Reinforcement Learning Improves Edge Computing

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 [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 [2].

Introduction

Mobile Edge Computing is emerging as a promising paradigm to enhance the computational capacity of mobile devices by offloading tasks to nearby edge servers. This paradigm aims to reduce latency, energy consumption, and improve Quality of Experience (QoE) for end-users. However, one of the major challenges in MEC is the efficient decision-making process for computation offloading, 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 systems.

Motivation

The rapid growth of mobile applications, such as augmented reality, real-time gaming, and high-definition video streaming, has placed significant demands on the computational resources of mobile devices. While the processing power of these devices has improved, there remain significant limitations in battery life, processing speed, and memory capacity. Mobile Edge Computing (MEC) has emerged as a solution, enabling computation offloading to nearby edge servers to alleviate the processing burden on mobile devices. However, the challenge lies in determining how and when to offload computational tasks efficiently, especially in a dynamic network environment with varying resources and user demands. The decision-making process becomes more complex with the presence of multiple devices competing for limited edge resources. Current solutions often employ heuristic or single-agent approaches, which are not robust in highly dynamic and multi-user MEC environments. Multi-Agent Deep Reinforcement Learning (DRL) offers a promising avenue for addressing this challenge. By allowing multiple agents (mobile devices and edge servers) to autonomously learn and adapt their offloading strategies, it becomes possible to optimize system-wide performance metrics such as energy consumption, latency, and resource utilization.

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 Reinforcement Learning (RL) or Deep Reinforcement Learning (DRL), 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. Multi-Agent Reinforcement Learning (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.

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 the MEC, the task offloading decision-making process 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 proceses involves two layers of heterogeneous decisions making prolems (**P1, P2**) as follow,

- **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, ensuring better resource utilization.

Problem Model: The main problem can be formoulated as decomposation of sub-problems **P1** and **P2** as a **Decenteralized partially observable markove decision procsses (Dec-POMDP)**, where multibe devices and edge servers interacting with each other by its own observation of environment, which is a part of main overall state.

Research Methodology:

1. **Algorithm Design:** Developing a Multi-Agent Deep Reinforcement Learning algorithm using techniques such as **Deep Deterministic Policy Gradiant (DDPG)** or **Dueling Deep Q-Networks (DDQN)**, with a focus on communication and collaboration, cordination or competitation 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 thier computation workloads, under different network conditions.
3. **Key Challenges:** (a) Coordination or competition between agents. (b) 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. 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 (e.g., from urban to rural networks), 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.** The cloud caches all services with sufficient storage spaces. Considering 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.** The task offloading decision-making process focuses on efficiently distributing tasks among edge servers. Edge-edge collaborations enables edge servers to offloade their computation workload to neighboring servers, ensuring better resource utilization.
- **P3. Intra-edge resource allocation.** 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 problems can be formoulated as an individual MDP models. The MDP learning process shoud 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 **Multi-Agent Meta-Actor and Meta-Critic Networks**, 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 cload and edge servers be able to cache services and distribut 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 Multi-Agent 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.

Problem Statement:

- **Dependency-Aware Task Partitioning**
  - Incorporate a **Task Call Graph Representation** to account for dependencies among tasks, improving task model accuracy and partitioning effectiveness.
- **Federated Deep Reinforcement Learning**
  - Leverage federated learning for mobile devices in training process.
  - Enable mobile devices to collectively contribute to enhancing the offloading model.
  - Support continuous learning as new mobile devices join the network.

Problem Model:

Research Methodology

1. **Algorithm Design:** Developing a federated deep reinforcement learning framework for optimizing interdependent task offloading in MEC networks, using teqnics such as DQN or DDPG, which enable network entities to collaboratively learn and adapt offloading strategies without compromising user privacy.
2. **Simulation Environment:**
3. **Key Challenges:** (a) Communication efficiency, (b) Heterogeneity

1 Summary of Differences

- **Multi-Agent DRL** focuses on interactions between multiple agents, each learning in the presence of others.
- **Federated DRL** focuses on collaborative learning across decentralized devices while preserving data privacy.
- **Meta DRL** focuses on quick adaptation to new tasks by using past learning experiences.

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, 2015.

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