Task Offloading of Edge Computing Based on Reinforcement Learning: A Survey

Pu Yangyizhen, Xie Yijie, Zheng Tongzhou, Zhou Xufei

***Abstract*—abstract here**

***Index Terms*—Edge Computing, Task Offloading, Reinforcement Learning, Multi-Agent Reinforcement Learning.**

I. Introduction

EDGE computing refers to a distributed and open platform that integrates the core capabilities of networking, computing, storage, and applications on the network edge side close to the data source. It provides edge intelligence services nearby to meet the key requirements of industry digitalization in aspects such as agile connection, real-time services, data optimization, application intelligence, security, and privacy protection [1]. Traditional cloud computing generally centralizes data in data centers for processing, while edge computing assigns some computing tasks to the network edge for processing, handling data where it is generated.

The importance of edge computing is becoming increasingly prominent. Taking the booming autonomous driving in recent years as an example, autonomous driving requires extremely high real-time performance. If traditional cloud computing is used, the delay caused by vehicles transmitting data to the data center for processing and then getting the results back is unacceptable and may even lead to traffic accidents. However, edge computing enables vehicles to process data locally and make timely decisions, ensuring real-time performance.

Meanwhile, edge computing has also become important in terms of data privacy protection. For instance, many sensitive data (such as medical and health data, financial transaction data, etc.) are not suitable for being transmitted to remote cloud for processing. Edge computing allows data processing and analysis to be carried out on local devices or edge servers, reducing the risk of data leakage.

Although edge computing is gradually becoming a key technology to support various emerging applications and business models, with the development of mobile devices and the Internet of Things, the amount of data that devices need to handle and computing tasks are becoming increasingly complex. To further improve the processing efficiency, task offloading has emerged.

Task offloading means transferring the computing tasks of a device to other devices or servers with stronger computing capabilities for processing. It is common to offload the tasks of mobile devices to edge servers or the cloud [2].

However, in actual dynamic environments, task offloading faces numerous challenges:

- **Latency Challenge**: In dynamic environments, the network state can change at any time. Situations such as network congestion and unstable signals may lead to a significant increase in data transmission latency. For example, in the Internet of Vehicles scenario, vehicles are in a high-speed moving state and network connections are constantly switching. During the process of offloading tasks from vehicles to edge servers or the cloud, it is difficult to predict and control the data transmission latency [3].
- **Energy Consumption Challenge**: Although offloading tasks to external computing resources can utilize stronger computing capabilities, the energy consumption issue during the task offloading process also needs to be considered. Packaging and transmitting data as well as interacting with external resources all consume energy. Especially for mobile devices with limited battery power, an unreasonable task offloading strategy may cause a sharp increase in device energy consumption and shorten the device's battery life [4].
- **Device Heterogeneity Challenge**: The development of mobile devices and the Internet of Things has led to an increasingly diverse range of devices, with huge differences in hardware architectures, computing capabilities, storage capacities, and so on. This heterogeneity has caused two main problems.
  - On the one hand, the software and hardware environments of different devices are different. If the offloaded tasks are to run smoothly on the target devices, complex compatibility adaptation work is required.
  - On the other hand, the performance and computing capabilities of different devices also vary. If the same tasks are assigned to each device, it will lead to a decline in performance.

  Therefore, tasks need to be reasonably allocated according to the actual situation of the devices.

Faced with a series of complex challenges such as latency, energy consumption, and device heterogeneity that task offloading encounters in dynamic environments, traditional methods based on rules or static optimization are gradually proving to be inadequate. These traditional methods often struggle to cope with the dynamic changes of the environment and complex system constraints, and thus cannot achieve optimal performance.

Against this background, reinforcement learning, as a powerful machine learning technique, has provided highly promising ideas and methods for solving the problem of task offloading.

The core of reinforcement learning lies in the fact that an agent interacts with the environment, continuously tries different actions, and learns the optimal strategy based on the obtained reward feedback, so as to maximize the cumulative reward in the long term [5]. This learning mode is inherently suitable for the dynamically changing task offloading scenarios because it can dynamically adjust task offloading decisions according to the real-time environmental states (such as network conditions, device loads, etc.), achieving self-adaptation to complex environments.

For example, when dealing with the latency issue, the reinforcement learning algorithm can, through continuous trial-and-error learning, choose to offload tasks to local devices, edge servers, or the cloud according to the real-time latency situation of the current network, so as to ensure that the overall task execution latency is minimized [6].

Through continuous learning and optimization, reinforcement learning can help agents find near-optimal task offloading strategies, providing a powerful technical support for solving the difficulties faced by task offloading in dynamic environments. It is expected to become a key tool for promoting the development of edge computing task offloading technology.

II. Research Status of Reinforcement Learning in Task Offloading

A. The Foundation of Reinforcement Learning

B. Applications of Classical Reinforcement Learning

C. Applications of Deep Reinforcement Learning

In the research on task offloading, enhanced reinforcement learning methods have also been continuously developed and innovated. Here, we mainly introduce two methods. One is the Double Dueling DQN (D3QN), which demonstrates unique advantages in solving offloading problems in highly dynamic environments. The other is Multi-Agent Reinforcement Learning (MARL), which plays an important role in edge collaboration scenarios.

Edge computing networks face difficulties such as limited device resources and dynamic environmental changes. Zhang et al. [7] modeled the online offloading problem as a Markov Decision Process (MDP) and, on this basis, proposed the C - D3QN algorithm [7]. This algorithm combines the D3QN algorithm with a clustering algorithm and effectively improves the decision-making performance in highly dynamic environments by introducing a double dueling mechanism.

In terms of algorithm design, the centralized agent collects information on edge servers, mobile terminal devices, and channel states to form the state space. The action space covers the task execution decisions of terminal devices, and the reward function combines incentives such as task completion latency, terminal energy consumption, and task success rate to guide the agent's decisions.

When it comes to exploration and action selection, the agent adopts the ε-greed method. In the early stage, it randomly explores with a high probability, and in the later stage, it selects the action with the maximum Q value. The C - D3QN model introduces a fixed target mechanism, initializing the online learning and target networks. The online learning network updates parameters in real time, and the target network is periodically updated to the parameter values of the online learning network to stabilize training and avoid overestimation of Q values. Meanwhile, the dueling mechanism changes the network output from a single Q value to a value function V value and an advantage function A value, enabling a more accurate estimation of Q values and enhancing the adaptability to highly dynamic environments.

Experiments show that the C - D3QN algorithm has obvious advantages over DQN in complex dynamic scenarios. It has a faster convergence speed and can quickly adapt to environmental changes for decision-making. In terms of task completion latency, it selects appropriate execution methods according to tasks and resources to reduce latency. Regarding terminal energy consumption, it makes reasonable offloading decisions to avoid excessive energy consumption and extend the battery life.

As the scale of edge computing expands, the collaboration among edge nodes becomes increasingly important. Multi-Agent Reinforcement Learning (MARL) can better cope with complex edge collaboration scenarios through the interaction and collaboration among multiple agents.

In edge computing task offloading, different edge servers or devices can be regarded as different agents. They work together to complete the offloading tasks while also taking into account factors such as the communication costs and differences in computing capabilities among different devices and servers [8]. The MARL approach allows each agent to make decisions based on its own observations and local information, and at the same time communicate and collaborate with other agents to gradually achieve the globally optimal collaboration strategy.

However, MARL also faces challenges. Conflicts of interest among agents may lead to difficulties in collaboration, and an increase in the number of agents causes the state and action spaces to grow exponentially, resulting in the "curse of dimensionality" and increasing the computational complexity and convergence difficulty. To address these challenges, researchers have proposed various solutions. A reasonable reward mechanism can adjust the reward function, rewarding agents that actively collaborate and punishing selfish behaviors, prompting them to pursue their own interests while achieving the goals of the system [9]. The hierarchical MARL architecture divides agents into different layers, with higher-level agents coordinating lower-level ones, reducing complexity and improving collaboration efficiency [10].

In practical applications, MARL algorithms can support edge devices to coordinate the allocation of task shares and resource utilization in real time. They can also assist vehicles and Roadside Units (RSUs) in sharing traffic information and facilitating task offloading. In the future, MARL algorithms are also expected to play a significant role in the Internet of Things.

In conclusion, the applications of Double Dueling DQN and Multi-Agent Reinforcement Learning in the field of edge computing task offloading provide powerful means to solve the problems in highly dynamic environments and edge col-

laboration. However, they also face numerous challenges at the same time. Future research needs to further optimize the algorithms to adapt to the continuously developing demands of edge computing and promote the application and development of edge computing technology in a broader range of fields.

III. Optimization and Innovation of Reinforcement Learning Methods

A. Method optimization

The task offloading technology plays a crucial role in edge computing systems, directly impacting both system efficiency and user experience. However, in real-world applications, task offloading faces several challenges, particularly in the context of delay-sensitive tasks. In areas such as autonomous driving and healthcare, task deadlines are stringent, and any delays may result in serious consequences. Additionally, the heterogeneity of devices and tasks introduces significant complexity to task offloading. Differences in resource capacities, task complexity, and data characteristics complicate the formulation of a unified offloading strategy. Achieving a balance between generic policies and individualized requirements remains a significant challenge, as centralized policy training, despite its low overhead, lacks specificity, while individualized policies incur high training costs. The cold-start problem further exacerbates these issues; newly introduced devices may not be covered by existing policies, requiring retraining and adding to system overhead.

To address these challenges, several approaches have been proposed. The DOQO method, as introduced by Chen et al., enhances real-time performance and optimizes resource utilization by dynamically adjusting offloading decisions [11]. The Transfer Reinforcement Learning (TRL) framework, developed by Shuai et al., combines transfer learning and domain adaptation to reduce training costs and improve model adaptability [12]. The C-D3QN algorithm, proposed by Zhang et al., uses K-means clustering and deep reinforcement learning to optimize task offloading, significantly enhancing exploration efficiency [13]. Robles-Enciso et al. introduced the Multi-Layer Reinforcement Learning (ML-RL) system, which improves task offloading efficiency through cooperation among the edge, fog, and cloud layers [14]. Wang et al. proposed MRLCO, a meta-weighted learning-based system, which rapidly adapts to dynamic environments while maintaining high sample efficiency and stable performance. [15]

1) DOQO: DOQO is a real-time, sense-dependent task offloading method that primarily addresses the task offloading problem in dynamic environments. The method optimizes the task offloading process by modeling mobile applications as directed acyclic graphs (DAGs) to capture inter-task dependencies and parallelism.DOQO employs a deep Q-network (DQN) algorithm to evaluate the Q-value of the offloading decision and dynamically adjust the offloading strategy. A distinguishing feature of DOQO is its approach to task prioritization, which eschews preset priorities in favor of a flexible scheduling mechanism that responds to real-time changes in the environment. This flexibility enables the system to swiftly generate offloading plans and make timely adjustments, a capability that traditional methods often lack.

However, DOQO presents several challenges. The uncertainty in task scheduling and execution location leads to an expansive solution space, increasing computational complexity. Conflicting priorities may also hinder the efficient execution of parallel tasks. Furthermore, DOQO's disregard for wireless channel fading effects may limit its effectiveness in real-world mobile environments. Despite these limitations, DOQO remains a viable method, and further optimization is required, particularly in its adaptability to complex environments. [11]

2) TRL Framework: The TRL framework proposes a task offloading method that combines migration learning with domain adaptive mechanisms. The objective of this combination is to reduce training overhead and improve model adaptation. The framework is composed of two modules: a data alignment module and a reinforcement learning module. The data alignment module maps data from disparate devices to the reproducing kernel Hilbert space (RKHS) through the domain adaptation mechanism, thereby reducing data distribution differences and enhancing the sharing capability among devices. The reinforcement learning module employs an Actor-Critic architecture, enabling devices to make globally optimal decisions under limited observation conditions and optimize strategies through feedback. The incorporation of Migration Learning has been demonstrated to expedite the convergence of Deep Reinforcement Learning (DRL) models, thereby facilitating the rapid adaptation of devices to novel environments and enhancing execution efficiency.

While the TRL framework effectively improves adaptability and convergence, there is room for improvement, particularly in adapting to high-mobility devices and optimizing resource utilization with varying task granularity. Thus, while the TRL framework presents an effective solution for dynamic task offloading, further enhancements are necessary for its practical implementation. [12]

3) C-D3QN Algorithm: The C-D3QN algorithm combines K-means clustering with deep reinforcement learning (DRL) to optimize task offloading in dynamic environments. It uses a centralized decision-making agent to collect global state information, ensuring globally optimal decisions. This centralized framework enables efficient coordination among devices, addressing the challenge of suboptimal local decisions common in traditional methods.C-D3QN leverages K-means clustering to partition devices, reducing the action space and enhancing exploration efficiency, especially with a larger number of devices or tasks. By compressing the decision space, it facilitates more effective resource allocation, particularly when task requirements or device resources are complex.The reward function in C-D3QN accounts for energy consumption and penalties, dynamically adjusting offloading decisions based on task load and device location, thereby optimizing resource allocation. This flexibility allows the system to adapt to environmental changes, improving overall efficiency.

Despite its benefits, C-D3QN is limited by its reliance on a coarse-grained task model, hindering fine-grained offloading optimization. It also lacks support for continuous action space models, which limits its applicability in some scenarios. Future research should incorporate fine-grained task models and policy gradient algorithms to further optimize resource utilization

and performance. [13]

4) ML-RL: The ML-RL system enhances task offloading efficiency by leveraging collaborative agents across edge, fog, and cloud layers. Each layer is responsible for tasks with varying latency and computational power requirements: the edge layer handles low-latency tasks, the fog layer processes medium-latency tasks, and the cloud layer manages tasks requiring high computational power. This collaboration allows lower-layer agents to delegate offloading decisions to upper-layer agents, ensuring globally optimal decisions when local knowledge is insufficient.

The ML-RL system integrates greedy methods with reinforcement learning to improve offloading efficiency and convergence speed. Greedy methods, while providing suboptimal solutions through heuristic rules, rely heavily on real-time information, which can result in unstable execution. In contrast, reinforcement learning enables edge devices to improve their offloading strategy over time by interacting with the environment, maximizing long-term rewards. However, the system's inability to support task decomposition and parallel processing limits its applicability in complex scenarios. Enhancing task granularity processing and exploring more flexible policy learning algorithms would further improve system performance. [14]

5) MRLCO: MRLCO is an innovative task offloading solution leveraging Meta-Reinforcement Learning (MRL) and sequence-to-sequence (seq2seq) neural networks. The core innovation of MRLCO lies in its ability to quickly adapt to new environments with minimal data, reducing training time and computational overhead. This makes it significantly more efficient than traditional reinforcement learning methods.

The method models task offloading as multiple Markov Decision Processes (MDPs) and transforms the decision-making into a sequence prediction problem. Seq2seq networks are used to enhance decision flexibility, while an attention mechanism addresses information loss, improving decision-making accuracy and robustness. This flexibility allows MRLCO to adapt to dynamic challenges such as network fluctuations and device disconnections, ensuring accurate offloading decisions.Additionally, MRLCO features an adaptive client selection algorithm that filters underperforming clients, improving system stability and resource utilization. This ensures high offloading performance even in unstable environments.

Despite its advantages, MRLCO's performance still falls short of the optimal solution. Future research should explore more efficient non-strategic MRL techniques to further enhance decision-making and training efficiency. [15]

To further optimize existing methods, it is essential to develop fine-grained task offloading models. For instance, task decomposition based on Directed Acyclic Graphs (DAGs) could enhance decision-making capabilities when combined with policy gradient learning. [13] Additionally, research should focus on addressing the impact of high device mobility and dynamic network environments on task offloading performance to improve algorithm adaptability in real-world scenarios [11], [12]. Lastly, multi-objective optimization techniques should be prioritized to balance delay, energy consumption, and resource utilization, ultimately enhancing offloading efficiency and system stability in edge computing environments. [13]

B. Compression of action space and state space

IV. Current Challenges and Future Prospects

See for resources on formatting math into text and additional help in working with LaTeX.

A. Challenges

The application of reinforcement learning (RL) for edge computing task offloading encounters several challenges in practical settings. These challenges primarily involve the design of scenarios, algorithmic complexity, mobility issues, device and network heterogeneity, security concerns, and communication interference, among others.

1) Scenario Design Challenges: One of the primary challenges in applying RL to edge computing task offloading lies in scenario design [16]. While numerous studies rely on simulations and analyses based on static scenarios, real-world scenarios are inherently dynamic. Factors such as device mobility, network bandwidth fluctuations, and variations in user behavior can affect task loads, revealing the limitations of static offloading decisions. In contrast, RL methods, known for their adaptability in dynamic environments, offer a promising solution through the implementation of adaptive strategies. However, the ability to effectively model and adapt to environmental changes, ensuring that offloading decisions can be made in real time without introducing delays or inefficiencies, remains a critical problem that requires further development.

2) Algorithm Complexity and Latency Problem: The complexity of RL algorithms and the associated latency are significant challenges in edge computing task offloading [17]. In time-varying environments, RL often requires substantial computational resources and time to explore the high-dimensional action space, leading to slow processing speeds and delayed convergence during training. This issue is particularly problematic in real-time edge computing applications, where latency is a critical concern. Overly complex algorithms can increase computation times for offloading decisions, undermining system responsiveness. Therefore, the development of more efficient RL algorithms that reduce computational overhead while maintaining offloading effectiveness is crucial to improving the overall efficiency of task offloading in edge computing systems.

3) Mobility Challenges of Terminal Devices: The mobility of terminal devices presents significant challenges, particularly when these devices move across different service areas, requiring seamless task offloading and system stability [18], [19]. Devices may experience network fluctuations, bandwidth variations, and other issues during movement, which can negatively impact offloading efficiency. RL has the potential to address these challenges by enabling dynamic adjustments to offloading strategies. However, a critical issue remains the ability to maintain both timeliness and accuracy in offloading decisions, especially in environments characterized by device mobility and network fluctuations. Ensuring the adaptability and robustness of RL models is essential to enable effective task offloading in diverse network conditions.

4) Heterogeneous Network and Device Compatibility Issues: Heterogeneous device and network environments represent a significant obstacle for RL-based task offloading in edge computing [19]. The rapid proliferation of new devices and network topologies, driven by advancements in IoT and 5G technologies, has rendered traditional offloading strategies less effective due to the diversity of devices and networks. RL-based offloading solutions must address these compatibility issues to ensure the effectiveness of the offloading process. Developing adaptable and compatible offloading schemes that maintain efficiency across a wide variety of devices and network environments is a key challenge in current RL research for edge computing.

5) Security Challenges: Security remains a critical concern in edge computing task offloading [17]–[19]. The distributed nature of edge computing environments introduces vulnerabilities that can be exploited by malicious actors, especially in multi-tenant settings where system failures at a single point can lead to cascading effects. Ensuring the security of data during the offloading process is paramount to prevent data leakage or tampering. The integration of security measures into RL-based offloading decision-making processes is essential to mitigate potential risks and enhance system resilience against adversarial attacks. Addressing these concerns is crucial for the practical deployment of RL in edge computing systems.

6) Communication Interference and Resource Management: The increasing number of devices in edge computing environments has led to a rise in network interference, which exacerbates the challenges of task offloading due to resource contention and communication congestion [18]. RL offers a promising approach to mitigate these issues by enabling dynamic resource allocation. However, a major challenge remains in optimizing resource distribution across multiple devices and complex network environments to ensure successful task offloading and improve quality of service. Developing RL algorithms that effectively balance offloading, resource allocation, and interference management is essential to maintain system stability and efficiency.

The challenges outlined above are inherently linked to the practical implementation of RL-based task offloading in edge computing. Addressing these issues will significantly enhance the feasibility and efficiency of RL-based task offloading systems. Solving these challenges not only provides theoretical grounding but also offers technological advancements, paving the way for broader adoption of edge computing technologies in practical applications.

B. Future Prospects

V. Conclusion

The conclusion goes here.

References

[1] M. Satyanarayanan, "The emergence of edge computing," *Computer*, vol. 50, no. 1, pp. 30–39, 2017.

[2] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Transactions on Wireless Communications*, vol. 16, no. 3, pp. 1397–1411, 2017.

[3] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[4] X. Chen and G. Liu, "Energy-efficient task offloading and resource allocation via deep reinforcement learning for augmented reality in mobile edge networks," *IEEE Internet of Things Journal*, vol. 8, no. 13, pp. 10 843–10 856, 2021.

[5] R. S. Sutton and A. G. Barto, "Reinforcement learning: An introduction," *IEEE Transactions on Neural Networks*, vol. 9, no. 5, p. 1054, 1998.

[6] S. Nath and J. Wu, "Deep reinforcement learning for dynamic computation offloading and resource allocation in cache-assisted mobile edge computing systems," *Intelligent and Converged Networks*, vol. 1, no. 2, pp. 181–198, 2020.

[7] Z. Zhang, H. Li, Z. Tang, D. Gu, and J. Zhang, "A clustering offloading decision method for edge computing tasks based on deep reinforcement learning," *New Generation Computing*, vol. 41, pp. 85–108, 2023. [Online]. Available: https://doi.org/10.1007/s00354-022-00199-7

[8] X. Huang, S. Leng, S. Maharjan, and Y. Zhang, "Multi-agent deep reinforcement learning for computation offloading and interference coordination in small cell networks," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 9, pp. 9282–9293, 2021.

[9] R. Zhang, F. R. Yu, J. Liu, T. Huang, and Y. Liu, "Deep reinforcement learning (drl)-based device-to-device (d2d) caching with blockchain and mobile edge computing," *IEEE Transactions on Wireless Communications*, vol. 19, no. 10, pp. 6469–6485, 2020.

[10] W. Hou, H. Wen, H. Song, W. Lei, and W. Zhang, "Multiagent deep reinforcement learning for task offloading and resource allocation in cybertwin-based networks," *IEEE Internet of Things Journal*, vol. 8, no. 22, pp. 16 256–16 268, 2021.

[11] X. Chen, S. Hu, C. Yu, Z. Chen, and G. Min, "Real-time offloading for dependent and parallel tasks in cloud-edge environments using deep reinforcement learning," *IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS*, vol. 35, no. 3, pp. 391–404, MAR 2024.

[12] K. Shuai, Y. Miao, K. Hwang, and Z. Li, "Transfer reinforcement learning for adaptive task offloading over distributed edge clouds," *IEEE TRANSACTIONS ON CLOUD COMPUTING*, vol. 11, no. 2, pp. 2175–2187, APR-JUN 2023.

[13] Z. Zhang, H. Li, Z. Tang, D. Gu, and J. Zhang, "A clustering offloading decision method for edge computing tasks based on deep reinforcement learning," *NEW GENERATION COMPUTING*, vol. 41, no. 1, pp. 85–108, MAR 2023.

[14] A. Robles-Enciso and A. F. Skarmeta, "A multi-layer guided reinforcement learning-based tasks offloading in edge computing," *COMPUTER NETWORKS*, vol. 220, JAN 2023.

[15] J. Wang, J. Hu, G. Min, A. Y. Zomaya, and N. Georgalas, "Fast adaptive task offloading in edge computing based on meta reinforcement learning," *IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS*, vol. 32, no. 1, pp. 242–253, JAN 1 2021.

[16] Y. Tian, Z. Liu, K. Zhang, Z. Li, and Y. Xie, "A survey on task offloading techniques in cloud-edge resource collaboration," *Computer Science and Exploration*, vol. 17, no. 10, pp. 2325–2342, 2023.

[17] J. Lv, J. Zhang, Z. Zhang, and C. Gan, "A survey on offloading strategies in mobile edge computing," *Mini-Micro Computer Systems*, vol. 41, no. 09, pp. 1866–1877, 2020.

[18] R. Xie, X. Lian, Q. Jia, T. Huang, and Y. Liu, "A survey on mobile edge computing offloading techniques," *Journal of Communications*, vol. 39, no. 11, pp. 138–155, 2018.

[19] Y. Zhang, Y. Liang, M. Yin, H. Quan, T. Wang, and W. Jia, "A survey of computation offloading schemes in mobile edge computing," *Journal of Computer Research and Development*, vol. 44, no. 12, pp. 2406–2430, 2021.