Original scientific paper
https://doi.org/10.33180/InfMIDEM2025.305

Informacije MIDEM
Journal of Microelectronics,
Electronic Components and Materials
Vol. 55, No. 3(2025), 183 – 192

Multi-user Task Offloading for Mobile Edge Computing Based on Reinforcement Learning

Jembu Mohanram Nandhini[1], Kaliaperumal Saravanan[2], Kesavan Anuratha[3] and Sankar Uma[4]

[1]Department of CSE, Sri Sai Ram Institute of Technology,Chennai Tamil Nadu, India.
[2]Department of CSE, Saveetha School of Engineering, SIMATS, Chennai Tamilnadu, India.
[3]Department of IT, Sri Sai Ram Institute of Technology, Chennai Tamil Nadu, India.
[4]Department of IT, Panimalar Engineering college, Chennai Tamil Nadu, India.

Abstract: Mobile Edge computing (MEC) enables network functions and control programmable and operates key constituents of social networks in terms of increasing user's support on devices to carry out compute. It requires traffic offloading and task scheduling to improve the storage and fast computing. In this paper, a novel method, including data driven traffic modeling enabled by a Reinforcement learning algorithm (RLTOA), is proposed for offloading traffic and improving the computing speed and minimizing the application latency of the social network. The result of the proposed data driven modeling is compared with existing methods and validate how the data driven traffic modeling for providing the computation offloading service in terms of energy budget and the mobile drop and execution of edge server. The presented computation offloading, and energy management solutions can provide valuable perceptions for practical applications of MEC. Extensive numerical findings are presented to endorse the efficacy of RLTOA and display the effect of the social network requirement.

Keywords: MEC; Reinforcement Learning; Traffic offloading; Task scheduling

Razbremenitev večuporabniških nalog za mobilno robno računalništvo na podlagi okrepljenega učenja

Izvleček: Mobilno robno računalništvo (MEC) omogoča programiranje omrežnih funkcij in nadzora ter upravlja ključne sestavne dele družbenih omrežij z vidika povečanja podpore uporabnikom na napravah za izvajanje računalniških operacij. Za izboljšanje shranjevanja in hitrega računalniškega delovanja je potrebno razbremenjevanje prometa in načrtovanje nalog. V članku je predlagana nova metoda, vključno z modeliranjem prometa na podlagi podatkov, ki ga omogoča algoritem okrepljenega učenja (RLTOA), za razbremenitev prometa in izboljšanje hitrosti računalniškega obdelovanja ter zmanjšanje zakasnitve aplikacij družbenega omrežja. Rezultat predlaganega modeliranja na podlagi podatkov so primerjani z obstoječimi metodami in potrjujejo modeliranje prometa na podlagi podatkov za zagotavljanje storitve razbremenitve računalniških operacij v smislu energijskega proračuna in mobilnega padca ter izvajanja robnega strežnika. Predstavljene rešitve za razbremenitev računalniških operacij in upravljanje z energijo lahko zagotovijo dragocene ugotovitve za praktične aplikacije MEC. Predstavljeni so obsežni numerični rezultati, ki potrjujejo učinkovitost RLTOA in prikazujejo učinek zahtev družbenega omrežja.

Ključne besede: MEC; okrepljeno učenje; razbremenitev prometa; načtovanje nalog

** Corresponding Author's e-mail: anujournal381@gmail.com*

1 Introduction

Beyond 5G network (B5G) is assessed through the intensive and sensitive applications through traffic and computation offloading by increasing computational capacity to the edge of B5G networks. Reinforcement Learning (RL) can solve this problem using sparse and inaccurate network data. In this paper, we employ RL to develop an ideal task scheduling and computation offloading technique that reduces system energy usage. A framework for reinforcement learning based

How to cite:
J. M. Nandhini et al., "Multi-user task offloading for mobile edge computing based on reinforcement learning", Inf. Midem-J. Microelectron. Electron. Compon. Mater., Vol. 55, No. 3(2025), pp. 183–192

edge computing is introduced in B5G networks for mobile edge computing (MEC) server for social related network applications and for battery-powered and resource-controlled devices [1]. In this paradigm, these social related delay-sensitive applications are shifted from resident users to nearby network edge server; edge computing is a promising technology to address the problem [2]. Since the processing capacity of these pervasive mobile edge servers are frequently constrained, offloading all work from devices to edge servers may, on the other hand, result in larger latency. [3-5] Additionally, compute task offloading, particularly in 5G networks that are diverse and ultra-dense, can result in increased interference and unanticipated transmission delays. On the other hand, local computing can considerably minimize the latency of job execution [6]. The contradiction is found between computation delay with energy consumption is essentially what determines whether to execute tasks locally or offload them when creating an offloading strategy [7-9]. There are various algorithms focusing on minimizing the transmitted powers, and to maximize throughput [10]. In the framework of the B5G network, computation complexity and task scheduling are handled by means of adding Small base stations (BS) as indoor base stations that have expanded significant attention [11]. Heterogeneous network (Hetnet) comprises of Macro BS (MBS) overlapped with small BS (SBS) and an autonomous power distribution is possible with the aid of RL. This heterogeneity structure offers features such as network data rate, connectivity, and energy efficiency [12-16]. In this B5G Hetnet, Co-operative and distributive learning outcomes optimization have been playing significant role in addressing energy related problem [17-22]. Many researchers suggested binary offloading and partial offloading to adopt the system flexibly [23], the game-theoretic modeling and the quadratic programs are also implemented with non-convex optimization. Then, the design concepts for B5G networks with MEC are different from those for MEC systems with SBS to minimize system energy consumption. A deep Q-Network (DQN) is based on the reinforcement learning working with deep neural networks, enabling more effective decision-making in complex MEC environments. Battery-powered device systems are preferable in B5G networks [24]. Collaborative design is important in task offloading since it is difficult to make decision on when to offload and when to execute, in this case, the key parameters are the CPU-cycle frequencies and time interval for execution [25]. The design goal is to minimize battery energy usage to optimize computation performance in contrast to MEC systems with battery-powered components. Along with these novel design considerations, managing the service constraints and the battery energy dynamics presents additional difficulties, this paper contributes to reduce the latency and computational cycle of MEC server using RL algorithm. The key ideas of this research are as follows:
By jointly optimizing the offloading decision, compute the total computation capability, energy consumption, and battery energy. The co-operative problem is formulated.

To optimize the traffic offloading process, the Enhanced RL algorithm is introduced so that the decision making of task scheduling and computational speed is increased. In each time slot, the algorithm tries to optimize the CPU-cycle frequency and battery energy through trial and error.

To analyze and compare the simulated results with prevailing algorithms reported with greedy allocation so as to impart the effectiveness of the proposed RL based MEC system

The articulation of proposed B5G MEC system is illustrated in Section 2 and figure 1. The experimental findings and a comparison with the current approaches are discussed open in Section 3. The conclusion part is given in Section 4.

2 System methodology

MEC is an effective system to equip the management of mobile devices and is illustrated in figure 1. MEC server acts as a cloud head and runs a virtual machine offloading the computational task for edge mobile users. We consider the communication model consists of one macrocell and smallcells which are surrounded by the mobile devices and MEC server. The Euclidean distance between the macrocell and MEC server is defined as

$$d_{j,t} = \sqrt{\| w_t^{MC} - w_{j,t}^{MEC} \|^2}, \ \forall t \in \mathfrak{F}, j \in \mathrm{M}. \quad (1)$$

The channel gain between communication model is considered as free space path loss and is denoted as

$$h_{j,t} = \eta d_{j,t}^{-2} = \frac{\eta}{\| w_t^{MC} - w_{j,t}^{MEC} \|^2} \ \forall t \in \mathfrak{F}, j \in \mathrm{M}. \quad (2)$$

Frequency division multiple access serves the time intervals between the MEC server and all users. The complete bandwidth is divided into G smaller bands and each mobile device gets assigned B5G bands.[4] The signal-to-noise ratio (SNR) is calculated as

$$SNR_{j,t} = \frac{h_{j,t} P_{j,t}}{\mu B / G} \quad (3)$$

$$SNR_{j,t} = \frac{\eta p_{j,t}}{\frac{\mu B}{G \| w_t^{MC} - w_{j,t}^{MEC} \|^2}} \quad \forall t \in \mathfrak{F}, j \in \mathrm{M}. \tag{4}$$

where $P_{j,t}$ is the transmit power of j[th] mobile edge device and μ specifies additive white Gaussian noise. The data rate of the MEC system is

$$R_{j,t} = \frac{B}{G} log_2 \left(1 + \frac{\rho_0 p_{j,t}}{\frac{\mu B}{G \| w_t^{MC} - w_{j,t}^{MEC} \|^2}} \right) \forall t \in \mathfrak{F}, j \in \mathrm{M}. \tag{5}$$

Similarly, the optimality of MEC is a process to attain the best outcomes when the j[th] mobile device offloads the duty to the edge MEC server. It can effectively located in a fixed coordinates and denoted as $w_t^{EC} = (x_{EC}, y_{EC}, 0)$ and the corresponding distance between j[th] mobile device and edge MEC server in time slot is represented as

$$d_{EC,t} = \| w_{j,t}^{M} - w_t^{EC} \| \tag{6}$$

The channel gain is defined as

$$h_{EC,t} = \frac{g}{\left[d_{Ec,t} \right]^2} \tag{7}$$

G is denoted as the power gain and the Euclidean distance assumed as 5 meters. The data rate in the specified duration t is assumed as

$$R_{EC,t} = B_u log_2 \left(1 + \frac{h_{EC,t} P_{j,t}}{\sigma^2} \right), \forall t \in \mathfrak{F}, j \in \mathrm{M} \tag{8}$$

B_u is the total bandwidth and σ is the noise power.

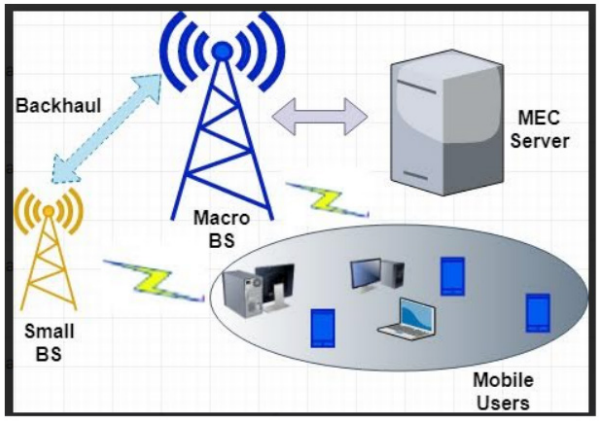

Figure 1: Articulation of B5G MEC system

The articulation of the proposed B5G MEC system is shown in figure 1, It comprised of heterogeneous network with small BS and Macro BS to the core network or MEC server. If the wireless backhaul is overloaded due to resource allocation, task offloading may suffer from high transmission delays. Task scheduling and computational offloading plays vital role in the backhaul communication.

2.1 Task scheduling model

In this work, each mobile device has a computational task computed locally in the region of MEC server which deployed near the SBS in the time slot t. The computing decision action is assumed as $a_j^l = 1$ representing task is offloaded if it is $a_j^l = 0$ means the task is computed. Execution time and battery usage are used to differentiate between local and edge computing.

2.1.1 Local computing

The computational ability is discriminated by means of the frequency cycle of CPU and is denoted as $f_{h,K}$. The local task execution time is calculated using [5]

$$T_k^{exe} = \frac{L_{j,t}}{f_{h,K}} \tag{9}$$

where $L_{j,t}$ is the CPU cycle essential to complete a task of the edge device. At the same time the energy consumption for execution is given by

$$E_K^{local} = kL_{j,t} f_{kj,t}^2 \tag{10}$$

The switched capacitance of the edge device is denoted as K. we consider that $k = 10^{-28}$ [11].

The inclusion of local execution time and energy consumption results in the overall cost of task execution [5],

$$U_k^{local} = \alpha_1^l \frac{T^{local}}{\max T^{local}} + \beta_2^l \frac{E_k^{local}}{\max E_k^{local}} \tag{11}$$

α_1^l and β_2^l are the weights to control over the computing phase.

2.2 Task offloading model

Task is offloaded when the computing resources are running out of memory, The task are of two types such as delay-sensitive and energy-sensitive which are varying with sizes and computational requirements. so the generated tasks are continuously dropped. The formula for calculating transmission delay and energy consumption are as follows,

$$T_k^{MEC,tx} = \frac{s_{j,t}}{R_{j,t}} \tag{12}$$

$$T_k^{MEC,exe} = \frac{L_{j,t}}{f_{h,k}} \tag{13}$$

$$E_k^{MEC,tx} = P_{j,t} T_k^{MEC,tx} \tag{14}$$

$$E_k^{MEC,exe} = k_H L_{j,t} f_{h,k}^2 \tag{15}$$

The effective switched capacitance k_H and is set to the value of 10^{-28} Farad. We consider a high speed processor with a clock frequency of f = 2.4 GHz, the cycle length is derived from the clock frequency $L_{j,t}$ = 1/f, which results approximately as 0.5 ns, using an effective switched capacitance value and clock cycle values, the energy consumption is calculated as 1 x 10^{-28} watts. Smartphones and Internet of Things end nodes are edge devices that must adhere to strict energy constraints. The effective switched capacitance should be chosen to guarantee practical viability in energy models for low-power processors. The theoretical extreme energy efficiency in computational modeling is reflected in this ultra-small power consumption estimate, especially in mobile edge computing scenarios with advanced nanotechnology concerns. For this reason, in accordance with CMOS standards, we have selected the switching capacitance value 10^{-28} Farad as low as feasible on MEC energy consumption, particularly with task offloading and computational efficiency in order to stay consistent with earlier benchmarks.

The computational delay of a task can be calculated using the sum of communication/transmission delay and the execution delay from MEC server to edge devices.

$$T_k^{MEC} = T_k^{MEC,tx} + T_k^{MEC,exe} \tag{16}$$

$$E_k^{MEC} = E_k^{MEC,tx} + E^{MEC,exe} \quad (\tag{17}$$

The total cost is calculated as

$$U_k^{MEC} = \alpha_1^{MEC} \frac{T_k^{MEC}}{\max T_k^{MEC}} + \beta_2^{MEC} \frac{E_k^{MEC}}{\max E_k^{MEC}} \tag{18}$$

The total cost for edge execution is derived as

$$U_k^{MC} = \alpha_1^{MC} \frac{T_k^{MC}}{\max T_k^{MC}} + \beta_2^{MC} \frac{E_k^{MC}}{\max E_k^{MC}} \tag{19}$$

The computational effort is determined by weights in the areas of energy consumption and latency in the task transmission, edge computing, and result transmission phases.

The main objective is to formulate the sequential task function R and is denoted as

$$U_j = \sum_{k=1}^{B} U_{kj,i} = \sum_{i=0}^{l} a_j^1 a_j^2 U_{kj,i}^{MEC} + \\ + a_j^1 \left(1 - a_j^2\right) U_{kj,i}^{MC} + a_j^1 \left(1 - a_j^2\right) U_{kj,i}^{local} \tag{20}$$

The total size is represented as B in a set R. Furthermore the above problem is formulated as

$$min_A = \sum_j U_j \\ s.t\, a_j^1 a_j^2 T_{kj,i}^{MEC} + a_j^1 \left(1 - a_j^2\right) U_{kj,i}^{MC} + \\ + \left(1 - a_j^1\right) T^{loc,exe} \le T_{kj,i}^{max}, \forall k = 1, \ldots\ldots, B \tag{21}$$

A is the set of tasks to be completed within the time slot T_k^{min}. The optimal offloading action is needed to be a decision variable that indicates the decision to reduce the overall system cost. In order to determine the best choice within a time slot, network data such as job information and processing capacity are utilized. The traditional methods such as NP-hard, MINLP and non-convex optimization are not much effective due to the intelligence. That's why we consider the RL based Markov decision process making (MDP) learning to provide more efficient decision to schedule/offload a task.

3 RL based MDP framework

The traditional methods are not efficient in optimization due to the following reasons. 1. The task specific environments are dynamic in nature due to traffic, load, and delay characteristics. Traditional optimization techniques cannot handle the dynamic behavior of the MEC B5G network. 2. Due to the lengthy convergence of time and scalability problem intelligent decision making is required to offload. 3. Prior knowledge about the network environment is a challenging task for the traditional techniques. But RL based MDP technique follows the learning by using trial and error method. To address the above shortcomings, we propose the RL based MDP algorithm as a model-free methodology so as to make intelligent decisions and information exchange between agents (Mobile Edge device). To demonstrate the RL understanding of the suggested MEC system, following are the brief description,

3.1.1 State Space $S_{j,t}$

The set of input metrics of each agent for task offloading decision occupies as a state space $S_{j,t}$ = {*D, c, f and dt*}. The symbol set represents the size D of the network, c is the cycle with computational capability f and

energy-sensitive or delay sensitive is the type of task and can be chosen as $dt \in [0,1]$

3.1.2 Action Space $a_{j,t}$

Each mobile device can choose a particular action in a given time slot t. according to the local information from the MEC network. The action can be of binary offloading either to be executed or to be offloaded.

3.1.3 Reward function $r_{j,t}$

The proposed RL framework optimizes the computational capacity by means of selecting the cumulative reward function as the decision based upon the reward function as follows

$$r_{j,t} = -a_j^2 a_j^1 \frac{U_{kj,i}^{MEC}}{\max U_{kj,i}^{MEC}} - a_j^1 \left(1 - a_j^2\right) \frac{U_{kj,i}^{MC}}{\max U_{kj,i}^{MC}} - \\ - \left(1 - a_j^2\right) \frac{U_{kj,i}^{local}}{\max U_{kj,i}^{local}} \quad (22)$$

I t indicates that the negative reward is given for higher cost function and vice versa. Then the utility function of each agent is denoted as

$$R_{j,k} = \sum_{k=1}^{B} r_{j,k} \quad (23)$$

The reward utility function is formulated as

$$u_t^j = \begin{cases} p\, if\, r_{j,t} - r_{j,t-1} < 0 \\ q\, if r_{j,t} - r_{j,t-1} > 0 \\ 0 \qquad otherwise \end{cases} \quad (24)$$

Case 1: p represents the positive reward if the computational cost is low.

Case 2: q represents the negative reward if the computational cost is high.

We consider co-operative Q-learning for the information exchange between the agents. As number of tasks is sent to the edge server, the pressure on mobile devices is lessened, as evidenced by the MEC server execution steadily increasing. Meanwhile, mobile execution remains relatively low but stable, suggesting that certain jobs are still completed locally.

RL based Task offloading Algorithm (RLTOA)		
Initialize	:	Task Values for {D, c, f and dt}
Ensure	:	$dt \in [0,1]$ Episode counter b = 1 to Max

for		episode =1 to max do
		for j=1 to M do
		for t=1 to T do

Choose action	:	Random action Or based on ε-greedy policy Schedule Task or offload task based on the MEC server
Calculate and Return	:	Reward utility function based on equation (24)
Apply		Trial and error until b > max

Return		end if
		end for
		end for

3.2 RL based task offloading algorithm

The de-centralized multi-agent task computation offloading technique, which combines a MDP with Q-learning to create an ideal offloading decision as illustrated, is the subject of the proposal in this section. It is built on the discussion from the previous section.

In Figure 2, each agent will compute locally or on the MEC server whenever new tasks are created in each time slot. As per algorithm, researches the distributing policy offloading, and then decides on a course of action based on its knowledge of the surrounding area. It is then promptly rewarded for that action. The action probability and state function is updated every iterations to optimize the weights.

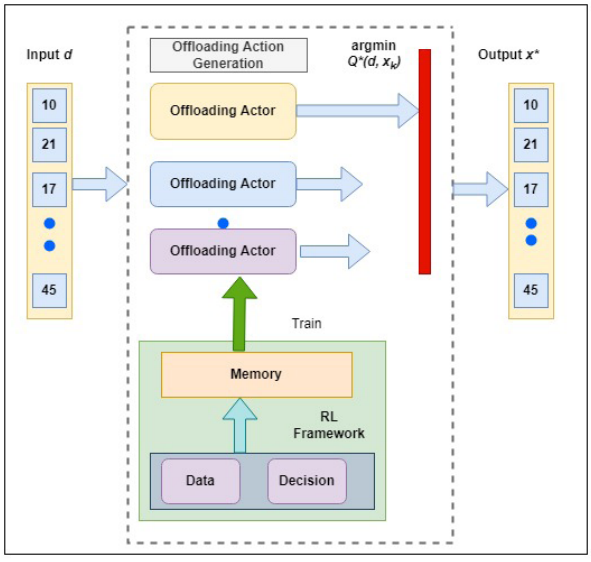

Figure 2: Articulation of RL based task offloading.

Because the edge servers and BSs in a cluster are limited in this scenario, we may cycle through every conceivable combination of BSs and edge servers to find the optimal task offload rate. algorithm for task offloading based on reconstruction from reinforcement learning. It seeks to create a single collection of data by combining the Action policy and offloading.

4 Results and discussion

The experimental results of RL based task offloading algorithm is tested and the comparison of results with conventional study through MATLAB simulation environment.

MEC eanbled Hetnet is simulated to assess the effectiveness of our proposed offloading decision algorithm for MEC edge systems. We regard the mobile edge devices as being dispersed randomly and uniformly within a 500 m radius disc. Femtocells that enable multiple edge users (MUEs and FUEs) range in number from two to forty. Every time slot has exactly one active FUE and MUE. The following are the detailed tabulation of simulation parameters.

Table 1: Simulation Parameters

Parameters	Value
Macro cell	1
No. of small cells	15
Bandwidth	20 MHz
Edge Bandwidth	1 MHz
Channel Power gain	1.42 x 10-4
Noise Power spectrum density	-174 dBm
CPU cycles	0~1.75 GHz
Task size	2-25 Mb
Computational capacity of MEC server	20 GHz
Computational capacity of MC server	15 GHz
Effective switched capacitance	10^{-28} Farad
Learning Rate	0.001
Discount factor	0.9
Total episode (max)	2000
Weights $\alpha_1^l = \alpha_1^h = \alpha_1^e$ Weights $\beta_1^l = \beta_1^h = \beta_1^e$	0.5
Total time steps per episode	100
Length of each slot	1 sec

The evaluation parameters are verified using MATLAB with a machine learning toolbox. RL enabled edge-computing setup is evaluated with average ratio of chosen modes to enable execution, drop and MEC server execution.

The cumulative reward is assessed for the convergence analysis. Due to this fact, average cumulative reward values are stored. The proposed problem is formulated with a delay restriction, and A penalty factor of 3 is added to the reward value if tasks exceed the deadline in order to manage the learning advancement.. The number of iterations was set to 500, 1000 and 2000 respectively to reduce the computational complexity. The proposed RLTOA show that when the iteration is reached to 500 times, the convergence stability is shown by using the loss function. The accuracy is validated with 96.34% , Figures 3 and 4 represent the mobile drop, MEC edge computing and local computing. That requires numerous interactions. Speciously,. The QoS parameters SNR, Battery energy, computational speed and energy efficiency (EE) are the key parameters for performance analysis. The results include the training phase and evaluation shows the training phase of the proposed RLTOA. The online learning system trains the agent to learn from the dynamic environment through trial and error. The delay-sensitive and intensive operation is checked with the MUEs and FUEs. The reward function designed to guarantee the QoS of UEs at the time slot. The RL optimization for the computational execution cost and average energy is shown in figures 3 and 4. To ensure that the task is completed, the MUE must meet a specific QoS standard that is consistently higher than a set threshold at all times.

Figure 3 and 4 depict the average ratio of execution and drop in a given time, There are different access possibilities for executing the total number of tasks in s given slot. The number of agents and their capacity to do the computing work determine the overall average cost. When compared to local, edge, and Q-learning networks, RLTOA is found to have a lower average cost. The suggested RLTOA controls the computational complexity as the number of agent's increases. As a result, the outcomes were better than the handling capacity when there were more agents. When compared to the three benchmarks, RLTOA eliminates the scalability problem. Overall, the pattern points to an adaptive execution strategy in which task offloading improves with time, resulting in fewer task drops and better use of available resources. Figures 5 and 6 represent the average execution cost and battery energy level of the edge user. The graphical illustration shows how effective the computing capacity interms of their execution cost and battery energy. The computation-intensive application from the ground edge level to the MEC server execution cost is reduced due to the shifting from local computing to edge node. The processing delays are significantly reduced thereby improving energy efficient task transmission.

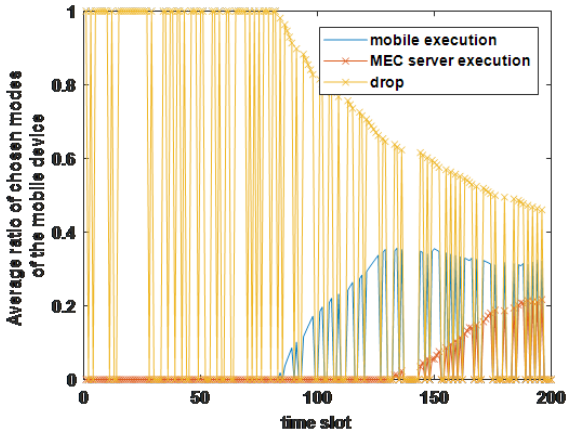

Figure 3: Average ratios of task execution strategies in mobile edge computing

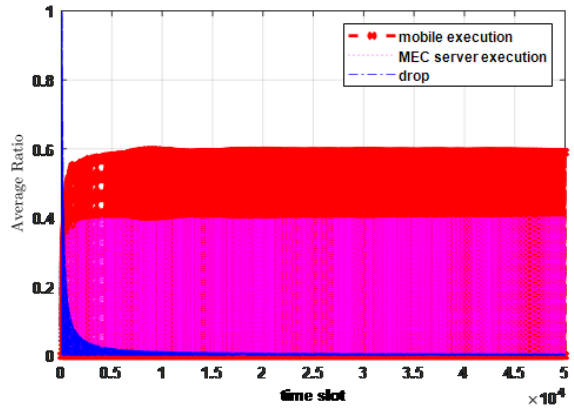

Figure 4: Average performance metrics of task execution strategies

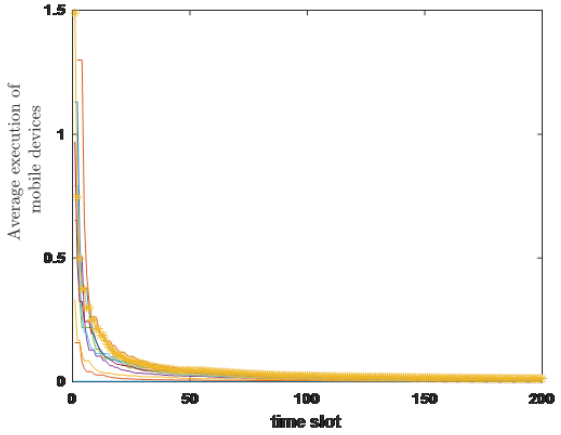

Figure 5: Temporal evolution of cumulative cost of MEC

The battery energy level based on the the latency is shown in Figure 7. Whenever the job size increases, so increases task execution delay, since they depends on the CPU cycles to complete the process. Computational speed depends on how large the task size and how the delay rises.

.

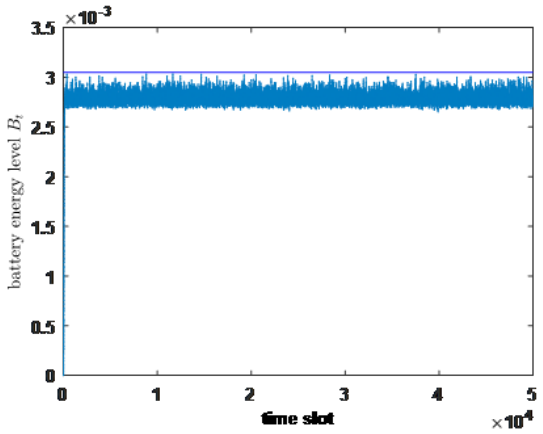

Figure 6: Battery energy consumption dynamics in MEC

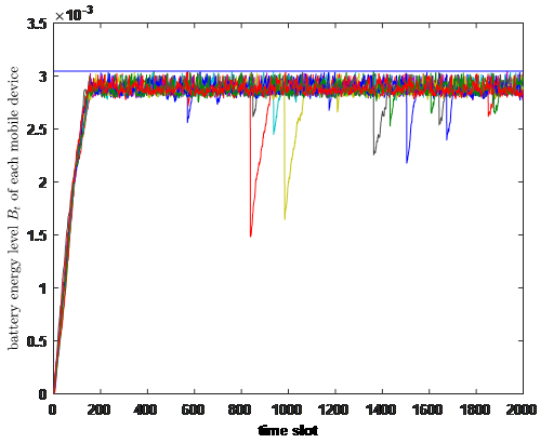

Figure 7: Dynamic resource allocation strategies across different execution modes

Tables 2 and 3 present the computational power and task size results for the RLTOA in terms of cost and delay performance. The speed of computing for edge users is influenced by the time it takes to execute tasks.

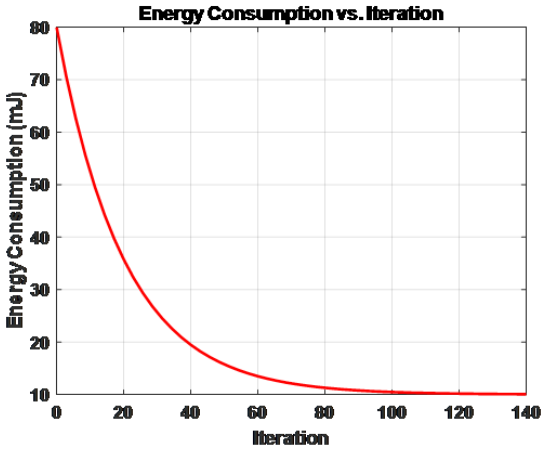

Figure 8: Energy consumption decay of RLTOA

The ability of processing capacity was 2.4 GHz, It is evidenced that the computing power increases when the computation time decreases. As a result agents are monitored as energy-intensive applications. The convergence of RLTOA is shown in figure 8 for the energy-intensive applications. The algorithm operates by learning optimal computational offloading strategies between mobile devices and edge servers through a deep reinforcement learning approach. The energy consumption decays over iterations.

Table 2 and 3 comparative analysis with existing system

Table 2: Total cost performance

Contributers	Algorithm	Avg cost	Avg cost with computational capacity	Avg cost varying with offloading task size
[11]	Local	40.25	70.55	26.50
[18]	Edge	35.20	68.95	20.20
[22]	DQN	24.35	63.50	17.50
Proposed method	RL TOA	20.2	55	12.5

Table 3: Energy consumption performance

Contributers	Algorithm	Avg EC(Joules)	Avg EC with computational capacity	Avg EC varying with offloading task size
[11]	Local	28.36	48.45	49.25
[18]	Edge	22.45	43.25	45.50
[22]	DQN	19.10	39.35	40.87
Proposed method	RLTOA	18.16	37.62	38.25

The suggested RL-based edge computing algorithm's effectiveness was demonstrated by the average total cost associated with computing power and energy consumption. . The numbers in the list are [18, 22, 23, 11]. The suggested RLTOA lowered the offloading expense and average execution cost by 52.13%, 43.5%, and 28.7% in computational cost, task size, and drop. Tables 2 and 3 thoroughly analyzed the impact of the suggested RLTOA when compared DQN, as well as local and edge computing. The results show how battery energy usage and job execution timing are affected by task size and processor power. RLTOA outperforms conventional techniques by 4% and 10%, respectively, during testing. Energy utilization is adequate since it has a major effect on overall expenses and task completion time. The hierarchical architecture for task execution is one of the parameters attributed to the RLTOA method. 2) A new reward function for task delegation in an RL framework. 3) By implementing RLTOA with reduced complexity and minimal processing delay, the overestimation problem is intended to be addressed.

5 Conclusion

The suggested RLTOA approach has been executed with successful outcomes in computation offloading. Offloading facilitation is carried out by the ground edge server, assisting edge users with demanding computations, ensuring the successful fulfillment of all offloading and execution responsibilities. Energy consumption and task execution latency are reduced by offloading the task. The RL framework assists in optimizing costs and computational power by calculating a weighted sum average. An agent achieves optimal results by undergoing intense training and selecting the most effective offloading strategy, while making decisions based on the new reward functions of the suggested RLTOA plan. Finally, the RLTOA convergence is evaluated using simulation. The performance analysis is compared with the DQN, Edge and local system performance.

6 Data availability statement

The raw data of this article will be made available from the authors upon request.

7 Conflict of interest

The authors declare no conflict of interest.

8 References

1. Peng Wei et al, "Reinforcement Learning-Empowered Mobile Edge Computing for 6G Edge Intelligence" IEEE Access, Vol.10, 2022, https://doi.org/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. Netw. Service Manage., vol. 18, no. 3, pp. 3527–3540, Sep. 2021

3. P. Zhou, Y. Xie, B. Niu, L. Pu, Z. Xu, H. Jiang, and H. Huang, "QoEaware 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, Jan. 2021.
4. Y. Kunpeng, H. Shan, T. Sun, R. Hu, Y. Wu, L. Yu, Z. Zhang, and T. Q. S. Quek, "Reinforcement learning-based mobile edge computing and transmission scheduling for video surveillance," IEEE Trans. Emerg. Topics Comput., vol. 10, no. 2, pp. 1142–1156, Apr./Jun. 2021.
5. Samuel, Amalorpava Mary Rajee, Yamuna Devi MM, and S. Madhusudhanan. "Multi-agent Task Assignment in Unmanned Aerial Vehicle Edge Computing based on Deep Learning Approach." 2024 3rd International Conference on Automation, Computing and Renewable Systems (ICACRS). IEEE, 2024.
6. Amalorpava Mary Rajee,S., Merline.A.: Machine Intelligence Technique for Blockage Effects in Next-Generation Heterogeneous Networks,Radio Engineering, Vol.29, Issue 3, Sep 2020.
7. B. Batagelj, L. Pavlovic, L. Naglic and S. Tomazic, ''Convergence of fixed and mobile networks by radio over fibre technology", Informacije MIDEM, vol. 41 (2011), no. 2, pp. 144-149
8. P. Ramamoorthy, K. Ramanathan, ''A Novel Method for 5Generation Multiple-Input, Multiple-Output Orthogonal Frequency-Division Multiplexing using Cauchy Evading Golden Tortoise Adaptive-Bi Directional-Long Short-Term Memory", Informacije MIDEM, vol. 54 (2024), no. 3, pp. 201-213
9. A. Bozorgchenani, F. Mashhadi, D. Tarchi and S. A. Salinas Monroy, "Multi-Objective Computation Sharing in Energy and Delay Constrained Mobile Edge Computing Environments," in *IEEE Transactions on Mobile Computing*, vol. 20, no. 10, pp. 2992-3005, 1 Oct. 2021, https://doi.org/10.1109/TMC.2020.2994232.
10. M. Peng, D. Liang, Y. Wei, J. Li, and H. Chen, "Self-configuration and self-optimization in LTE-advanced heterogeneous networks," IEEE Commun. Mag., vol. 51, no. 5, pp.36-45, May 2013.
11. X. Xia *et al*., "OL-MEDC: An Online Approach for Cost-Effective Data Caching in Mobile Edge Computing Systems," in *IEEE Transactions on Mobile Computing*, vol. 22, no. 3, pp. 1646-1658, 1 March 2023, https://doi.org/10.1109/TMC.2021.3107918
12. T. Mlinar, S. Tomažič, B. Batagelj, Centimeter positioning accuracy in modern wireless cellular networks – wish or reality?", Informacije MIDEM, vol. 53 (2023), no. 4, pp. 239-248
13. Z. Gao, B. Wen, L. Huang, C. Chen, and Z. Su, "Q-learning-based power control for LTE enterprise femtocell networks," IEEE Syst. J., vol. 11, no. 4, pp. 2699-2707, Dec 2017.
14. Roohollah Amiri, Mojtaba Ahmadi Almasi, Jeffrey G. Andrews,Hani Mehrpouyan, "Reinforcement Learning for self organization and power control of Two-Tier Heterogeneous Networks,IEEE Transactions on Wireless Communications, 2019, available online at. arXiv:1812.09778v2 [cs.IT]
15. Mar 2019 15. T. Wang, A. Hussain, L. Zhang, and C. Zhao, "Collaborative edge computing for social internet of vehicles to alleviate traffic congestion," IEEE Trans. Computat. Social Syst., vol. 9, no. 1, pp. 184–196, Feb. 2022.
16. 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.
17. 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, Fourthquarter 2017
18. R. Li, Z. Zhao, X. Zhou, G. Ding, Y. Chen, Z. Wang, and H.Zhang, "Intelligent 5G: When cellular networks meet artificial intelligence," IEEE Wirel. Commun., vol. 24, no. 5, pp. 175-183,Oct 2017.
19. 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, Feb. 2021
20. C.Kai et al. Collaborative cloud-edge-end task offloading in mobile-edge computing networks with limited communication capability" IEEE Trans. Cogn. Commun. Netw. (2020)
21. A.M.R. Samuel and M. Arulraj, "Performance analysis of flexible indoor and outdoor user distribution in urban multi-tier heterogeneous network", Int. J. Mob. Commun, vol. 21, no. 1, pp. 119-133, 2023.
22. Ramesh, Parameswaran, Bhuvaneswari Mohan, Lavanya Viswanath, and Bino Jesu Stephen. "Software Defined Network Architecture Based Network Slicing in Fifth Generation Networks." Informacije MIDEM 54, no. 2 (2024).
23. S. A. M. Rajee, A. Merline, and M. M. Y. Devi, "Game theoretic model for power optimization in next-generation heterogeneous network," Signal Image and Video Processing., vol. 17, no. 7, pp. 3721– 3729, Oct. 2023, https://doi.org/10.1007/s11760-023-02599-8.
24. Zhou, S.; Jadoon, W.; Khan, I.A. Computing Offloading Strategy in Mobile Edge Computing

Environment: A Comparison between Adopted Frameworks, Challenges, and Future Directions. *Electronics* 2023, *12*, 2452. https://doi.org/10.3390/electronics12112452

25. Somesula, M.K., Brahma, B., Raju, M.R. et al. An online approach for cooperative cache updating and forwarding in mobile edge network. Wireless Netw 31, 149–163 (2025). https://doi.org/10.1007/s11276-024-03749-7

Arrived: 24. 05. 2024
Accepted: 02. 04. 2025