DELCAS: Deep Reinforcement Learning based GPU CaaS Packet Scheduling for Stabilizing QoE in 5G Multi-Access Edge Computing*

Changha Lee, Kyungchae Lee, Gyusang Cho, and Chan-Hyun Youn

School of Electrical Engineering, KAIST, Daejeon, Republic of Korea
{changha.lee, kyungchae.lee, gyusang.cho, chyoun}@kaist.ac.kr

Abstract. Recently, Docker Container as a Service (CaaS) has been provided for multi-user services in the 5G Multi-Access Edge Computing (MEC) environment, and servers that support accelerators such as GPUs, not conventional CPU servers, are being considered. In addition, as the number of AI services is increasing and the computation power required by deep neural network model increases, offloading to edge servers is required due to insufficient computational capacity and heat problem of user devices (UE). However, there is a resource scheduling problem because all users' packets cannot be offloaded to the edge server due to resource limitations. To address this problem, we suggest deep reinforcement learning-based GPU CaaS Packet scheduling named as *Delcas* for stabilizing quality of AI experience. First, we design the architecture using containerized target AI application on MEC GPUs and multiple users send video stream to MEC server. We evaluate video stream to identify the dynamic amount of resource requirement among each users using optical flow and adjust user task queue. To satisfy equal latency quality of experience, we apply lower quality first serve approach and respond hand pose estimation results to each user. Finally, we evaluate our approach and compare to conventional scheduling method in the aspect of both accuracy and latency quality.

Keywords: 5G MEC · Task Offloading · Deep Reinforcement Learning

1 Introduction

The rise of 5G networks has unlocked new opportunities for communication and computing, enabling ultra-low latency, high bandwidth, and massive connectivity. This has paved the way for advancements in various domains, including edge computing, virtualization, the Internet of Things (IoT), and immersive user experiences. In particular, the integration of 5G with multi-access edge computing (MEC) and virtualization has emerged as a promising paradigm for efficient and scalable computing at the network edge, with the potential to revolutionize applications that require real-time interaction between humans and artificial intelligence (AI).

* Supported by Samsung Network (SNIC) and BK21

One such application that has gained significant attention is hand pose estimation, which involves accurately tracking and predicting the positions and orientations of human hand joints in real-time. Hand pose estimation has numerous practical applications, including virtual and augmented reality, gaming, human-computer interaction, sign language recognition, and robotics, among others [1]. The ability to accurately estimate hand poses in real-time is critical for enabling natural and intuitive interactions between humans and AI systems in immersive virtual environments, where users can interact with virtual objects and manipulate them using hand gestures.

To achieve real-time and accurate hand pose estimation in immersive user experiences, the integration of 5G, MEC, virtualization (with a focus on containerization), and COTS hardware, including Graphics Processing Units (GPUs), has become a topic of significant interest [2]. Containerization, specifically container as a service (CaaS), is an emerging virtualization technology that allows for lightweight and portable deployment of software applications, encapsulating them in self-contained containers that can be easily deployed, scaled, and managed across different computing environments [3].

In this paper, we aim to investigate of the multi-user offloading packet intergpu scheduling for our target AI application to satisfy service level objective in 5G multi-access edge computing. Inspired by DRL-based previous work, we analyze multi-user AI application offloading task and suggest a deep reinforcement learning based scheduling algorithm for stable quality of experience in terms of accuracy and latency. We evaluate our proposed method in experiment section and concludes the paper.

2 Related Work

Optical Flow Computation. To identify the amount of offloading resources in hand pose estimation, we need to extract movement information in the first person view because if the hand moves quickly, more resources must be served to process more frames to increase accuracy. Therefore, we are going to refer to optical flow computation. Optical flow estimation has been extensively studied in computer vision literature, with numerous techniques proposed over the years. Traditional techniques [4, 5], are based on classical optimization or differential techniques and are widely used due to their simplicity and low computational requirements.

Resource Controller based on Deep Reinforcement Learning. Deep reinforcement learning has shown promising results in resource scheduling and resource allocation problems, where the goal is to optimize the allocation of limited resources, such as CPU, GPU, or memory, to different tasks or processes to achieve a specific objective, such as maximizing performance, minimizing energy consumption, or reducing computation time. The previous research [6, 7], which combined Q-learning with deep neural networks and genetic algorithms to learn optimal control policies in an end-to-end manner. However, applying deep reinforcement learning to utilize MEC services for large-scale users is very difficult

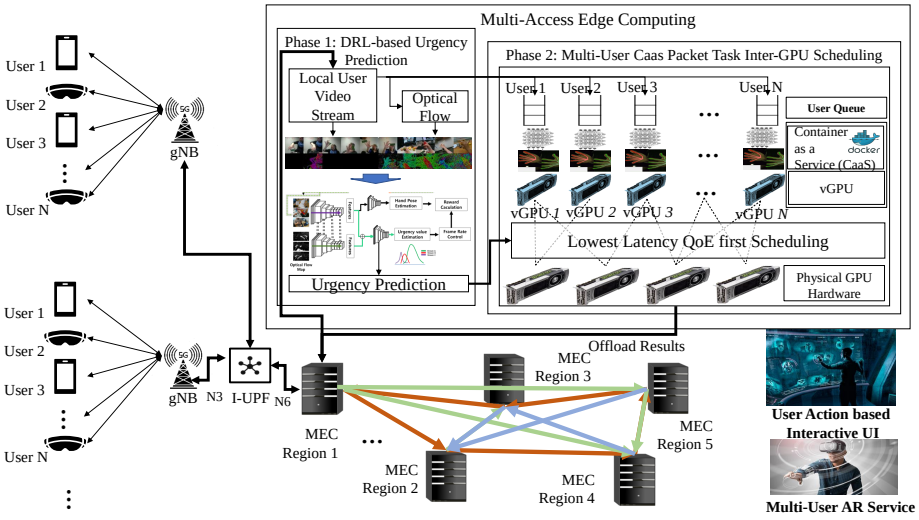

Fig. 1. The overview of the proposed architecture. Phase 1 predicts the urgency score, which depicts the relative amount of computing resource requirements among multiple user input streams and Phase 2 shows multi-user CaaS Packet Inter-GPU scheduling

due to resource limitations and trade-off between accuracy and latency in real-world case. To address these challenges, we carefully design the AI application system using MEC and introduce how to provide stable QoE with accuracy and latency in the next section.

3 Methodology

3.1 System Overview

Fig. 1 shows an proposed architecture that can estimate the computing resources required for multi-user hand posture estimation. For each user equipment UE_i send video stream to MEC server through 5G base station (gNB) and user plane function (UPF). Based on this, we propose scheduling to provide stable quality of experience for multiple requests of deep learning model computation in a containerized GPU Pool in a 5G MEC environment. First, we need to train deep reinforcement learning-based network to predict the urgency score, which depicts the relative amount of computing resource requirements among multiple user input streams to stabilize accuracy quality in *Phase 1*. After training urgency prediction network in phase-1, we use DRL-based model inference results to adjust served resource ratio according to multiple user in *Phase 2*. We will explain details in the next phase sections.

3.2 Problem Setting

In this paper, some of the environmental conditions for target applications (hand posture estimation) of extended reality (XR) through 5G MEC offloading service are assumed as follows:

1. The amount of computational resources required to provide the same QoE varies according to the dynamically changing movement of multiple users.
2. 5G MEC GPU servers are computationally more powerful than XR devices, but computing resources may be limited.
3. The 5G MEC GPU server supports container-based virtualization and can provide CaaS to users by virtualizing GPU resources.

In the assumption above, multiple user equipment $UE = \{ue_1, ..., ue_N\}$ send video stream $X = \{x_1, ..., x_N\}$ to edge server and call for DL model-based estimation of their hand pose in the first person view. This paper proposes adaptive GPU resource scheduling, focusing on offloading scenarios of 5G MEC in the task of estimating hand posture.

3.3 Phase 1: DRL-based Multi-User Input Stream Control

In this section, we first introduce the deep reinforcement learning based Multi-Agent Resource Coordination (MARCO) agent proposed in our previous works[8, 9] to dynamically allocate computational resources to multiple users based on their urgency scores, which results in heterogeneous user inference rates. This phase aims to control the user inference rates where users with relatively high urgency(e.g. fast movements in object detection or hand-pose estimation) gets higher inference rate capacity given the limited amount of the edge-server resources, thus improving the user experience. We enable options such as optical flow-based motion information, entropy maximization, golden baseline, and static-scheduling baseline in the MARCO agent's reward function to improve its training stability and performance. Here, the golden baseline refers to the accuracy achieved when the server can handle all frame requests under the ideal assumption of an infinite resource situation. We defined a reward function that minimizes the decrease in accuracy obtained during the finite resource scheduling in our experiment, as shown in the Eqa.(1) and (2). Additionally, we also used the relative reward advantage compared to the static-scheduling case as our objective to further ensure better convergence. Details of the reward formulation can be found in [8].

$$R_{golden} = \frac{1}{m} \frac{\sum_{i=1}^{m} (Y_i - \hat{Y_i^*})^2 - (Y_i - \hat{Y_i})^2}{\sum_{i=1}^{m} (Y_i - \hat{Y_i^*})^2} \tag{1}$$

$$\begin{aligned} R_{final} &= R_{golden} - R_{static} \\ &= \frac{1}{m} \frac{\sum_{i=1}^{m} (Y_i - \hat{Y_i^s})^2 - (Y_i - \hat{Y_i})^2}{\sum_{i=1}^{m} (Y_i - \hat{Y_i^*})^2} \end{aligned} \tag{2}$$

3.4 Phase 2: Multi-User CaaS Packet Scheduling

In this section, we introduce containerized AI application task scheduling for equal quality of latency. Following assumption in this paper, we have limited GPU resources in MEC server. Therefore, we need to apply GPU virtualization because GPU virtualization allows multiple users or containers to share the same physical GPU, effectively utilizing the GPU resources.

GPU Containerization. To virtualize GPU resources using Docker containers [10], a technology called *NVIDIA Docker* can be used. NVIDIA Docker is an extension of the Docker containerization platform that provides GPU virtualization capabilities. It allows Docker containers to access and utilize GPUs on the host system, enabling GPU-accelerated computing within containers. The NVIDIA Docker runtime includes the necessary GPU drivers, libraries, and tools required for GPU-accelerated computing. Docker containers with GPU support can be deployed to a host system with NVIDIA Docker runtime installed. The containers can specify the required GPU resources, such as the number of GPUs, GPU memory, and GPU device IDs, using Docker command line options. In this paper, for full GPU utilization, main scheduler container possesses all GPU and has the permission to allocate GPU to each users. Each user AI application container has its owned queue denoted by $vGPUReq_{ue_i}$.

Hand Pose Estimation CaaS. For hand pose estimation container as a service, we refer to a lightweight hand pose estimation model based on deep neural network [11], denoted by f_{hpe}. Let the hand pose result is Y_i, then the result is presented by the point set $\{k^1, ..., k^m\}$ where $k \in \mathbb{R}^3$ representing three dimensional coordinates and m means the number of hand joint. Due to end-to-end running in DL application, we can obtain the result from input stream $Y_i = f_{hpe}(X_i) = \{k^1, ..., k^m\}$ for each user ue_i. This hand pose estimation application is deployed on the nvidia-docker container.

The best way to improve accuracy is always calculating the hand joint position, but due to the prior assumption that the resource is insufficient, it is impossible to measure the hand joint position in all users' video streams. To allocate resource efficiently, we utilize the urgency obtained by the DRL network. From the urgency μ_i, we compute the urgency ratio set U as follows:

$$\begin{aligned} U &= \{U_1, ..., U_i, ..., U_N\} \\ &= \{\frac{e^{\mu_1}}{\sum_{i=1}^{N} e^{\mu_i}}, ..., \frac{e^{\mu_i}}{\sum_{i=1}^{N} e^{\mu_i}}, ..., \frac{e^{\mu_N}}{\sum_{i=1}^{N} e^{\mu_i}}\} \end{aligned} \tag{3}$$

where the sum of urgency ratio element is $\sum_i U_i = 1$ and $U_i \leq 1$.

As the hand pose estimation service can not be provided among all video stream $X = \{X_1, ..., X_i\}$, the efficient input stream $\hat{X}_i^*$ in which the hand joint is to be predicted is adjusted according to the ratio as follows:

$$\hat{X}_i^* = U_i \times X_i \tag{4}$$

where $\hat{X}_i^* \leq X_i$. This efficient input stream will be pushed into virtualized request queue $vGPUReq_{ue_i}$. Then, the length of the user queue is given by $|vGPUReq_{ue_i}| = \hat{X}_i^*$.

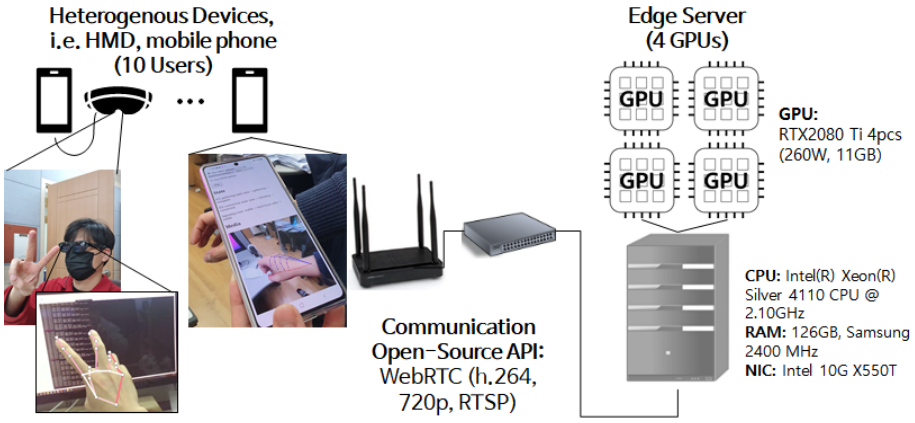

Fig. 2. The overview of the lab-scale experimental environment

Lowest Latency QoE First Scheduling. To provide the equal latency-QoE in the hand pose estimation application for multiple user, the user packet having larger queue length should be served first than the others. We define the Latency QoE, l^i_{qoe} as follows:

$$l^i_{qoe} = \alpha_i / |vGPUReq_{ue_i}| \tag{5}$$

where α_i presents a priority coefficient for user i. Equation (5) means that when a user i has larger queue length than the other users, the latency QoE will be the lowest and will be served first in our proposed architecture.

4 Experiment and Evaluation

In order to evaluate the performance of our proposed architecture, we implemented hand pose estimation model [11] on the nvidia-docker with Pytorch framework [12]. For real-time communication and supporting heterogeneous user device, we apply Web Real-Time Communication (Web-RTC), which is an open-source technology that enables real-time communication between browsers and mobile applications without the need for external plugins or software. By using WebRTC, video stream protocol follows Real Time Streaming Protocol (RTSP) in the format of H.264 and 720p. Fig. 2 shows our lab experimental environment.

Our proposed approach is compared and evaluated against a static scheduling method satisfying latency and a golden case method maximizing accuracy, in the aspect of both accuracy and latency. Our target service level objective in latency is 20 ms and accuracy is to provide as high as possible by the refinement of the hand-pose observation model based on deep reinforcement learning.

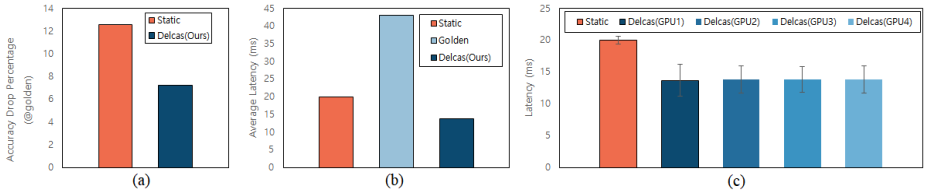

Fig. 3. Performance Result. **(a)** presents the average accuracy drop from the golden case. **(b)** shows the average latency among 10 users. **(c)** represents the latency and standard deviation according to each GPUs or method

4.1 Evaluation Metric

We measure the accuracy according to the curve area under the percentage correct keypoints as known as AUC of PCK with the range of $20mm$ to $50mm$ by following [11]. For the latency measurement, we measure the elapsed time at edge server side because it is hard to exact time in client devices in video streaming via WebRTC. Although the frame rate may be slightly lower than that of the edge server on the client device, we measured the processing time per frame at the server stage because we confirmed that it was almost the same.

4.2 Performance Evaluation and Discussion

Fig. 3 shows performance results on our target application. In Fig. 3(a), our proposed method shows average latency 13.75 ms, of which standard deviation is 2.19. In static scheduling, we set the target latency as 20 ms and we confirmed that the target latency is satisfied according to the average latency results, however, the average accuracy drop is larger than ours in Fig. 3(b). The golden case shows the best accuracy but it shows 43.1 ms which is not satisfying our service level objective. In Fig. 3(c), it shows that the average latency among physical 4 GPUs is stabilized via our proposed method which satisfies the latency under the 20ms.

5 Conclusion

In conclusion, we propose a novel approach for deep reinforcement learning-based GPU Container as a Service (CaaS) packet scheduling to enhance the quality of AI experience. With the increasing demand for AI services and the limitations of user devices in terms of computational capacity and heat dissipation, offloading to edge servers with accelerators such as GPUs has become crucial. However, due to resource constraints, not all users' packets can be offloaded to the edge server. To tackle this issue, we utilize optical flow-based evaluation of video stream to dynamically adjust the resource allocation for each user's task queue, while maintaining stable latency quality of experience through a lower quality first serve approach. Our approach is compared to conventional

scheduling methods in terms of accuracy and latency quality, and the results demonstrate its effectiveness in stabilizing the quality of AI experience in the 5G MEC environment.

Acknowledgements This work is supported in part by Samsung Electronics Co., Ltd and in part by BK21.

References

1. Chakraborty, B.K., Sarma, D., Bhuyan, M.K., MacDorman, K.F.: Review of constraints on vision-based gesture recognition for human–computer interaction. IET Computer Vision **12**(1), 3–15 (2018)
2. Raj, P., Saini, K., Surianarayanan, C.: Edge/Fog Computing Paradigm: The Concept, Platforms and Applications. Academic Press (2022)
3. Goniwada, S.R., Goniwada, S.R.: Containerization and virtualization. Cloud Native Architecture and Design: A Handbook for Modern Day Architecture and Design with Enterprise-Grade Examples pp. 573–617 (2022)
4. Horn, B.K., Schunck, B.G.: Determining optical flow. Artificial intelligence **17**(1-3), 185–203 (1981)
5. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: IJCAI'81: 7th international joint conference on Artificial intelligence. vol. 2, pp. 674–679 (1981)
6. Xue, F., Hai, Q., Dong, T., Cui, Z., Gong, Y.: A deep reinforcement learning based hybrid algorithm for efficient resource scheduling in edge computing environment. Information Sciences **608**, 362–374 (2022)
7. Yang, T., Hu, Y., Gursoy, M.C., Schmeink, A., Mathar, R.: Deep reinforcement learning based resource allocation in low latency edge computing networks. In: 2018 15th international symposium on wireless communication systems (ISWCS). pp. 1–5. IEEE (2018)
8. Lee, K., Youn, C.H.: Reinforcement learning based adaptive resource allocation scheme for multi-user augmented reality service. In: 2022 13th International Conference on Information and Communication Technology Convergence (ICTC). pp. 1989–1994 (2022). https://doi.org/10.1109/ICTC55196.2022.9952934
9. Lee, K., Youn, C.H.: Reindear: Reinforcement learning agent for dynamic system control in edge-assisted augmented reality service. In: 2020 International Conference on Information and Communication Technology Convergence (ICTC). pp. 949–954. IEEE (2020)
10. Merkel, D., et al.: Docker: lightweight linux containers for consistent development and deployment. Linux j **239**(2), 2 (2014)
11. Zhou, Y., Habermann, M., Xu, W., Habibie, I., Theobalt, C., Xu, F.: Monocular real-time hand shape and motion capture using multi-modal data. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5346–5355 (2020)
12. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems **32** (2019)