İSTANBUL UNIVERSITY CERRAHPAŞA

ORIGINAL ARTICLE
Electrica 2022; 22(3): 387-394

Task Unloading Algorithm for Mobile Edge Computing Based on Artificial Intelligence

Yuan Yuan[1], Mohammad Asif Ikbal[2], Afroj Alam[3]

[1]Department of Engineering Management, Sichuan College of Architectural Technology, Deyang, Sichuan, 618000, China
[2]Department of Electronics and Communication Engineering, Lovely Professional University, Punjab
[3]Department of Computer Science, Sambhram University, Jizzax, Uzbekistan

Cite this article as: Y. Yuan, M. Asif Ikbal and A. Alam, "Task unloading algorithm for mobile edge computing based on artificial intelligence," *Electrica*, 22(3), 387-394, 2022.

ABSTRACT

In order to study the task unloading algorithm for mobile edge computation, this paper proposes an artificial intelligence-based approach. First, a load-unloading model is developed for multi-dependent multi-service nodes within large-scale non-homogeneous mobile edge computing, and then an advanced in-depth training algorithm is used to optimize the task in combination with real-world application options for mobile edge computing. Unloading strategy. Finally, the unloading strategy comprehensively compares energy consumption, cost, load balancing, delays, network operation, and average execution time and analyzes the advantages and disadvantages of each unloading strategy. The simulation results show that the edge algorithm distributes all sub-tasks evenly to all peripheral servers, so the decision to lower central processing unit (CPU) usage to peripheral servers is kept between 20% and 80%, which ensures load balancing performance. As for the Deep Q Network (DQN) algorithm, these two algorithms are better than DQN because the DRQN algorithm and the HERDRQN algorithm are less commonly used on the edge server with the lowest average performance and power ratio, while the CPU utilization is [80%, 100%]. power consumption.algorithm. This proves that the strategy created by the HERDRQN algorithm is scientific and effective in solving the task of unloading the task.

Index Terms—Moving edge computing, task unloading, deep reinforcement learning, short and long term memory network, post experience playback

I. INTRODUCTION

With the rapid development of cloud computing, Internet of things (IoT), and artificial intelligence, new applications such as video analysis, virtual reality, and intelligent vehicles are emerging. The number of wearable devices, household appliances, and sensors is increasing explosively. In the future, the network architecture is changing from centralized to distributed. Although cloud computing can provide centralized computing resources, due to the large amount of data to be transmitted, transmitting all IoT aware mobile terminal data to the cloud data center will bring huge congestion pressure and high delay to the network, which will seriously affect the user experience quality [1] This data processing method of transmitting IoT-aware device data to the cloud data center also brings great risks and challenges in security. The traditional operation mode with cloud data center as the core needs to carry more and more data per unit time, and the data blocking and network delay caused by it greatly affect the quality of user service. On the one hand, because the business data interaction needs to be transmitted through the core network, it will produce great load pressure on the core network; on the other hand, large network delay is caused according to the relative distance between smart devices and cloud data center, which seriously affects the service experience of delay sensitive applications. In order to solve the related problems caused by the above method of transmitting IoT-aware mobile terminal data to the cloud data center, the mobile edge computing (MEC) mode came into being. The MEC mode is to arrange servers near the edge of the IoT mobile devices to form an edge cloud to provide users with computing and storage resources, so as to reduce the share of network resources and shorten the network delay [2]. At present, the edge computing mode has attracted extensive attention in the industry. As shown in Fig. 1, MEC technology includes four aspects: computing offload, resource management, mobility management, and security and privacy protection. As an extension of the mobile cloud computing model, MEC is considered by the industry as a promising solution. Resource management aims to solve the management problem of computing and

Corresponding author:
Yuan Yuan

E-mail: yuanyuan661@163.com

Received: December 14, 2021

Revised: March 24, 2022

Accepted: April 3, 2022

Publication Date: June 3, 2022

DOI: 10.54614/electrica.2022.21171

Content of this journal is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

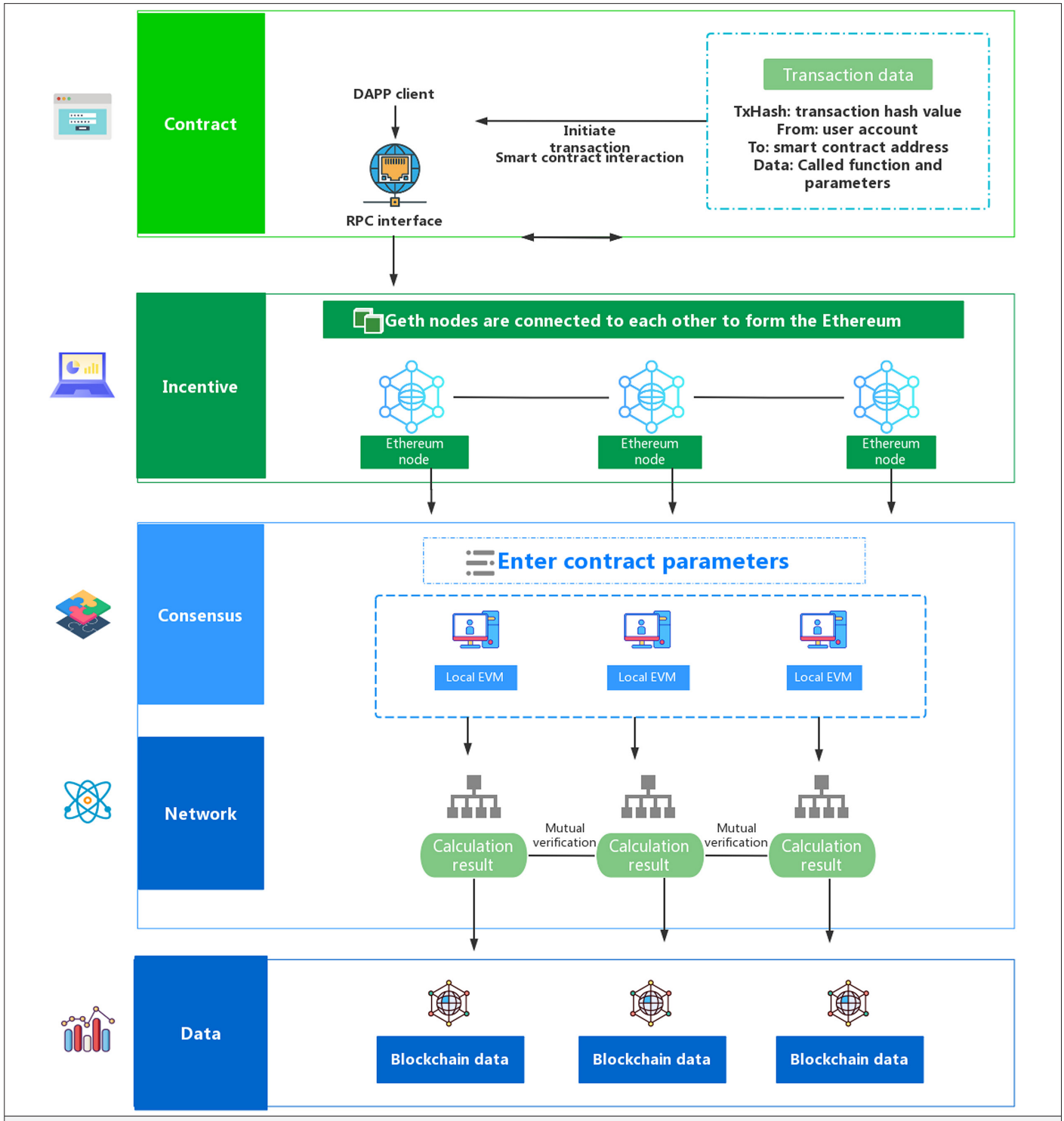


Fig. 1. Mobile edge computing task unloading algorithm.

storage resources in edge computing system. Mobility management is responsible for providing continuous and stable high-quality communication services. Security and privacy protection is an important hub for edge computing. It is responsible for protecting personal information and allowing users to query and process. It is not only an effective scheme to prevent network malicious attacks but also a necessary condition for enterprise intelligent authentication [3].

II. LITERATURE REVIEW

Fang mainly studied the energy efficiency maximization of MEC system with joint task unloading and computing resource allocation and proposed a computing efficiency index, which is solved by iterative and gradient descent methods [4]. Pan proposed an energy-saving task unloading algorithm based on non-orthogonal

multi-channel access MEC environment, which determines the uplink power control solution of network link. The problem of task offload division and time allocation thus is solved. Aiming at the task offload problem that MEC server can charge IoT devices [5], Yang proposed an offload algorithm with time delay constraint, which can minimize the energy consumption [6]. Wang aimed to minimize the energy consumption of IoT device tasks transmitted over the uplink network link based on orthogonal frequency division multiple access protocol. Three optimization strategies such as computing shunting, subcarrier allocation, and computing resource allocation were comprehensively used to reduce the energy consumption of IoT equipment [7]. Chien proposed an IoT equipment unloading scheduling algorithm in the wireless power transmission environment. In this algorithm, the IoT equipment will decide whether to calculate the task locally or unload the task to the MEC server for calculation [8]. Dai mainly studied the collaborative computing offload of MEC and ECS and proposed a collaborative partial computing offload algorithm for MEC and ECS environments. This paper uses the branch and bound method to solve the computing offload problem of a single MEC server and then extends it to multiple MEC servers and ECS scenarios [9]. Chen studied the joint unloading and automatic capacity expansion of energy collection MEC system. This paper points out that the key to the reliable and efficient operation of energy collection MEC lies in its foresight and adaptability. In order to learn quickly when the system parameters are unknown, this paper proposes a reinforcement learning algorithm based on PDS to learn the optimal unloading and automatic capacity expansion strategy [10]. Saeik integrating renewable energy into MEC system, an unloading scheduling algorithm considering MEC battery and service experience quality is proposed. The algorithm proposed in this paper includes the admission control of unloading request and the calculation of MEC server frequency. However, the above algorithms are not suitable for equipping each IoT device with EH module proposed in this paper and green task unloading scheduling for equipment tasks. Of course, the research on IoT system with energy collection function has also attracted the attention of many researchers [11]. Yang proposed an unloading algorithm based on reinforcement learning for energy collection IoT equipment to determine the unloading rate according to the battery power [12]. Chemouil proposed a dynamic computing offload algorithm focusing on the computing power of MEC server. The algorithm only mentions the role of energy collection in prolonging the network life and does not take energy collection as the main consideration of system task offload scheduling [13]. Sun assuming that IoT equipment has energy collection module, proposed an algorithm for IoT equipment to determine its calculation frequency according to the energy collection state using Lyapunov optimization technology. However, this paper only considers the unloading scheduling of a single IoT equipment task and the unloading scheduling of multiple IoT equipment tasks that can use green energy [14].

Based on this, this paper proposes an approach based on AI. First, a load unloading model with a large number of dependent multi-service nodes within a large non-homogeneous mobile market computation is developed and then used to optimize the advanced training algorithm in combination with real-world application options for mobile market computation. Task unloading strategy. Finally, the task unloading strategy comprehensively compared energy consumption, cost, load balance, delay, network operation, and average execution time and analyzed the advantages and disadvantages of each unloading strategy.

III. DESIGN OF DEEP REINFORCEMENT LEARNING ALGORITHM

Reinforcement learning is an algorithmic model that can make optimal decisions through self-study in a specific scenario. It models it by abstracting all real problems into an interactive process between the agent and the environment. At each time step during the interaction, the agent receives the state of the environment and selects the corresponding response action, and then at the next time step, the agent obtains a reward value and a new state based on the feedback from the environment. Reinforcement learning constantly learns knowledge to adapt to the environment according to the rewards received, and all its agents aim to maximize the sum of expected cumulative rewards or expected rewards received at all time steps. Although reinforcement learning has many advantages, at the same time, the method lacks scalability and is essentially limited to rather low-dimensional problems. These limitations exist because reinforcement learning algorithms have the same memory complexity, computational complexity as well as sample complexity in machine learning algorithms.

A. lstm Network

DQA is a deep reinforcement learning algorithm based on value iteration. Its goal is to estimate the Q value of the optimal strategy. The algorithm uses the deep neural network to calculate the approximate value function, turns the q-table update problem into a function fitting problem, and makes it obtain similar output actions according to similar states so as to solve the shortcomings of the traditional Q-learning algorithm in high-dimensional and continuous problems [15]. The calculation result of $\hat{Q}$ function is made to approach Q value by updating parameter θ as shown in the equation below:

$$\begin{cases} \hat{Q}\left(s_t, a_t; \theta\right) \approx Q\left(s_t, a_t\right) \\ Q\left(s_t, a_t\right) := Q\left(s_t, a_t\right) + a\left(r_{t+1} + \gamma \max_{a'} Q\left(s_{t+1}, a'\right) - Q\left(s_t, a_t\right.\right. \end{cases} \tag{1}$$

Wherein, S_{t+1} represents the next state of state a_t after taking action a' in time step T, S_{t+1} is the immediate reward after taking action a_t, and a' is all actions that state S_{t+1} can take; γ is the discount coefficient in the value accumulation process, which determines the importance of future returns relative to current returns; a is the learning rate, and the greater the value, the less the effect of previous training will be retained [16]. Therefore, by maintaining the difference between the two network parameters for a period of time, the loss function is calculated by using the difference between the current Q value and the target Q value, and then the parameters of the main net network are updated reversely by using methods such as random gradient descent. The loss function of DQN algorithm is calculated as:

$$\begin{cases} L\left(\theta\right) = E\left[\left(T\arg etQ - \hat{Q}\left(s_t, a_t; \theta\right)\right)^2\right] \\ T\arg etQ = r_{t+1} + \gamma \max_{a'} \hat{Q}\left(s_{t+1}, a'; \theta^-\right) \end{cases} \tag{2}$$

where $\hat{Q}\left(s_t, a_t; \theta\right)$ represents the output of the current network main net, which is used to calculate the Q value of the current state action pair; $\hat{Q}\left(s_{t-1}, a'; \theta^-\right)$ represents the output of target net, which is used to calculate the target Q value after taking all possible actions. In view of the gradual change of resources in MEC with time and the memory ability of LSTM network for long-time state, this paper proposes to combine LSTM and DQN to deal with the actual

MEC problem dependent on time sequence. By replacing one full connection layer of DQN network with LSTM layer, the cyclic structure is used to integrate long-time historical data to better estimate the current state [17]. z_t and action a_t form a state action pair, which can be integrated with the output value in LSTM to deduce the real environment state s_t and then imported into the deep neural network for training [18]. Therefore, compared with $\hat{Q}(s_t, a_t; \theta)$ used in DQN algorithm, DRQN prefers to use $\hat{Q}(z_t, h_t; \theta)$ for function fitting, in which h_t represents the output value of LSTM layer in the current time step, and its iteration is as follows:

$$h_{t+1} = LSTM(h_t, z_t, a_t) \tag{3}$$

B. Post Experience Playback

HER is a sample data storage structure used to solve the sparse feedback reward. It adjusts the task goal through the progressive learning method to improve the strategy exploration ability of the model. Now it is assumed that the agent will experience the learning process from the initial state s_0 to the target state g, but its termination state is g' at the end of learning, then the generated real learning trajectory can be expressed as:

$$\{(s_0, g, a_0, r_0, s_1), (s_1, g, a_1, r_1, s_2), ..., (s_n, g, a_n, r_n, g')\} \tag{4}$$

where a_n represents the action taken by the agent in time step n, and r_n represents the reward obtained by the agent in time step n [19]. Based on the above assumptions, HER replaces the target state g with the termination state g', which indicates that the agent achieves the goal and obtains effective feedback in the learning process, and the generated imaginary learning track can be expressed as:

$$\{(s_0, g', a_0, r_0, s_1), (s_1, g', a_1, r_1, s_2), ..., (s_n, g', a_n, r_n, g')\} \tag{5}$$

Because the learning objectives of the model are different in each iteration, the selected action will also change. The action obtained according to the current state s_t and the target state g in time step t is calculated as:

$$r_t = \text{Reward}(s_t, a_t, g) \tag{6}$$

Finally, the experience calculated according to the target state G is stored in the experience pool. Each experience based on HER will be composed of five elements: current state s, action s', timely reward R, next state a, and current target g. Meanwhile, during the training process, the experience playback based on HER can generate imaginary target g' through the target sampling strategy. The new reward is calculated in combination with state S_t and action a_t and stored in the experience pool [20] so as to generate some additional training experience, which is calculated as:

$$r' = \text{Reward}(s_t, a_t, g') \tag{7}$$

Among them, the goal sampling strategy adopted in this paper is future, that is, randomly sampling the states after time step T and selecting K states as a new set of imaginary goals. HER makes full use of the idea that human beings obtain useful experience from failure experience and obtains effective rewards by achieving imaginary goals in the learning process through imagination trajectory. In order to ensure that any strategy generated can use feedback reward for learning, in which the agent first reaches the imaginary target state in a small area close to the initial state then gradually explores the surrounding area, uses progressive learning to meet the task target with increasing difficulty, and finally makes the model learn to the actual target state.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

In order to reflect the change of resource utilization of mobile applications in different time periods, this paper uses Google cluster trace data set to simulate the change of utilization of each module over time. In addition, in order to ensure the efficient availability of the strategies generated by each deep reinforcement learning algorithm, this paper first selects 1000 applications from Google data set to train each neural network then selects other data to test the trained network model so as to compare the universality and efficiency of each strategy. Figs. 2-7 are the resource loss diagrams generated by each algorithm in task unloading [21]. It can be seen from the figure that with the growth of the number of applications, the

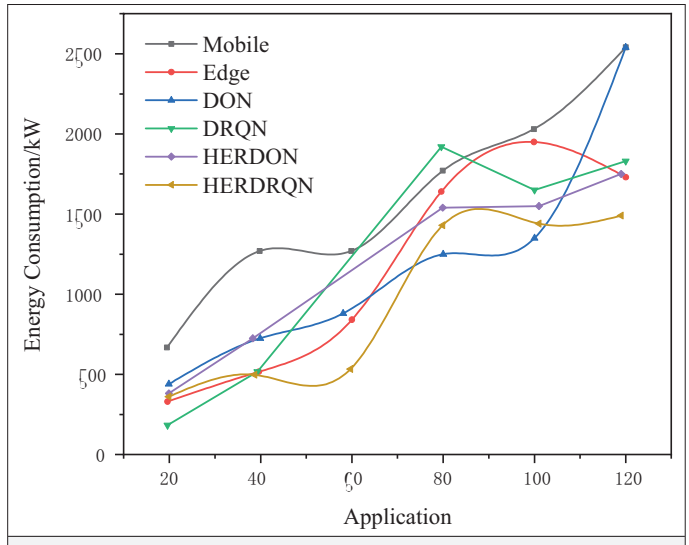


Fig. 2. Energy consumption of different applications.

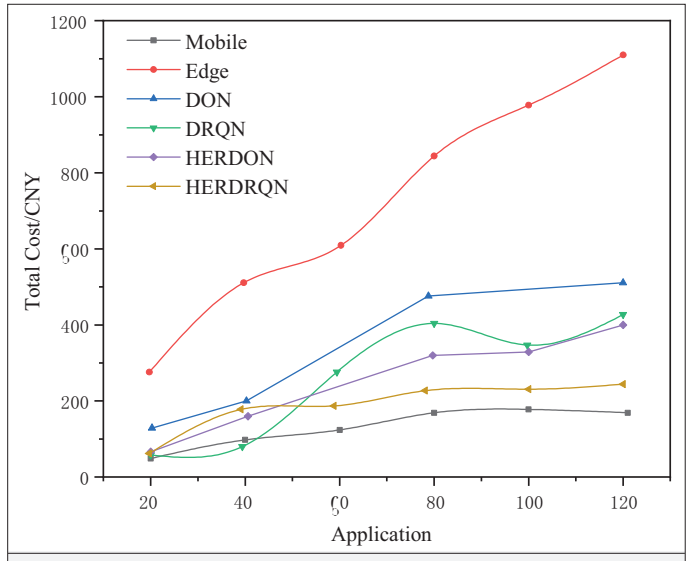


Fig. 3. Cost of different applications.

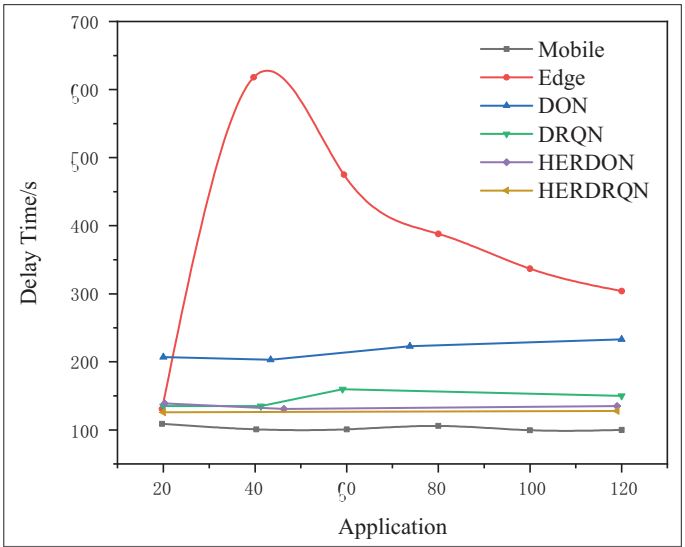


Fig. 4. Delay of different applications.

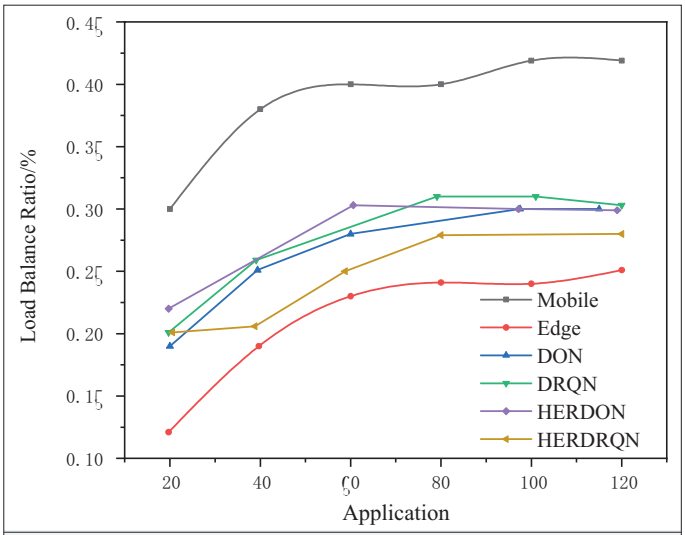


Fig. 5. Load balancing of different applications.

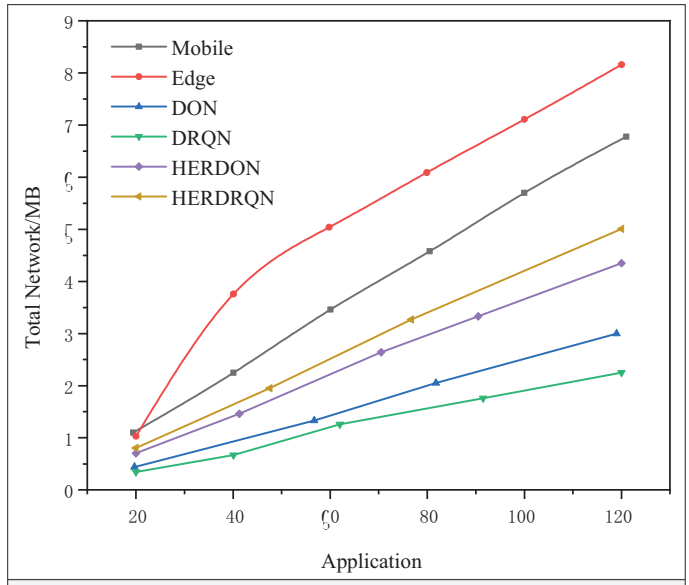


Fig. 6. Total network for different applications.

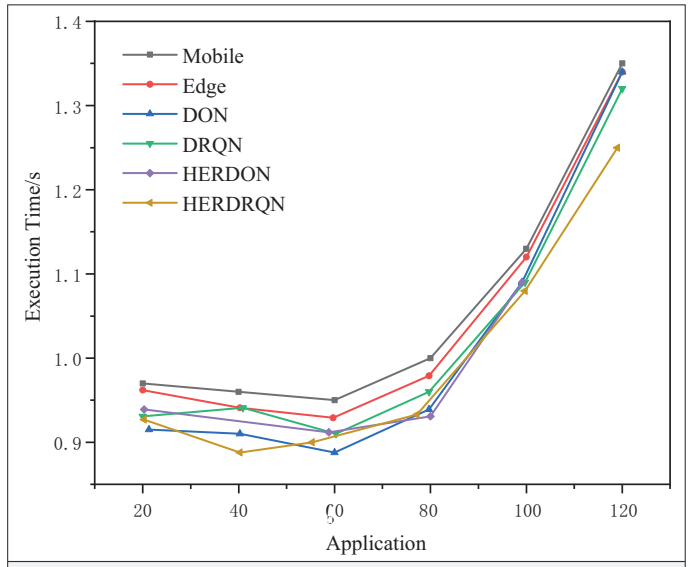


Fig. 7. Execution of different applications.

unloading strategy generated by each algorithm shows an increasing relationship in terms of energy consumption, cost, load balancing, delay, network usage, and execution time. Among them, the unloading strategy based on mobile algorithm can achieve good results in terms of cost and delay, but poor in other aspects. This is mainly because the mobile algorithm offloads the subtasks to the local device for execution first and then gradually offloads to the upper device when the resources are insufficient. Therefore, the algorithm has lower delay at the network level [22]. In addition, because the local device belongs to the user, there is no need to pay the corresponding calculation cost when processing the task, so the cost of the algorithm is very low.

This part of the experiment is mainly used to verify the performance of each algorithm in the actual MEC unloading task. The whole test platform uses three servers and two laptops for experimental simulation, of which two servers are used to simulate edge service nodes in different geographical locations, one server is used to simulate cloud data center, and two laptops are used to simulate users' mobile devices [23] All servers adopted Ubuntu 16.04.6lts operating system, 16 core processor model intelxeone5-2660, 2 gigabit network cards, and 32GB memory.

By freezing the running application and saving its execution state to the disk in the form of snapshot, the software can uninstall the application. Sysstat provides the performance monitoring function of Linux system. It can be used to count the resource utilization of CPU and calculate the energy consumption and cost of server equipment. The unit price of each type of equipment is consistent with Table 1 [24]. At the same time, in order to simulate the dynamic resource changes of different user applications, this experiment uses container technology to build a certain number of web applications:

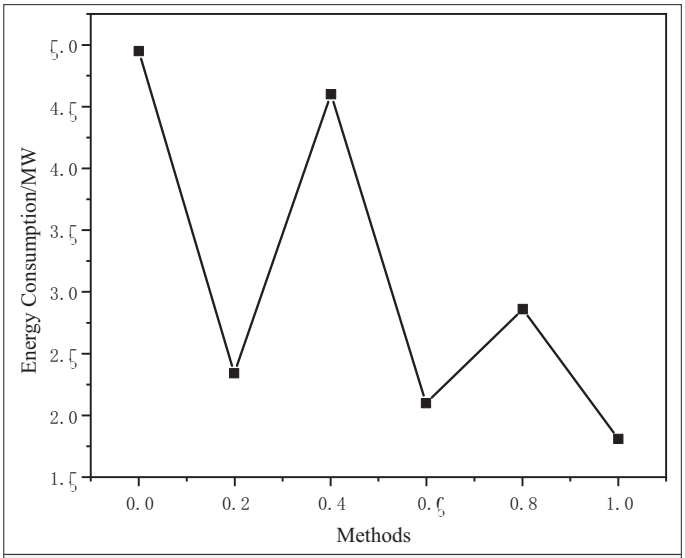


Fig. 8. Comparison of energy consumption of each algorithm in the test environment.

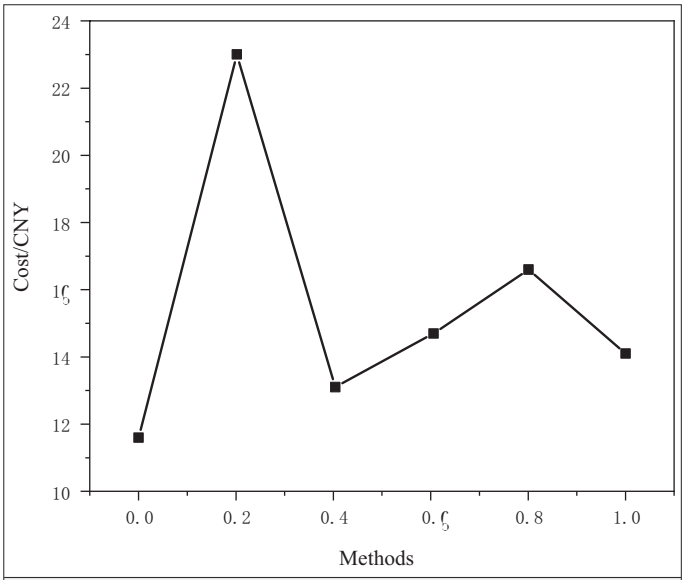


Fig. 9. Cost comparison of algorithms in test environment.

websites using static pages, dynamic websites using SQLite3 database, and dynamic websites using MySQL database and uses web bench website stress test tool to simulate users' continuous request operation on websites [25]. In addition, in order to test the impact of users' geographical location on task unloading strategy, in this experiment, the EUA data set is used to simulate the location change of users and the Linux traffic control tool is used to simulate the transmission delay caused by relative distance. Figs 8-10 show the performance of energy consumption, cost, and delay of each algorithm in this test environment [26]. It can be seen from the figure that the real-world experimental results of each algorithm are basically the same as the simulation experimental results. Finally, HERDRQN algorithm performs best in terms of energy consumption and exceeds most algorithms in terms of cost and delay. Therefore, its comprehensive performance is the best of all algorithms [27-29].

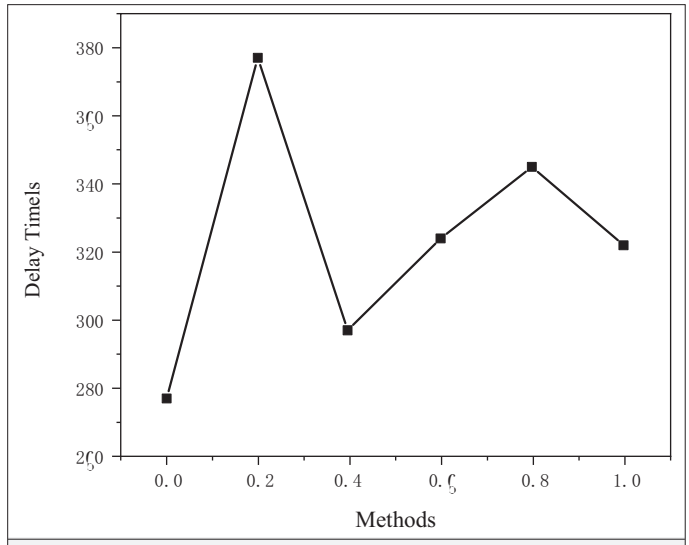


Fig. 10. Delay comparison of algorithms in test environment.

V. CONCLUSION

In order to study the task unloading algorithm for mobile edge computation, this paper proposes an AIs-based approach. The algorithm for unloading the algorithm generated by each algorithm is then tested on the iFogSim peripheral computational simulation platform factors. Comparing the different outcomes of each algorithm, the LSTM network and the improved HERDRQN algorithm based on HER are effective in balancing power consumption, cost, load, and latency, proving that the strategy generated by the HERDRQN algorithm can solve the problem. MEC task. Science and rationale for unloading. Edge intelligence will change the architecture of production and play an important role in future life. Therefore, it is possible to further study the issues and technologies related to MEC unloading.

Peer-review: Externally peer-reviewed.

Author Contributions: Design – Y.Y.; Supervision – Y.Y.; Funding – Y.Y.; Materials – Y.Y.; Data Collection and/or Processing – Y.Y.; Analysis and/or Interpretation – M.A.I.; Literature Review – A.A.; Writing – M.A.I.; Critical Review – A.A.

Declaration of Interests: The authors declare that they have no competing interest.

Funding: The authors declared that this study has received no financial support.

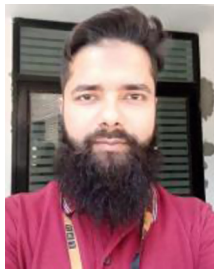
REFERENCES

1. W. J. Chang, L. B. Chen, C. Y. Sie, and C. H. Yang, "An artificial intelligence edge computing-based assistive system for visually impaired pedestrian safety at zebra crossings, " *IEEE Trans. Con. Electron.*, vol.67, no.1, pp.3–11, 2021. [CrossRef]
2. T. Li, J. Yang, and D. Cui, "Artificial-intelligence-based algorithms in multi-access edge computing for the performance optimization control of a benchmark microgrid, " *Phys. Commun.*, vol.44, no.3, p.101240, 2021. [CrossRef]
3. Z. Liu, X. Yang, and J. Shen, "Optimization of multitask parallel mobile edge computing strategy based on deep learning architecture, " *Des. Autom. Embedded Syst.*, vol.24, no.3, pp.129–143, 2020. [CrossRef]
4. J. Fang, J. Shi, S. Lu, M. Zhang, and Z. Ye, "An efficient computation offloading strategy with mobile edge computing for iot, " *Micromachines*, vol.12, no.2, p.204, 2021. [CrossRef]

5. Y. Pan, M. Chen, Z. Yang, N. Huang, and M. Shikh-Bahaei, "Energy-efficient noma-based mobile edge computing offloading, " *IEEE Commun. Lett.*, vol.23, no.2, pp.310–313, 2019. [CrossRef]
6. L. Yang, X. Chen, S. M. Perlaza, and J. Zhang, "Special issue on artificial-intelligence-powered edge computing for internet of things, " *IEEE Internet Things J.*, vol.7, no.10, pp. 9224–9226, 2020. [CrossRef]
7. J. Wang, J. Hu, G. Min, W. Zhan, Q. Ni, and N. Georgalas, "Computation offloading in multi-access edge computing using a deep sequential model based on reinforcement learning, " *IEEE Commun. Mag.*, vol. 57, no.5, pp.64–69, 2019. [CrossRef]
8. W. C. Chien, H. Y. Weng, and C. F. Lai, "Q-learning based collaborative cache allocation in mobile edge computing, " *Future Gener. Comput. Syst.*, vol.102, no.Jan., pp.603–610, 2020. [CrossRef]
9. S. Dai, M. L. Wang, Z. Gao, L. Huang, X. Du, and M. Guizani, "An adaptive computation offloading mechanism for mobile health applications, " *IEEE Trans. Veh. Technol.*, vol.69, no.1, pp.998–1007, 2020. [CrossRef]
10. S. Chen *et al.*, "Internet of things based smart grids supported by intelligent edge computing, " *IEEE Access*, vol.7, no.1, pp. 74089–74102, 2019. [CrossRef]
11. F. Saeik *et al.*, "Task offloading in edge and cloud computing: A survey on mathematical, artificial intelligence and control theory solutions, " *Comput. Netw.*, vol. 195, no.3, p. 108177, 2021. [CrossRef]
12. Z. Yang *et al.*, "Efficient resource-aware convolutional neural architecture search for edge computing with pareto-bayesian optimization, " *Sensors (Basel)*, vol.21, no.2, p.444, 2021. [CrossRef]
13. P. Chemouil *et al.*, "Special issue on artificial intelligence and machine learning for networking and communications, " *IEEE J. Sel. Areas Commun.*, vol.37, no.6, pp.1185–1191, 2019. [CrossRef]
14. Y. Sun et al., "Adaptive learning-based task offloading for vehicular edge computing systems, " *IEEE Trans. Veh. Technol.*, vol.68, no.4, pp.3061–3074, 2019. [CrossRef]
15. C. Zhang, and Z. Zheng, "Task migration for mobile edge computing using deep reinforcement learning, " *Future Gener. Comput. Syst.*, vol.96, no.JUL, pp.111–118, 2019. [CrossRef]
16. R. Zhang, P. Cheng, Z. Chen, S. Liu, Y. Li, and B. Vucetic, "Online learning enabled task offloading for vehicular edge computing, " *IEEE Wirel. Commun. Lett.*, vol.9, no.7, pp.1–1, 2020. [CrossRef]
17. H. Y. Shishido, J. C. Estrella, C. F. M. Toledo, and S. Reiff-Marganiec, "Optimizing security and cost of workflow execution using task annotation and genetic-based algorithm, " *Computing*, vol.103, no.6, pp.1281–1303, 2021. [CrossRef]
18. S. Bhamidipati, K. J. Kim, H. Sun, and P. V. Orlik, "Artificial-intelligence-based distributed belief propagation and recurrent neural network algorithm for wide-area monitoring systems, " *IEEE Network*, vol.34, no. 3, pp.64–72, 2020. [CrossRef]
19. K. Sreenu, and S. Malempati, "Mfgmts: Epsilon constraint-based modified fractional grey wolf optimizer for multi-objective task scheduling in cloud computing, " *IETE J. Res.*, vol.65, no.2, pp. 201–215, 2019. [CrossRef]
20. J. Liu, H. Lu, Y. Luo, and S. Yang, "Spiking neural network-based multi-task autonomous learning for mobile robots, " *Eng. Appl. Artif. Intell.*, vol.104, no.1, p.104362, 2021. [CrossRef]
21. S. Rashidi, and S. Sharifian, "A hybrid heuristic queue based algorithm for task assignment in mobile cloud, "*Future Gener. Comput. Syst.*, vol.68, no.mar., pp.331–345, 2017. [CrossRef]
22. Y. Chen, Y. Yang, C. Liu, C. Li, and L. Li, "A hybrid application algorithm based on the support vector machine and artificial intelligence: An example of electric load forecasting, " *Appl. Math. Modell.*, vol.39, no.9, pp.2617–2632, 2015. [CrossRef]
23. G. Liu, W. Guo, Y. Niu, G. Chen, and X. Huang, "A pso-based timing-driven octilinear steiner tree algorithm forvlsi routing considering bend reduction, " *Soft Comput.*, vol.19, no.5, pp. 1153–1169, 2015. [CrossRef]
24. X. Liu, Y. Ma, G. Huang, J. Zhao, H. Mei, and Y. Liu, "Data-driven composition for service-oriented situational web applications, " *IEEE Trans. Serv. Comput.*, vol.8, no.1, pp.2–16, 2015. [CrossRef]
25. A. N. Escalante-B and L. Wiskott, "Theoretical analysis of the optimal free responses of graph-based sfa for the design of training graphs, " *J. Mach. Learn. Res.*, vol.17, no.157, pp.1–36, 2016.
26. Z. Zhang, F. He, and Y. Zhou, "Bounding volume hierarchy construction algorithm based on dynamic task scheduling, " *Jisuanji Fuzhu Sheji Yu Tuxingxue Xuebao J. Computer-Aided Des. Comput. Graph.*, vol.30, no.3, p. 491, 2018. [CrossRef]
27. Z. Huang, L. He, X. Li, Y. Kang, and D. Xie, "Depth control for storage tank in-service inspection robot based on artificial intelligence control, " *Ind. Robot*, vol.45, no.6, pp.732–743, 2018. [CrossRef]
28. S. Tabandeh, W. Melek, M. Biglarbegian, S. P. Won, and C. Clark, "A memetic algorithm approach for solving the task-based configuration optimization problem in serial modular and reconfigurable robots, ", *Robotica*, vol.34, no.9, pp.1979–2008, 2016. [CrossRef]
29. T. S. Li, P. H. Kuo, Y. F. Ho, M. C. Kao, and L. H. Tai, "A biped gait learning algorithm for humanoid robots based on environmental impact assessed artificial bee colony, " *IEEE Access*, vol.3, no.1, pp.13–26, 2015. [CrossRef]
30. L. Li, Y. Li, and R. Li, "Double auction-based two-level resource allocation mechanism for computation offloading in mobile blockchain application," *Mob. Inf. Syst.*, vol. 2021, pp. 1–15, 2021. [CrossRef]
31. F. Xu, and W. Chu, "Sports dance movement assessment method using augment reality and mobile edge computing," *Mob. Inf. Syst.*, vol. 2021, pp. 1–8, 2021. [CrossRef]

Yuanyuan (1981.04), Chengdu City, Sichuan Province, female, master's degree, associate professor, mainly engaged in computer application and algorithm design and analysis. Presided over 1 project of Sichuan Provincial Department of education "Research on the method of BIM model based on genetic algorithm in construction progress" and 1 project at the institute level. He wrote one book alone and edited one textbook. There are 16 utility model patents and 30 soft works.

Afroj Alam is an assistant professor of computer science department in Sambhram University. It is located at Jizzax Uzbekistan. Prior to this, he was Assistant professor in Bakhtar University, Kabul Afghanistan. Currently he is a Ph.D. Scholar in Integral University, Lucknow, India. His area of research is data mining technique with machine learning algorithm in smart healthcare for prediction of disease. We have published some book chapter and research papers in good publisher and journal. He has done his master's and bachelor degree in computer science.



Mohammad Asif Ikbal is working as an Assistant Professor in Lovely professional University, Punjab, with more than 10 years of teaching experience. His research areas are very vast and he has contributed in various emerging fields.