Journal of Electrical Engineering, Vol. 75, No. 2, 2024, pp. 94-101

sciendo

PAPERS

Deep reinforcement learning based computing offloading in unmanned aerial vehicles for disaster management

Anuratha Kesavan[1*], Nandhini Jembu Mohanram[1], Soshya Joshi[2], Uma Sankar[3]

The emergence of Internet of Things enabled with mobile computing has the applications in the field of unmanned aerial vehicle (UAV) development. The development of mobile edge computational offloading in UAV is dependent on low latency applications such as disaster management, Forest fire control and remote operations. The task completion efficiency is improved by means of using edge intelligence algorithm and the optimal offloading policy is constructed on the application of deep reinforcement learning (DRL) in order to fulfill the target demand and to ease the transmission delay. The joint optimization curtails the weighted sum of average energy consumption and execution delay. This edge intelligence algorithm combined with DRL network exploits computing operation to increase the probability that at least one of the tracking and data transmission is usable. The proposed joint optimization significantly performs well in terms of execution delay, offloading cost and effective convergence over the prevailing methodologies proposed for UAV development. The proposed DRL enables the UAV to real-time decisions based on the disaster scenario and computing resources availability.

Keywords: deep reinforcement learning algorithm, edge intelligence, UAV energy consumption

1 Introduction

The emergence of computation offloading rises when the mobile applications are needed to track on remote servers and to consume energy. The timing necessity of offloading a task is a challenging one due to their execution time constraints [1]. Mobile edge server (MEC) should decide when to execute a task and when to offload a task in order to minimize the energy. UAV is embedded with the on-device camera and sensors to work for navigation, disaster management and IoT based agricultural applications. The quality of experience (QoE) has to be ensured between resource limited devices and MEC server [2]. MEC significantly reduces the latency by avoiding congestion between transmitted packets and prolonging the UAV battery lifetime for the QoE. The featuring tasks are computationally offloaded with the aid of deep reinforcement learning enabled with water strider optimization algorithm. There are various algorithms related in optimizing the task offloading such as dynamic partitioning and programming, Lyapunov optimization, Game theoretic approach and machine learning algorithms [3-7]. However, the problem is based upon execution time constraint. Firstly, the locations of UAV are adjusted as per the real-time offloading methodologies of users. Secondly the trajectory has to be well planned for energy consumption, maximum throughput. MEC enables UAV to strengthen their coverage since the channel impairments are the major problem to have line-of-sight links to the ground users [8-10]. The energy minimization is the major goal in [11-14] using linear and dynamic programming. Computation-intensive UAV enabled with MEC has the probable to be applied in forest fire monitoring, earth quake disaster management environment where the possibility of data collection is very difficult to handle [15]. The proposed work aims to handle the difficult environment to sense and collect the data for processing. The UAV data processing and analysis require the high performance of the computer, while it is difficult for the mobile terminal on site to meet this requirement. Therefore, we need to apply various algorithms to transmit data, establish a high-performance data processing center to promote the efficiency and effectiveness of data processing and analysis. It is necessary to keep the energy precious to prolong the lifetime of the entire network. The task computation is either executed or offloaded in order to keep the network alive in energy saving manner. The mobile cloud computing scenario is solved using semi definite approach in [16]. But in [17], a stochastic optimization problem combined with the fog computing and MEC is proposed for the task execution. A stochastic game method is proposed in [18] to reduce the energy cost function and to save the UAV's energy. Moreover, a time-consuming backtracking procedure was required to determine the final decisions. A dynamic

[1] Sri Sai Ram Institute of Technology, Chennai, India
[2] SRM Institute of Science and Technology, Chennai, India
[3] Panimalar Engineering College, Chennai, India
*anujournal381@gmail.com

https://doi.org/10.2478/jee-2024-0013, Print (till 2015) ISSN 1335-3632, On-line ISSN 1339-309X

offloading technique based on Lyapunov optimization was reported in [18].

From the above discussion, aerial-ground computational cooperation is required for task execution and UAV's coverage to be expanded in the midst of signal fading and other obstacles. Using overhead photos, firefighters were able to evaluate the situation and make plans for what to do next. To enable a single person to control the whole fleet, multi-UAV systems must be simple to use and effective. Instead of "steering" individual UAVs in this instance, the operator designates high-level duties on a digital map, such as regions to be watched and prohibited areas [20]. There are two metrics to be considered such as computation throughput and energy consumption.

- To optimize the UAV edge intelligence based on DRL cooperative methodology and to allocate minimum power constraint to each UAV.
- To formulate the UAV energy minimization problem as a Markov decision process to generate the maximum reward and to design edge intelligence algorithm.
- To compute low energy operation with computational resources of UAVs, DRL enabled MEC framework is proposed in the multi-UAV system for surveillance report.
- To compare the experimental results with prevailing methodologies refereed in previous research so as to enable the prominence of the proposed edge intelligence in UAV.

The paper is organized as the following manner. Chapter 2 details the MEC enabled UAV system, chapter 3 entitles about the MDP problem with Edge intelligence algorithm, chapter 4 discusses the experimental Multi-UAV results and discussion and chapter 5 details about the conclusion of the paper.

2 Methodology

MEC-enabled UAVs leverage edge servers to process data in real-time applications. In this framework, ground mobile Users (GU) receive computing services from many UAVs with restricted energy B for a predetermined amount of time. Using t = 0, 1, 2, ..., T−1, the operational period is discretized into T times slots, each having a non-uniform length. Assume that each time slot, or "association between the UAV and GU," can only have one GU served by the UAV. Only one of M fixed base stations (BS) may be hovered over by the UAV during each time slot in order to establish a direct link with the corresponding GU and carry out its offloaded responsibilities.

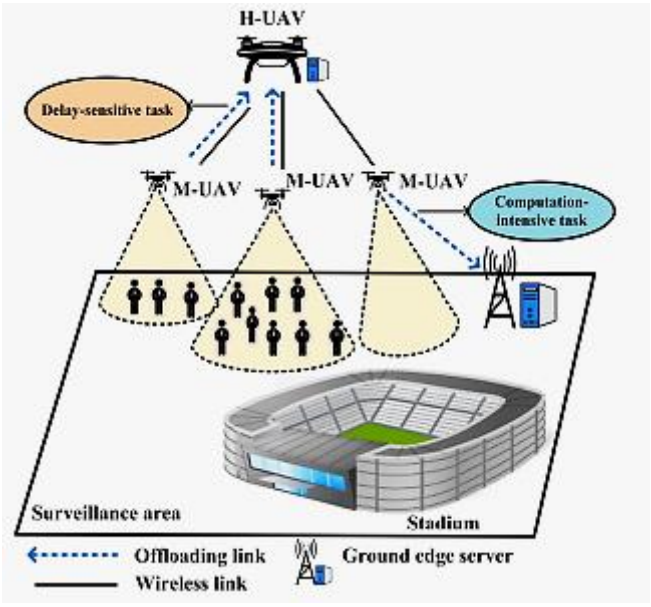

Fig. 1. UAV edge intelligence system model

Communication between edge server on people, vehicles, and embedded sensors in the vicinity is coordinated by catastrophe communication architecture. The vehicles may have UAVs, even if the employment of UAVs for ground server deployment is not given explicitly. Distributed and cooperative sensing enhances the information needed for command and control to sustain situational awareness.

2.1 Ground mobile user model

The distributions of GUs are deployed in random field in a circular area. The change in location are updated during the duration t=0 and $\triangle_{t,t-1}$ between t and t-1 time slots.

The velocity and direction of the GU is

$$v_n(t) = k_1 v_n(t-1) + (1-k_1)\bar{v} + \sqrt{1-k_1^2}\phi_n, \quad (1)$$

$$\theta_n(t) = k_2 \theta_n(t-1) + (1-k_2)\bar{\theta} + \sqrt{1-k_2^2}\varphi_n, \quad (2)$$

where $0 \le k_1, k_2 \le 1$ are the adjusted state of GUs with average velocity $\bar{v}$ and average direction $\bar{\theta}$ of all GUs. ϕ_n and φ_n are the Gaussian distributions. The location of the UAV in the t^{th} time slot is

$$l_m^{UAV}(t) = [x_m^{UAV}(t), y_m^{UAV}(t)]$$

$$m \in \{1,2,3 \dots\dots\dots\dots M\} \quad (3)$$

2.2 Energy consumption model

There are three categories considered for the energy consumption model of UAV.

2.2.1 Energy consumption during flying

The energy consumption during flying from one BS control to another BS control in the given time slot *t*–1 is computed as

$$E_f(t) = P_f \frac{\sqrt{[x_m^{UAV}(t) - x_m^{UAV}(t-1)]^2 + [y_m^{UAV}(t) - y_m^{UAV}(t-1)]^2}}{V} \quad (4)$$

2.2.2 Energy consumption during hovering

The energy consumption during hovering from the LoS channel between UAV and GU in the specified time slot t-1 is computed as

$$E_h(t) = P_h \frac{\mu_n(t) N_b}{R_m(t)}, \quad (5)$$

where P_h is the UAV power during hovering, $\mu_n(t)$ is the offloaded task in t^{th} slot, N_b is the transmitted bits per task. $R_m(t)$ is the throughput rate of UAV transmission and is given as

$$R_m(t) = log_2(1 + \frac{P_t g_m(t)}{\sigma^2}) \quad (6)$$

P_t is the UAV transmission power and $g_m(t)$ represents channel gain of the GU and BS. σ^2 is the Gaussian white noise power.

2.2.3. Energy consumption during computing

The total computing energy of UAV during offloading in the given slot *t*-1 is calculated as

$$E_c(t) = \gamma_c C (f_c)^2 \mu_n(t) N_b. \quad (7)$$

γ_c represents the effective switched capacitance and *C* represents the CPU cycles to complete one task, f_c is the system CPU frequency. $\mu_n(t)$ is the amount of offloaded task.

2.3 Task computation model

The computational task of M-UAV is to perform which can be of local computation or of edge computation linked with ground edge server. The local computing of M-UAV is described as $\alpha_j^l = 0$ and $\alpha_j^l = 1$ represents the task offloading of M-UAV.

2.3.1 Local computing

The task execution time duration depends on the clock frequency $f_{h,K}$ and CPU cycles $L_{j,t}$ to enable the computational capability of M-UAV.

$$T_K^{exe} = \frac{L_{j,t}}{f_{h,K}} \quad (8)$$

The energy consumption to execute a task is computed as

$$E_K^{loc.exe} = k L_{j,t} f_K^2 \quad (9)$$

In (9), *k* represents the switched capacitance of the device.

$$U_K^{local} = \alpha_1^l \frac{T^{loc.exe}}{\max T^{loc.exe}} + \beta_2^l \frac{E_K^{loc.exe}}{\max E_K^{loc.exe}} \quad (10)$$

In (10), α_1^l and β_2^l are predefined weight parameters to control latency and energy computing of the local phase.

2.3.2 Task offloading

The generated tasks after some period are forcedly dropped due to their characteristics and sensitive categorization either delay-oriented or energy-oriented service. Any such task is defined as $\alpha_j^2 \in [0,1]$. The delay-oriented task is offloaded to the H-UAV. The transmission delay and the energy consumption are computed as

$$T_K^{H-UAV,exe} = \frac{S_{j,t}}{R_{m,t}} \quad (11)$$

and

$$E^{H-UAV,exe} = P_{j,t} T_K^{H-UAV,exe}. \quad (12)$$

From (11) and (12) we can compute the total cost for edge computing as

$$U_K^{edge} = \alpha_l^{exe} \frac{T_K^{H-UAV,exe}}{\max T_K^{H-UAV,exe}} + \beta_l^{exe} \frac{E_K^{edge,tx}}{\max E_K^{edge,tx}} \quad (13)$$

The different weights are enabled to improve the energy consumption and low latency.

3 DRL framework for edge intelligence (DRLEI)

The DRL problem is formulated to solve the issues given below.

1. The location and direction of UAV are difficult to control due to the dynamic environment. The tasks may arrive and release dynamically so that the task specific requirements depend on when to execute and when to offload.

2. Even though the conventional algorithms such as linear and dynamic programming can give the optimal solution when the number of UAVs are limited, However, the scalability and complexity raises due to the increase in number of UAVs.

3. The traditional RL optimization depends on the action specific and reward specific environment. But we proposed the MDP strategy to learn the new energy efficient task offloading without prior knowledge about the dynamic environment.

DRL framework overcomes the limitations and solves scalability issue and computational complexity issues in order to provide the energy efficient offloading solution.

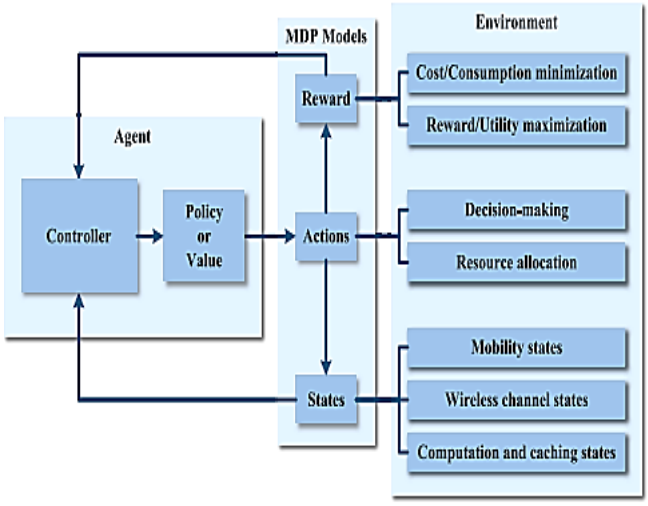

Fig. 2. DRL framework for edge intelligence

3.1 State space

A number of space input metrics are taken into consideration for the UAV network, including the job D's size, the task C's CPU cycle count, the CPU frequency f, the location l and the task type ∈ [0,1]. These are the state space of the DRL framework.

$$Q(s_t, a_t) = \mathbb{E}[\sum_{t=0}^{T} \omega r_{t+1} | s_t^*, a_t^*] \quad (14)$$

$\omega \in [0,1]$ is the discount factor. s_t^*, a_t^* are optimal state and action policy of M-UAV.

3.2 Action space

The DRL action agent is M-UAV which will selects a particular action from the transition state probabilities. The binary offloading is considered to offload or to execute the agent's task. It depends on the task type.

3.3 Energy efficient reward function

The agent is trained to maximize the total reward function without compromising energy consumption.

$$Z_{j,K} = -\alpha_j^{exe} \frac{U_K^{H-UAV,exe}}{\max U_K^{H-UAV,exe}} - \alpha_j^1 \left(1 - \alpha_j^2\right) \frac{U_K^{edge}}{\max U_K^{edge}} - (1 - \alpha_j^l) \frac{T^{loc.exe}}{\max T^{loc.exe}} \quad (15)$$

The smaller the reward, the lower the cost, and vice versa.

The cumulative utility function of each agent is computed as

$$Z_j = \sum_{k=1}^{M} Z_{j,K} \quad (16)$$

The reward function is based on the utility function and computed as

$$r_t^j = \begin{cases} p, & if\ Z_{j,t} - Z_{j,t-1} < 0 \\ q, & if\ Z_{j,t} - Z_{j,t-1} > 0 \\ 0, & otherwise \end{cases} \quad (17)$$

In Eqn. (17), p is the positive reward, q is the negative reward of each agent. It depends on the cumulative utility function.

3.4 MDP problem formulation

From [16], the DRL policy pertaining to the state transition probability of selecting the optimal action a_t in conjunction with the current state s_t. The main objective of MDP is to attain the optimal policy π^* to increase the reward function achieved for each M-UAV and is given as

$$\arg max\, r_t^j = \frac{\sum_{t=0}^{T-1} r_{t+1}}{T} \quad (18)$$

$$s.t\ \sum_{t=0}^{T-1} \mu_n(t) \geq Z_j \quad (19)$$

The constraint indicates the UAV energy consumption that guarantees minimum amount of task offloaded jobs from T slots.

The suggested algorithms allow the UAV to make decisions on its own, without largely depending on outside instructions, based on the information it gathers. The energy constraint is intended to compensate the agent for using resources efficiently. The agent should be rewarded more if it can complete the task with less processing power. Unmanned aerial vehicle (UAV) Edge Intelligence (DRLEI) algorithms that are based on DRL are essential for maximizing the capabilities and performance of UAVs through the use of edge computing.

Based on the above algorithm, the general approach for calculating the optimal reward function is solved using Eqn. (18). The novel energy constraint reward function is developed in this solution to set the initial state of UAV to optimal solution.

DRLEI Algorithm with MDP policy

Initialize : Values for D, C , f, $s_{j,t} = [0]$
$r=0$
N-Number of Episodes
Repeat : for j=0 to N do
Let t=0, T=0 and get initial state $s_{j,t}$
Repeat : update action a_t to obtain optimal solution
update s_t
Use (18) to update reward functions
If : $\sum_{t=0}^{T-1} \mu_n(t) \geq Z_j$
Return : Z_j cumulative reward, optimal task offloading and $R_m(t)$
End if

4 Results and discussion

4.1 Simulation environment used

In this section, Table 1 details the list of simulation parameters. The fundamental parameters for UAV are frequency, CPU cycles, UAV transmitted power, flying power and hovering power etc. The evaluated parameters are simulated through MATLAB software using Laptop core i3 with 16 GB RAM and 1 TB ROM.

4.2 Discussion

The evaluation parameters of the proposed algorithm are velocity, Average battery Energy of the UAV, computation system delay and average throughput. These parameters are compared with the existing algorithms such as Q-Learning which is a popular method in this MEC enabled UAV, The UAVs features, parameters, and implementation method (simulation) were all completed under identical circumstances and using the simulation parameters. The simulation environment consists of K UAVs, where K varies from 2 to 12 for better computational complexity.

Table 1. Simulation parameters

Parameter	Value
UAV environment	100×100 m^2
Number of GU	25
Base station radius	200 m
Velocity	20 m/s
UAV transmitted power	0.1 W
UAV flying power	110 W
UAV hovering power	80 W
Packet interval	0.1 sec
CPU frequency	2 GHz
Number of bits per task	100 Mb
Effective switched capacitance	10^{-27} F
Number of CPU cycles	1000

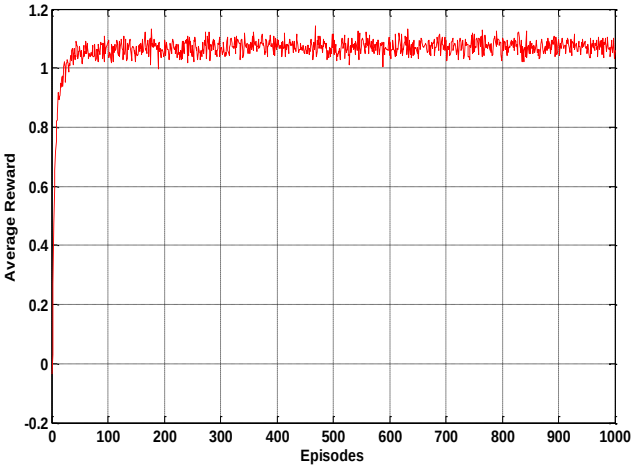

Fig. 3. Average reward of UAV

Figure 3 depicts the average reward of proposed algorithm which is higher than the conventional Q-Learning algorithms, the proposed method (DRL+MDP), is achieved larger cumulative reward and its convergence rate is slightly higher when the number of UAVs is increased.

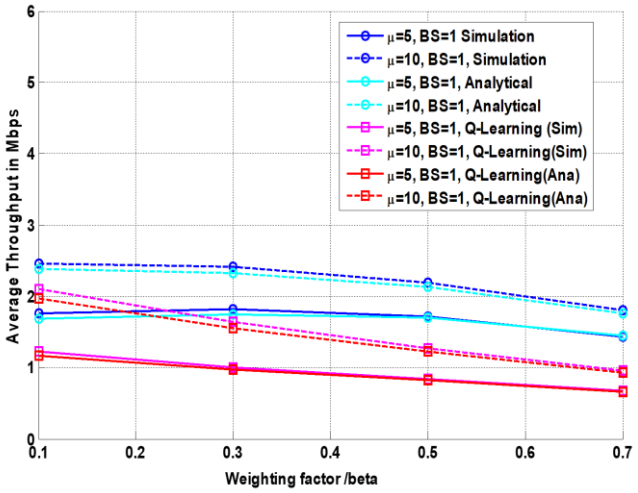

Fig. 4. Average throughput of UAV

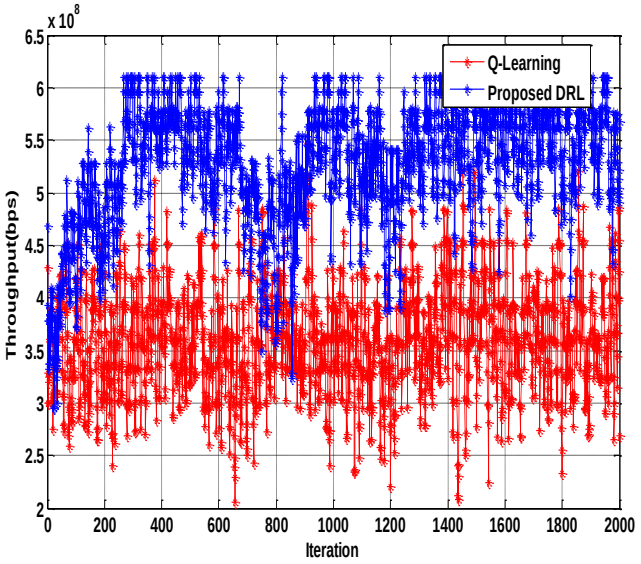

Fig. 5. Comparative analysis of UAV's throughput

The combined throughput for each episode of the suggested method and the Q-Learning algorithm is shown against the weighting parameter in Figs. 4 and 5. The product of the number of bits per job Nb and the offloaded tasks from all UAVs in an episode is our definition of the total throughput each episode. Initially, out of all the methods at any velocity, the suggested approach obtains the highest cumulative throughput each episode. Secondly, as *N* rises, the total throughput each episode decreases. Third, for all methods, the total throughput per episode rises as velocity μ decreases. For instance, at μ=5 m/s and μ=10 m/s, the suggested method and Q-Learning respectively reach their maximum cumulative throughput per episode. This is because the path planning issue progressively gets less problematic. We have found the global optimum values.

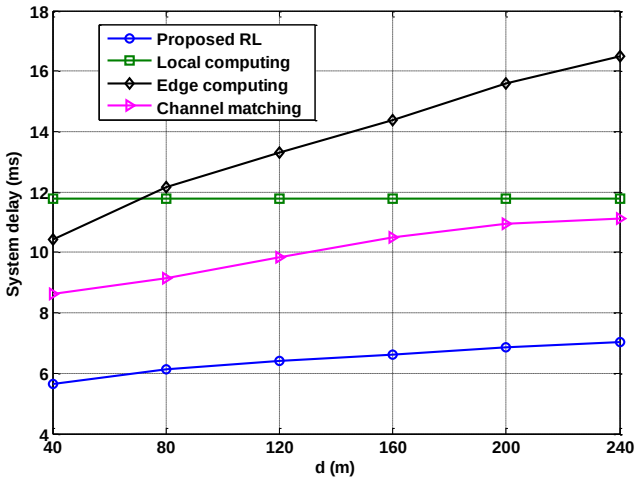

Fig. 6. System delay with UAV distance

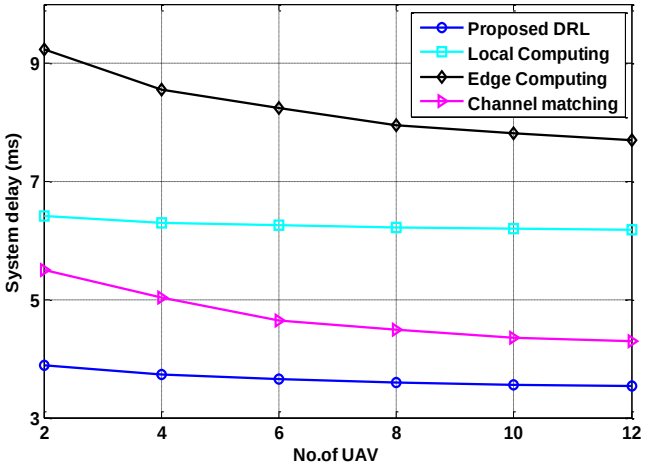

Fig. 7. System delay with UAV

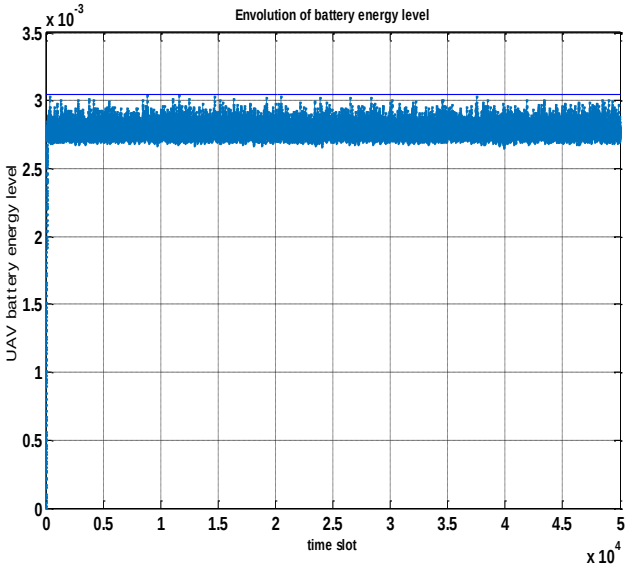

Fig. 8. UAV battery energy level

Figure 6 represents the computational delay against the UAV distance between M-UAV location point and MEC GU. It shows that the system delay increases when the distance tends to increase. There is a correlation between the distance of the device and the MEC system server delay.

Figure 7 compares the edge computing, local computing, and channel matching policies while varying the computation delay's performance in relation to the number of UAVs. It is evident that the three compute offloading policies converge and perform much better than the strategy when the mobile device is near the MEC server.

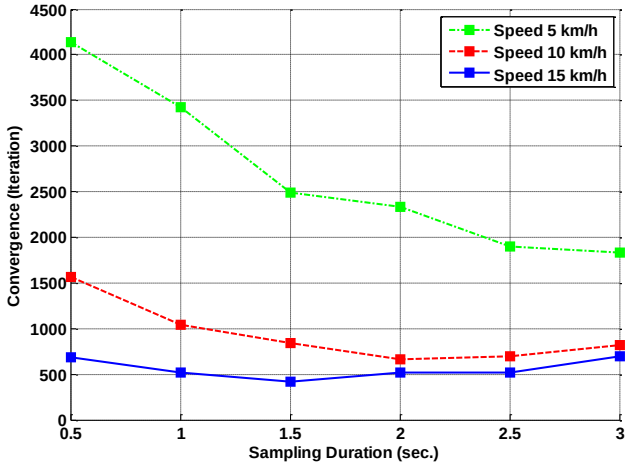

Fig. 9. Convergence rate *versus* sampling duration

Battery energy level is maintained as a number of rounds. Figure 8 displays the simulation and comparison outcomes of the aforementioned techniques in terms of the battery lifespan of UAV networks.

The proposed algorithm convergence analysis is shown in Fig. 9 with the increasing sample duration. The DRL algorithm tends to avoid dropping tasks by prolonging the average completion time in order to achieve a minimum execution cost.

Table 2 indicates that the average energy consumption of UAV while using DRLEI algorithm. The comparative parameters are average execution cost and with computational capacity and varying with offloading task with prevailing algorithms proposed in related works. It was discovered that the average total cost in terms of computing power and energy consumption are sufficient to demonstrate the effectiveness of the proposed RL-based edge computing algorithm. The proposed DRLEI reduced the offloading cost as well as average execution cost by 52.13%, 43.5% and 28.7% in terms of computational cost, tasksize and the drop. Table 2 clearly examined the effect of the proposed DRLEI in comparison with DQN, local and edge computing.

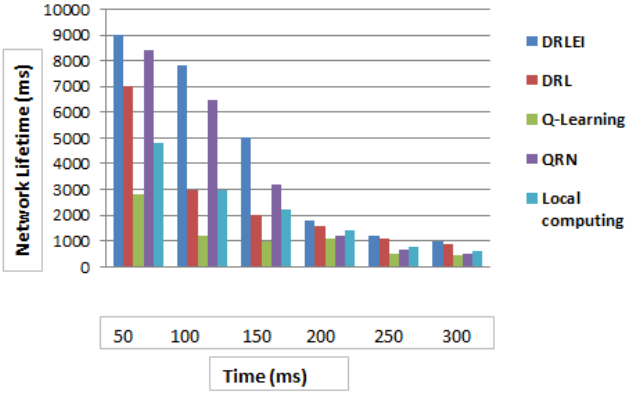

Fig. 10. Comparative network lifetime

The examined result shows the impact of task size and processing power on battery energy usage and task execution latency. DRLEI outperforms the conventional strategies by at least 4% and 10%, respectively during the testing phase. The M-UAV network lifetime is shown in Fig. 10 which is computed for local computing, edge computing, Q-learning, DRL and proposed DRLEI algorithms. Finally, the proposed DRLEI strategy is attributed with several parameters 1) the hierarchical architecture for task execution. 2) RL frame work with novel reward function for task offloading. 3) The implementation of DRLEI with minimum processing delay and reduced complexity in order to handle the overestimation problem.

5 Conclusion

The proposed DRLEI strategy is implemented with the proven results when offloading computation. The facilitation of offloading is done through the ground edge server which helps the edge users for computation-intensive activities, The successful completion of all execution and offloading tasks based on energy consumption and task execution latency. The DRLEI framework helps in cost optimization and computational power optimization as weighted sum average. An agent performs best optimization through rigorous training phase and deciding the best offloading strategy and taking actions regarded with the novel reward functions of the proposed DRLEI scheme. Lastly, the convergence of the DRLEI is tested through simulation. In comparison with DQN, edge, and local execution strategies. The comparative results are outperformed and remarkable. Reduced operational ranges, lower payloads, and shorter flight periods are outcomes of single-use UAV restrictions that could be solved by the proposed work combining networked and collaborative UAVs. System setup took less than five minutes in a large-scale fire practice, and it only took a few more minutes to provide aerial surveillance of the whole region.

Table 2. Results of energy consumption (EC) performance of the proposed DRLEI with existing algorithms

Refs.	Algorithm	Avg EC (Joules)	Avg EC with computational capacity	Avg EC varying with offloading task size
[11]	Local	28.54	48.18	49.55
[11]	Edge	22.58	43.5	45.87
[16]	DQN	19.18	39.35	41.84
[4]	DRL	19.78	38.75	41.48
Proposed method	DRLEI	18.17	37.65	38.24

References

[1] P. Wei et al., "Reinforcement Learning-Empowered Mobile Edge Computing for 6G Edge Intelligence," *IEEE Access*, vol.10, pp. 65156-65192, 2022. doi: 10.1109/ACCESS.2022.3183647

[2] H. Sami, H. Otrok, J. Bentahar, and A. Mourad, "AI-based resource provisioning of IoE services in 6G: A deep reinforcement learning approach," *IEEE Trans. Network and Service Management*, vol. 18, no. 3, pp. 3527-3540, 2021. doi:10.1109/TNSM.2021.3066625

[3] P. Zhou *et al*., "QoE aware 3D video streaming via deep reinforcement learning in software defined networking enabled mobile edge computing," *IEEE Trans. Netw. Sci. Eng*., vol. 8, no. 1, pp. 419-433, 2021. doi:10.1109/TNSE.2020.3038998

[4] Y. Kunpeng *et al*., "Reinforcement learning-based mobile edge computing and transmission scheduling for video surveillance," *IEEE Trans. Emerg. Topics Comput*, vol. 10, no. 2, pp. 1142-1156, 2021. doi: 10.1109/TETC.2021.3073744

[5] O. Yildiz and R. Sokullu, "Deep Q-Learning based resource allocation and load balancing in a mobile edge system serving different types of user requests," *Journal of Electrical Engineering*, vol. 74, no. 1, pp. 48-55, 2023. doi: 10.2478/jee-2023-0005

[6] M. Rouissat *et al*., "Implementing and evaluating a new silent rank attack in RPL-contiki based IoT networks," *Journal of Electrical Engineering*, vol.74, no. 6, pp. 454-462, 2023. doi: 10.2478/jee-2023-0053

[7] A. Khan, S. Gupta, and S. K. Gupta, "Multi-UAV integrated HetNet for maximum coverage in disaster management," *Journal of Electrical Engineering*, vol. 73, no. 2, pp. 116-123, 2022. doi:10.2478/jee-2022-0015

[8] Q. Wu, Y. Zeng, and R. Zhang, "Joint trajectory and communication design for multi-UAV enabled wireless networks," *IEEE Transactions on Wireless Communications*, vol. 17, no. 3, pp. 2109-2121, 2018. doi: 10.1109/TWC.2017.278929

[9] S. Amalorpava Mary Rajee and A. Merline, "Machine intelligence technique for blockage effects in next-generation heterogeneous networks," *Radioengineering*, vol. 29, no. 3, 2020. doi: 10.13164/re.2020.0555

[10] S. Nagarajan *et al*., "Multi-Agent Reinforcement Learning for resource allocation in container based cloud environment," *Expert Syst*., vol. 1, no. 22, 2023. https://doi.org/10.1111/exsy.13362

[11] V. T. M. Babu *et al*., "Survey on data communication for UAV and flight control system in MM wave communications," In *AIP Conference Proceedings*, vol. 2790, no. 1, Aug. 2023. doi: 10.1063/5.0152781

[12] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu, and N. D. Sidiropoulos, "Learning to optimize: Training deep neural networks for interference management," *IEEE Trans. Signal Processing*, vol. 66, no. 20, pp. 5438-5453, Oct 2018. doi:10.1109/TSP.2018.2866382

[13] M. Navaneethakrishnan *et al*., "Design of Biped Robot Using Reinforcement Learning and Asynchronous Actor-Critical Agent (A3C) Algorithm," In 2023 2nd *IEEE International Conference on Vision Towards Emerging Trends in Communication and Networking Technologies (ViTECoN)*, pp. 1-6, 2023.

[14] P. V. Klaine, M. A. Imran, O. Onireti, and R. D. Souza, "A survey of machine learning techniques applied to self-organizing cellular networks," *IEEE Commun. Surv. Tutor*., vol. 19, no. 4, pp. 2392-2431, 2017. doi:10.1109/COMST.2017.2727878.

[15] R. Li *et al*., "Intelligent 5G: When cellular networks meet artificial intelligence," *IEEE Wirel. Commun*.," vol. 24, no. 5, pp. 175-183, 2017. doi: 10.1109/MWC.2017.1600304WC

[16] K. Guo, R. Gao, W. Xia, and T. Q. S. Quek, "Online learning based computation offloading in MEC systems with communication and computation dynamics," *IEEE Trans. Commun*., vol. 69, no. 2, pp. 1147-1162, 2021. doi: 10.1109/TCOMM.2020.3038875

[17] S. A. M. Rajee, A. Merline, and M. M. Yamuna Devi, "Game theoretic model for power optimization in next- generation heterogeneous network," *SIViP*, vol. 17, pp. 3721-3729, 2023. doi:10.1007/s11760-023-02599-8

[18] F. Ahmed, and M. Jenihhin, "A Survey on UAV Computing Platforms: A Hardware Reliability Perspective," *Sensors*, vol. 22, no. 16, p. 6286. 2022. doi:10.3390/s22166286

Received 16 December 2023