network states based on the network feature correlation. Numerical studies showed that the GNN-A2C significantly reduces the task missing rate in aerial EdgeIoT while effectively controlling the flight cruise to process the computation tasks.

Acknowledgements

This work was supported in part by the National Funds through FCT/MCTES (Portuguese Foundation for Science and Technology), within the CISTER Research Unit under Grant UIDP/UIDB/04234/2020, and in part by national funds through FCT, within project PTDC/EEICOM/3362/2021 (ADANET).

References

[1] G. Premsankar, M. Di Francesco, and T. Taleb, "Edge computing for the internet of things: A case study," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1275–1284, 2018.

[2] P. Corcoran and S. K. Datta, "Mobile-edge computing and the internet of things for consumers: Extending cloud computing and services to the edge of the network," *IEEE Consumer Electronics Magazine*, vol. 5, no. 4, pp. 73–74, 2016.

[3] J. Ren, Y. He, G. Huang, G. Yu, Y. Cai, and Z. Zhang, "An edge-computing based architecture for mobile augmented reality," *IEEE Network*, vol. 33, no. 4, pp. 162–169, 2019.

[4] L. Wang, L. Jiao, T. He, J. Li, and M. Mühlhäuser, "Service entity placement for social virtual reality applications in edge computing," in *IEEE INFOCOM*. IEEE, 2018, pp. 468–476.

[5] R. S. Alonso, I. Sittón-Candanedo, Ó. García, J. Prieto, and S. Rodríguez-González, "An intelligent edge-IoT platform for monitoring livestock and crops in a dairy farming scenario," *Ad Hoc Networks*, vol. 98, p. 102047, 2020.

[6] K. Li, C. Yuen, B. Kusy, R. Jurdak, A. Ignjatovic, S. S. Kanhere, and S. Jha, "Fair scheduling for data collection in mobile sensor networks with energy harvesting," *IEEE Transactions on Mobile Computing*, vol. 18, no. 6, pp. 1274–1287, 2018.

[7] K. Li, W. Ni, M. Abolhasan, and E. Tovar, "Reinforcement learning for scheduling wireless powered sensor communications," *IEEE Transactions on Green Communications and Networking*, vol. 3, no. 2, pp. 264–274, 2018.

[8] Z. Zhou, J. Feng, L. Tan, Y. He, and J. Gong, "An air-ground integration approach for mobile edge computing in IoT," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 40–47, 2018.

[9] J. Feng, L. Zhao, J. Du, X. Chu, and F. R. Yu, "Computation offloading and resource allocation in D2D-enabled mobile edge computing," in *IEEE International Conference on Communications (ICC)*. IEEE, 2018, pp. 1–6.

[10] K. Li, W. Ni, E. Tovar, and A. Jamalipour, "Online velocity control and data capture of drones for the internet of things: An onboard deep reinforcement learning approach," *IEEE Vehicular Technology Magazine*, vol. 16, no. 1, pp. 49–56, 2020.

[11] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 1, pp. 4–24, 2020.

[12] J. Wang, K. Liu, and J. Pan, "Online UAV-mounted edge server dispatching for mobile-to-mobile edge computing," *IEEE Internet of Things Journal*, vol. 7, no. 2, pp. 1375–1386, 2019.

[13] M. Hua, Y. Wang, C. Li, Y. Huang, and L. Yang, “UAV-aided mobile edge computing systems with one by one access scheme,” *IEEE Transactions on Green Communications and Networking*, vol. 3, no. 3, pp. 664–678, 2019.

[14] Q. Hu, Y. Cai, G. Yu, Z. Qin, M. Zhao, and G. Y. Li, “Joint offloading and trajectory design for UAV-enabled mobile edge computing systems,” *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1879–1892, 2018.

[15] Q. Zhang, J. Chen, L. Ji, Z. Feng, Z. Han, and Z. Chen, “Response delay optimization in mobile edge computing enabled UAV swarm,” *IEEE Transactions on Vehicular Technology*, vol. 69, no. 3, pp. 3280–3295, 2020.

[16] X. Cao, J. Xu, and R. Zhang, “Mobile edge computing for cellular-connected UAV: Computation offloading and trajectory optimization,” in *IEEE International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*. IEEE, 2018, pp. 1–5.

[17] C. Zhan, H. Hu, X. Sui, Z. Liu, and D. Niyato, “Completion time and energy optimization in the UAV-enabled mobile-edge computing system,” *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7808–7822, 2020.

[18] T. Zhang, Y. Xu, J. Loo, D. Yang, and L. Xiao, “Joint computation and communication design for UAV-assisted mobile edge computing in IoT,” *IEEE Transactions on Industrial Informatics*, vol. 16, no. 8, pp. 5505–5516, 2019.

[19] Y. K. Tun, Y. M. Park, N. H. Tran, W. Saad, S. R. Pandey, and C. S. Hong, “Energy-efficient resource management in uav-assisted mobile edge computing,” *IEEE Communications Letters*, vol. 25, no. 1, pp. 249–253, 2020.

[20] M. Li, N. Cheng, J. Gao, Y. Wang, L. Zhao, and X. Shen, “Energy-efficient UAV-assisted mobile edge computing: Resource allocation and trajectory optimization,” *IEEE Transactions on Vehicular Technology*, vol. 69, no. 3, pp. 3424–3438, 2020.

[21] Z. Yu, Y. Gong, S. Gong, and Y. Guo, “Joint task offloading and resource allocation in UAV-enabled mobile edge computing,” *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3147–3159, 2020.

[22] X. Diao, J. Zheng, Y. Cai, Y. Wu, and A. Anpalagan, “Fair data allocation and trajectory optimization for UAV-assisted mobile edge computing,” *IEEE Communications Letters*, vol. 23, no. 12, pp. 2357–2361, 2019.

[23] Q. Liu, L. Shi, L. Sun, J. Li, M. Ding, and F. Shu, “Path planning for UAV-mounted mobile edge computing with deep reinforcement learning,” *IEEE Transactions on Vehicular Technology*, vol. 69, no. 5, pp. 5723–5728, 2020.

[24] L. Zhang, Z.-Y. Zhang, L. Min, C. Tang, H.-Y. Zhang, Y.-H. Wang, and P. Cai, “Task offloading and trajectory control for UAV-assisted mobile edge computing using deep reinforcement learning,” *IEEE Access*, vol. 9, pp. 53 708–53 719, 2021.

[25] K. Li, W. Ni, and F. Dressler, “LSTM-characterized deep reinforcement learning for continuous flight control and resource allocation in UAV-assisted sensor network,” *IEEE Internet of Things Journal*, 2021.

[26] K. Li, W. Ni, E. Tovar, and A. Jamalipour, “On-board deep q-network for UAV-assisted online power transfer and data collection,” *IEEE Transactions on Vehicular Technology*, vol. 68, no. 12, pp. 12 215–12 226, 2019.

[27] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, “Graph neural networks: A review of methods and applications,” *AI Open*, vol. 1, pp. 57–81, 2020.

[28] W. Liu, C. Yu, X. Wang, Y. Zhang, and Y. Yu, “The altitude hold algorithm of UAV based on millimeter wave radar sensors,” in *IEEE International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, vol. 1. IEEE, 2017, pp. 436–439.

[29] K. Li, Y. Emami, W. Ni, E. Tovar, and Z. Han, “Onboard deep deterministic policy gradients for online flight resource allocation of UAVs,” *IEEE Networking Letters*, vol. 2, no. 3, pp. 106–110, 2020.

[30] H. Huang, A. V. Savkin, and W. Ni, “Energy-efficient 3D navigation of a solar-powered UAV for secure communication in the presence of eavesdroppers and no-fly zones,” *Energies*, vol. 13, no. 6, p. 1445, 2020.

[31] M. Hua, Y. Wang, Q. Wu, H. Dai, Y. Huang, and L. Yang, “Energy-efficient cooperative secure transmission in multi-UAV-enabled wireless networks,” *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, pp. 7761–7775, 2019.

[32] Y. Emami, B. Wei, K. Li, W. Ni, and E. Tovar, “Joint communication scheduling and velocity control in multi-UAV-assisted sensor networks: A deep reinforcement learning approach,” *IEEE Transactions on Vehicular Technology*, vol. 70, no. 10, pp. 10 986–10 998, 2021.

[33] K. Li, W. Ni, and F. Dressler, “Continuous maneuver control and data capture scheduling of autonomous drone in wireless sensor networks,” *IEEE Transactions on Mobile Computing*, 2021.

[34] J. Tang, G. Chen, and J. P. Coon, “Secrecy performance analysis of wireless communications in the presence of UAV jammer and randomly located UAV eavesdroppers,” *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 11, pp. 3026–3041, 2019.

[35] K. Li, R. C. Voicu, S. S. Kanhere, W. Ni, and E. Tovar, “Energy efficient legitimate wireless surveillance of UAV communications,” *IEEE Transactions on Vehicular Technology*, vol. 68, no. 3, pp. 2283–2293, 2019.

[36] O. Nachum, M. Norouzi, K. Xu, and D. Schuurmans, “Bridging the gap between value and policy based reinforcement learning,” *Advances in neural information processing systems*, vol. 30, 2017.

[37] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, “Asynchronous methods for deep reinforcement learning,” in *International conference on machine learning*. PMLR, 2016, pp. 1928–1937.

[38] K. Li, W. Ni, E. Tovar, and A. Jamalipour, “Deep Q-learning based resource management in UAV-assisted wireless powered IoT networks,” in *IEEE International Conference on Communications (ICC)*. IEEE, 2020, pp. 1–6.

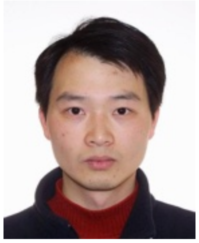
[39] H. Kurunathan, K. Li, W. Ni, E. Tovar, and F. Dressler, “Deep reinforcement learning for persistent cruise control in UAV-aided data collection,” in *IEEE Conference on Local Computer Networks (LCN)*. IEEE, 2021, pp. 347–350.

[40] P. Sun and A. Boukerche, “Performance modeling and analysis of a UAV path planning and target detection in a UAV-based wireless sensor network,” *Computer Networks*, vol. 146, pp. 217–231, 2018.

[41] H. Oh, H.-S. Shin, S. Kim, and W.-H. Chen, “Communication-aware trajectory planning for unmanned aerial vehicles in urban environments,” *Journal of Guidance, Control, and Dynamics*, vol. 41, no. 10, pp. 2271–2282, 2018.

Kai Li (S’09–M’14–SM’20) received the B.E. degree from Shandong University, China, in 2009, the M.S. degree from The Hong Kong University of Science and Technology, Hong Kong, in 2010, and the Ph.D. degree in computer science from The University of New South Wales, Sydney, NSW, Australia, in 2014.

Dr Li is currently a Senior Research Scientist with CISTER, Porto, Portugal. He is a CMU-Portugal Research Fellow, which is jointly supported by Carnegie Mellon University, Pittsburgh, PA, USA, and FCT, Lisbon, Portugal. Prior to this, he was a Postdoctoral Research Fellow with the SUTD-MIT International Design Centre, The Singapore University of Technology and Design, Singapore, from 2014 to 2016. He was a Visiting Research Assistant with ICT Centre, CSIRO, Australia, from 2012 to 2013. From 2010 to 2011, he was a Research Assistant with Mobile Technologies Centre, The Chinese University of Hong Kong, Hong Kong. His research interests include machine learning, vehicular communications and security, resource allocation optimization, cyber–physical systems, Internet of Things, and UAV networks.



Wei Ni (M'09–SM'15) received the B.E. and Ph.D. degrees in Electronic Engineering from Fudan University, Shanghai, China, in 2000 and 2005, respectively. Currently, he is a Principal Research Scientist at CSIRO, Sydney, Australia, and an Adjunct Professor at the University of Technology Sydney and Honorary Professor at Macquarie University. He was a Postdoctoral Research Fellow at Shanghai Jiaotong University from 2005 to 2008; Deputy Project Manager at the Bell Labs, Alcatel/Alcatel-Lucent from 2005 to 2008; and Senior Researcher at Devices R&D, Nokia from 2008 to 2009. He has authored five book chapters, more than 200 journal papers, more than 80 conference papers, 25 patents, and ten standard proposals accepted by IEEE. His research interests include machine learning, online learning, stochastic optimization, as well as their applications to system efficiency and integrity.

Dr Ni is the Chair of IEEE Vehicular Technology Society (VTS) New South Wales (NSW) Chapter since 2020, an Editor of IEEE Transactions on Wireless Communications since 2018, and an Editor of IEEE Transactions on Vehicular Technology. He served first the Secretary and then the Vice-Chair of IEEE NSW VTS Chapter from 2015 to 2019, Track Chair for VTC-Spring 2017, Track Co-chair for IEEE VTC-Spring 2016, Publication Chair for BodyNet 2015, and Student Travel Grant Chair for WPMC 2014.

Xin Yuan (S'16–M'19) received the B.E. degree from Taiyuan University of Technology, Shanxi, China, in 2013, and the dual Ph.D. degree from Beijing University of Posts and Telecommunications (BUPT), Beijing, China, and the University of Technology Sydney (UTS), Sydney, Australia, in 2019 and 2020, respectively. She is currently a Research Scientist at CSIRO, Sydney, NSW, Australia. Her research interests include machine learning and optimization, and their applications to UAV networks and intelligent systems.

Alam Noor (S'21) is a Researcher at CISTER Research Center, Portugal, and a Ph.D. candidate at the University of Porto, Portugal. His research interests include pattern recognition, deep learning, and data processing for real-time applications.

Abbas Jamalipour (S'86–M'91–SM'00–F'07) received the Ph.D. degree in Electrical Engineering from Nagoya University, Nagoya, Japan in 1996. He holds the positions of Professor of Ubiquitous Mobile Networking with the University of Sydney and since January 2022, the Editor-in-Chief of the IEEE Transactions on Vehicular Technology. He has authored nine technical books, eleven book chapters, over 550 technical papers, and five patents, all in the area of wireless communications and networking. Prof. Jamalipour is a recipient of the number of prestigious awards, such as the 2019 IEEE ComSoc Distinguished Technical Achievement Award in Green Communications, the 2016 IEEE ComSoc Distinguished Technical Achievement Award in Communications Switching and Routing, the 2010 IEEE ComSoc Harold Sobol Award, the 2006 IEEE ComSoc Best Tutorial Paper Award, as well as over 15 Best Paper Awards. He was the President of the IEEE Vehicular Technology Society (2020-2021). Previously, he held the positions of the Executive Vice-President and the Editor-in-Chief of VTS Mobile World and has been an elected member of the Board of Governors of the IEEE Vehicular Technology Society since 2014. He was the Editor-in-Chief IEEE WIRELESS COMMUNICATIONS, the Vice President-Conferences, and a member of Board of Governors of the IEEE Communications Society. He sits on the Editorial Board of the IEEE ACCESS and several other journals and is a member of Advisory Board of IEEE Internet of Things Journal. He has been the General Chair or Technical Program Chair for several prestigious conferences, including IEEE ICC, GLOBECOM, WCNC, and PIMRC. He is a Fellow of the Institute of Electrical and Electronics Engineers (IEEE), the Institute of Electrical, Information, and Communication Engineers (IEICE), and the Institution of Engineers Australia, an ACM Professional Member, and an IEEE Distinguished Speaker.