Intelligent Mobile Edge Computing Systems with Reinforcement Learning-Based Innovations

Abstract

Mobile edge computing often suffers from the dynamic and unknown nature of the environment, such as time-varying conditions, heterogeneous devices, and frequent communication requests, imposing significant challenges on improving system performance. To meet the rapidly growing demands of computation-intensive and time-sensitive applications, Reinforcement learning (RL)[1] has been proposed as an effective tool to establish low-latency and energy-efficient networks. RL enables network entities to interact with the environment and learn an optimal decision-making policy, usually modeled as a Markov decision process (MDP)[2].

***Terms*—** Mobile edge computing (MEC), Resource Management, Computation Offloading, Partially Observable MDP (POMDP), Deep RL (DRL), Multi-Agent RL (MARL), Meta RL, Federated RL

Introduction: MEC is emerging as a promising paradigm to enhance the computational capacity of mobile devices by offloading tasks to nearby edge servers [3]. This paradigm aims to reduce latency, energy consumption, and improve quality of experience for end-users. However, one of the major challenges in MEC is the efficient decision-making process for resource management, considering the dynamic nature of the network, user demands, and limited resources. Traditional offloading strategies, which often rely on heuristic or single-agent models, fail to capture the complexity and stochastic nature of modern MEC.

Background: To cope with the dynamic nature of the network, state-of-the-art research has proposed several task offloading algorithms using RL-based methods, which hold promises to determine optimal decision-making policies by capturing the dynamics of environments and learning strategies for accomplishing long-term objectives [4]. To minimize energy consumption, Munir *et al.* [5] developed a semi-distributed approach using a multi-agent RL (MARL) framework for self-powered MEC. Gong *et al.* in [6] proposed a DRL-based network structure in the industrial IoT (IIoT) systems to jointly optimize task offloading and resource allocation to achieve lower energy consumption and decreased task delay. Sun *et al.* in [7] explored both computation offloading and service caching problems in MEC. They proposed a hierarchical DRL framework, which effectively handles both problems under heterogeneous resources.

Although DRL-based methods in some releated research like [5]–[7] have demonstrated effectiveness in handling network dynamics, resource management still encounters several challenges that require further attention:

1. Single-agent **non-stationarity** issues [8], which motivate leveraging **MARL**.
2. **Partially observable environment** issues, which motivate the use of decentralized **POMDP**.
3. The **dynamic and heterogeneous nature of network** environments, motivates the use of **Meta RL**.
4. **Decentralization of learning process**, where data privacy, transmission link, and computational resources are critical concerns, motivate leveraging **Federated RL**.

[1]The author is with the EdgeAI Laboratory at the Department of Electrical Engineering, Sharif University of Technology, Tehran, Iran (Mail, Github)

Multi-Agent Deep Reinforcement Learning for Cooperative Task Offloading in Partially Observable Mobile Edge Computing Environment

Motivation: In MEC, each entity may need to make local decisions to improve network performance in dynamic and uncertain environments. Standard learning algorithms, such as single-agent RL or DRL [9], [10], have recently been used to enable each network entity to learn an optimal decision-making policy adaptively through interaction with the unknown environment. However, these algorithms fail to model cooperation or competition among network entities, treating other entities simply as part of the environment, which can lead to non-stationarity issues. MARL enables each network entity to learn its optimal policy by observing both the environment and the policies of other entities while interacting with a shared or separate environment to achieve specific objectives [11].

Problem Statement: Task offloading is a critical process to efficiently assign available resources to task requests, for high-performance, reliable, and cost-effective services. In MEC, the decision-making process of task offloading focuses on efficiently distributing tasks among edge servers, where resources refer to limited computation, storage, and communication resources of edge and cloud servers. Typically, the offloading process involves two layers of heterogeneous decisions making problems (**P1, P2**) as follows,

- **P1. Devise-edge task offloading.** Enables devices to independently make decisions on offloading resource-intensive tasks to nearby edge servers, fostering efficient utilization of resources.
- **P2. Edge-edge task offloading.** Leverages edge-edge collaborations, where tasks initially received by a local edge server can be offloaded to neighboring servers with underutilized resources.

Problem Model: The main problem can be formulated as the decomposition of sub-problems **P1** and **P2** as a **Decenteralized Partially Observable Markov Decision Processes (Dec-POMDP)** [12], where multiple devices and edge servers interacting with each other by its observation of the environment, which is a part of main overall state.

Research Methodology:

1. **Algorithm Design:** Developing a **MARL** algorithm using techniques such as **Deep Deterministic Policy Gradient (DDPG)** [13] or **Dueling Double Deep Q-Networks (D3QN)** [14], with a focus on communication and collaboration, coordination or competition between agents.
2. **Simulation Environment:** A simulated MEC environment will be developed using Python or a suitable simulation platform, where mobile devices can offload tasks to edge servers and edge servers can distribute their computation workloads, under different network conditions.
3. **Key Challenges:** (a) Coordination or competition between agents. (b) The non-stationary environment due to actions of other agents. (c) Scalability issues as the number of agents increases.

Meta-Reinforcement Learning for Optimized Resource Management in Heterogeneous Mobile Edge Computing

Motivation: Meta DRL focuses on training agents that can quickly adapt to new tasks or environments with minimal additional learning [15]. It is designed for scenarios where agents face a wide variety of tasks, and the aim is to learn a policy that generalizes well across different tasks. The primary objective is to equip the agent with meta-knowledge, allowing it to efficiently adapt to new tasks by leveraging past learning experiences. In MEC, a meta-trained agent could adapt its offloading strategy efficiently when moving between different environments, quickly optimizing its offloading decisions in unfamiliar settings.

Problem Statement: Efficient task offloading is crucial to ensure seamless resource distribution in MEC. Typically, the overall Resource Management process involves three layers of heterogeneous Resource scheduling decisions (**P1**, **P2**, **P3**), each of which performs in a specific collaboration manner.

- **P1. Edge-cloud service placement** [16]. The cloud caches all services with sufficient storage space. Considering the storage limits of edge servers, only a subset of services can be placed in each edge server. Services can be migrated from a cloud to an edge or between edge servers, which requires efficient collaboration.
- **P2. Edge-edge computation offloading** [17]. The task offloading decision-making process focuses on efficiently distributing tasks among edge servers. Edge-edge collaborations enable edge servers to offload their computation workload to neighboring servers, ensuring better resource utilization.
- **P3. Intra-edge resource allocation** [18]. On edge servers, there may be several tasks competing for resources among offloaded tasks on the same server. Intra edge there is a resource competition among offloaded tasks on the same server. Intra-edge resource allocation aims to determine how resources should be allocated to each offloaded task.

Problem Model: To apply Meta RL for address combination of sub-problems **P1**, **P2**, and **P3**, each problem can be formulated as an individual MDP model. The MDP learning process should be decomposed into two parts: **learning a meta policy efficiently across all MDPs** and **learning a specific strategy for an MDP quickly based on the learned meta policy**.

Research Methodology:

1. **Algorithm Design:** Developing a Multi-Agent Meta-Reinforcement Learning algorithm using techniques such as **Meta-Actor and Meta-Critic Networks** [19], with a focus on global optimization in MEC.
2. **Simulation Environment:** A simulated MEC environment will be developed using Python or a suitable simulation platform, where cloud and edge servers be able to cache services and distribute tasks in whole resources, under different network conditions.
3. **Key Challenges:** (a) The meta-learned policy should work well across different, unseen tasks. (b) Balancing between exploration (learning new tasks) and exploitation (using learned knowledge).

Federated Deep Reinforcement Learning for Continuous Improving Intradependente Task Offloading in Mobile Edge Computing Network

Motivation: Federated Reinforcement Learning extends traditional DRL by allowing multiple agents to collaboratively learn a global policy without sharing their local data directly. Each agent makes decisions based on its local observations while cooperating with others to achieve shared system goals. In a Mobile Edge Computing (MEC) environment, edge devices can independently train DRL models using their local data and periodically send updates to a central server. The server aggregates these updates to build a global model, optimizing task offloading across the network. Specificly, federated DRL focuses on collaborative learning across decentralized devices while preserving data privacy.

- **Accelerate training process** for agents involved in MEC network.
- Enable mobile devices to **collectively contribute to enhancing the offloading model**.
- Support **continuous learning** as new mobile devices join the network.

Problem Statement: Efficient task offloading is crucial for optimizing resource utilization and minimizing latency. However, existing approaches often treat tasks as independent, overlooking the complexities introduced by intra-dependencies among tasks, leading to suboptimal resource utilization and performance. To address Dependency-Aware Task Offloading, there is a need for effectively model intra-dependent tasks,

- **Dependency-Aware Task Offloading**
  - Incorporate a **Task Call Graph Representation** to account for dependencies among tasks, improving task model accuracy and offloading effectiveness. Task call graphs can effectively represent dependencies, allowing for a clearer understanding of task relationships.

Problem Model: The problem can be formulated as MDPs, where multiple devices and edge servers interacting with each other by its observation of the environment and learn global palices to achieve shared system goals.

Research Methodology

1. **Algorithm Design:** Developing a **federated DRL** framework for optimizing interdependent task offloading in MEC networks, using teqnics such as **D3QN**, **DDPG**, or **Actor-Critic Network**, which enable network entities to collaboratively learn and adapt offloading strategies without compromising user privacy.

2. **Simulation Environment:** A simulated MEC environment will be developed using Python or a suitable simulation platform, where devices be able to dispatch their tasks to edge servers and provides collabration for all devices and edge servers, under different network conditions.

3. **Key Challenges:** (a) Communication efficiency to optimize data exchange between devices and servers. (b) Heterogeneity to accommodate diverse network components and devices.

References

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb 2015.

[2] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

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

[4] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Process Mag*, vol. 34, no. 6, pp. 26–38, Nov 2017.

[5] M. S. Munir, N. H. Tran, W. Saad, and C. S. Hong, "Multi-agent meta-reinforcement learning for self-powered and sustainable edge computing systems," *IEEE Trans. Netw. Serv. Manag.*, vol. 18, no. 3, pp. 3353–3374, 2021.

[6] Y. Gong, H. Yao, J. Wang, M. Li, and S. Guo, "Edge intelligence-driven joint offloading and resource allocation for future 6G Industrial Internet of Things," *accepted for publication in IEEE Trans. Netw. Sci. Eng.*, 2024.

[7] C. Sun, X. Li, C. Wang, Q. He, X. Wang, and V. C. Leung, "Hierarchical deep reinforcement learning for joint service caching and computation offloading in mobile edge-cloud computing," *accepted for publication in IEEE Trans. Services Computing*, 2024.

[8] P. Hernandez-Leal, M. Kaisers, T. Baarslag, and E. M. De Cote, "A survey of learning in multiagent environments: Dealing with non-stationarity," *arXiv preprint arXiv:1707.09183*, 2017.

[9] L. Liao, Y. Lai, F. Yang, and W. Zeng, "Online computation offloading with reinforcement learning algorithm in mobile edge computing," *J Parallel Distrib Comput*, vol. 171, pp. 28–39, Jan 2023.

[10] L. Huang, S. Bi, and Y.-J. A. Zhang, "Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks," *IEEE Trans. Mob. Comput.*, vol. 19, no. 11, pp. 2581–2593, Jul 2019.

[11] K. Zhang, Z. Yang, and T. Başar, "Multi-agent reinforcement learning: A selective overview of theories and algorithms," *Handbook of reinforcement learning and control*, pp. 321–384, 2021.

[12] F. A. Oliehoek, C. Amato *et al.*, *A concise introduction to decentralized POMDPs*. Springer, 2016, vol. 1.

[13] T. Lillicrap, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[14] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 30, no. 1, 2016.

[15] J. Beck, R. Vuorio, E. Z. Liu, Z. Xiong, L. Zintgraf, C. Finn, and S. Whiteson, "A survey of meta-reinforcement learning," *arXiv preprint arXiv:2301.08028*, 2023.

[16] V. Farhadi, F. Mehmeti, T. He, T. F. La Porta, H. Khamfroush, S. Wang, K. S. Chan, and K. Poularakis, "Service placement and request scheduling for data-intensive applications in edge clouds," *IEEE/ACM Transactions on Networking*, vol. 29, no. 2, pp. 779–792, 2021.

[17] R. Han, S. Wen, C. H. Liu, Y. Yuan, G. Wang, and L. Y. Chen, "Edgetuner: Fast scheduling algorithm tuning for dynamic edge-cloud workloads and resources," in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*. IEEE, 2022, pp. 880–889.

[18] X. Xiong, K. Zheng, L. Lei, and L. Hou, "Resource allocation based on deep reinforcement learning in iot edge computing," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 6, pp. 1133–1146, 2020.

[19] W. Ding, F. Luo, C. Gu, Z. Dai, and H. Lu, "A multiagent meta-based task offloading strategy for mobile-edge computing," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 16, no. 1, pp. 100–114, 2023.