Delivering Augmented Reality to the Edge: An Approach Toward Object Recognition through the In-Network Computing Paradigm

Washington R. D. Silva*
washingtonrds@estudante.ufscar.br
Department of Computer Science, Federal University of São Carlos (UFSCar)
Sorocaba, São Paulo, Brazil

Fábio L. Verdi
verdi@ufscar.br
Department of Computer Science, Federal University of São Carlos (UFSCar)
Sorocaba, São Paulo, Brazil

Guilherme M. V. Matos
guilherme.matos@estudante.ufscar.br
Department of Computer Science, Federal University of São Carlos (UFSCar)
Sorocaba, São Paulo, Brazil

Andrew Williams
andrew.williams@ericsson.com
Ericsson Research
Stockholm, Sweden

Abstract

Recent advancements in 5G technology have enabled significant breakthroughs in applications requiring high data throughput and low latency, such as Mobile Augmented Reality (MAR). Nevertheless, mobile devices tend to be computationally and energy-constrained, limiting their capability to provide synchronous interactions between the real environment and visual augmentations. Thus, in this work, we propose a collaborative inference method to run compute-intensive AR tasks, such as object recognition, on heterogeneous devices at the network edge. This approach would benefit both end-users and service providers by reducing end-to-end latency and operating costs, respectively.

ACM Reference Format:
Washington R. D. Silva, Fábio L. Verdi, Guilherme M. V. Matos, and Andrew Williams. 2024. Delivering Augmented Reality to the Edge: An Approach Toward Object Recognition through the In-Network Computing Paradigm. In *Proceedings of the CoNEXT Student Workshop 2024 (CoNEXT-SW '24), December 9–12, 2024, Los Angeles, CA, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3694812.3699933

1 Introduction

Each generation of mobile network technology has enhanced connectivity and transformed communication, enabling voice services, multimedia streaming, and internet browsing. The 5th and Beyond generations of mobile wireless networks (5GB) promise to be disruptive technologies, enabling applications previously hindered by high data throughput and strict low-latency requirements, such as Mobile Augmented Reality (MAR). [2].

For optimal Quality of Experience (QoE), MAR applications must provide accurate and synchronous interactions between the environment, the user, and visual augmentations within the human visual reaction time (20 milliseconds) [2, 4]. However, due to computing and energy constraints, mobile devices may not deliver the expected QoE. A potential solution is to distribute the computation: tasks requiring fewer resources are executed locally, while high-demand tasks are offloaded to a remote server [2].

Hence, the Multi-access Edge Computing (MEC) network architecture plays a significant role in delivering AR's promises by moving the processing of latency-critical applications closer to end devices and minimizing communication delays compared to Mobile Cloud Computing (MCC). However, MEC provides fewer computing resources than the cloud, creating a tradeoff between communication latency and computation capability [2].

In this work, we aim to address this tradeoff by investigating how we can distribute the computation of AR tasks, such as object recognition (which comprises both detection and classification of objects), across the heterogeneous devices available at the Edge. This includes not only domain-specific accelerators (e.g., Graphics Processing Units (GPUs)), but also programmable networking devices like Data Processing Units (DPUs), Field-Programmable Gate Arrays (FPGAs), and ASIC switches [5].

By doing so, we would be able to understand how collaborative inference across heterogeneous devices can benefit both applications and service providers, while contributing to an emerging research trend: the in-network machine learning [5].

2 Problem Statement

Although many Machine Learning (ML) applications have been proven to be feasible in the in-network computing paradigm, most existing works focus on networking applications, especially network anomaly detection. However, employing this strategy beyond networking is worth exploring, particularly for data-intensive and ultra-low latency applications [5]. In this sense, AR is a prime candidate for benefiting from in-network ML.

Nonetheless, object recognition in AR applications relies on large-scale ML models, which are challenging to map to programmable

*Both authors contributed equally to this research.

CoNEXT-SW '24, December 9–12, 2024, Los Angeles, CA, USA
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-1255-5/24/12
https://doi.org/10.1145/3694812.3699933

networking devices due to their computing constraints. A possible solution is to decompose the model into multiple components and distribute them across several target devices. In this context, inference is performed by combining the intermediate results generated by each device [5].

Fig. 1 illustrates the proposed solution concept, which is distinguished by its focus on heterogeneous devices, each with specific requirements. This approach introduces unique challenges in ensuring compatibility and efficient communication, crucial for achieving an effective collaborative inference, such as:

1 Identifying the role of each device in the inference computation;
2 Distributing different blocks of a Deep Learning model across different hardware;
3 Defining an optimal model partitioning criteria;
4 Orchestrating the distributed computing among heterogeneous devices without increasing the application latency beyond the required limits;

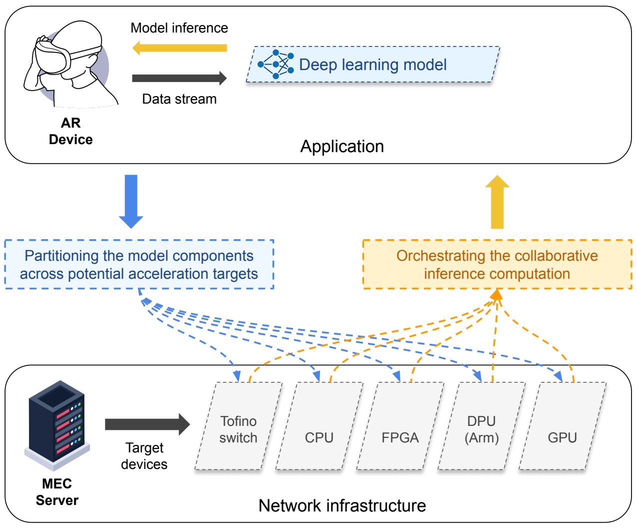

Figure 1: Collaborative inference conceptualization.

We will evaluate the collaborative inference effectiveness by assessing the object recognition accuracy and latency, the application frame rate, and the system energy consumption. Then, we will verify how the collaborative inference energy efficiency compares with traditional offloading strategies (e.g., GPUs and FPGAs). For such, we will use an edge server equipped with 2 Intel Xeon 6430 CPUs, an Nvidia L40s GPU, an Nvidia BlueField 3 DPU, and an Alveo U55C FPGA; all connected via a Tofino 2 switch with 400 Gb/s Ethernet interfaces.

3 Current research state

To address the previously identified challenges, we must first convert a trained model into an Intermediate Representation (IR). For such, we used ONNX [3], an open-source framework that represents Deep Neural Networks (DNNs) as Directed Acyclic Graphs (DAGs). By iterating over these graph nodes, we can estimate their computing complexity (in floating-point operations), energy cost, and computing time on each potential target device (e.g., CPU, DPU, GPU, FPGA). This enables us to derive partitioning criteria, split the DNN into subgraphs, and execute them on heterogeneous hardware with the ONNX runtime [1].

Distributing subgraphs across target devices using a single strategy may not be efficient. As observed in Table 1, while the energy cost of performing convolutions on the DPU is lower than the CPU, the computing time is higher, failing to meet the AR application's required latency. Although the GPU seems optimal for offloading, the FPGA offers comparable energy cost and computation time.

Table 1: ResNet18 DAG profiling example.

Energy cost				
Node	CPU	GPU	FPGA	DPU (ARM)
/layer1.0/conv1	0.0254 W	0.0004 W	0.0009 W	0.0092 W
/layer2.0/conv1	0.0127 W	0.0002 W	0.0005 W	0.0046 W
/layer3.0/conv1	0.0127 W	0.0002 W	0.0005 W	0.0046 W
/layer4.0/conv1	0.0127 W	0.0002 W	0.0005 W	0.0046 W
Computing time				
/layer1.0/conv1	0.0940 ms	0.0013 ms	0.0082 ms	0.5734 ms
/layer2.0/conv1	0.0470 ms	0.0006 ms	0.0041 ms	0.2867 ms
/layer3.0/conv1	0.0470 ms	0.0006 ms	0.0041 ms	0.2867 ms
/layer4.0/conv1	0.0470 ms	0.0006 ms	0.0041 ms	0.2867 ms

Comparing these devices directly may be unfair, given the GPU's superior computing capacity. However, our goal is not to prove that programmable networking devices can surpass GPUs in performance or efficiency but to demonstrate how they can execute complex computations while meeting the AR application's requirements. Since these devices are already expected to be part of the networking infrastructure, utilizing them would save further expenses with new hardware and energy.

Thus, throughout the PhD course, we will investigate how to use reinforcement learning to determine the role of each target device in collaborative inference and partition the model accordingly.

Acknowledgments

This work was supported by Ericsson Telecomunicações Ltda., and by the Sao Paulo Research Foundation (FAPESP), grant 2021/00199-8, CPE SMARTNESS. This study was partially funded by CAPES, Brazil - Finance Code 001.

References

[1] Tobias Alonso et al. 2021. Elastic-df: Scaling performance of dnn inference in fpga clouds through automatic partitioning. *ACM Trans. Reconfigurable Technol. Syst.* 15, 2 (2021), 1–34. doi:10.1145/3470567.
[2] Yuyi Mao et al. 2017. A Survey on Mobile Edge Computing: The Communication Perspective. *IEEE Commun. Surv. Tutor.* 19, 4 (2017), 2322–2358. doi:10.1109/COMST.2017.2745201.
[3] ONNX. 2024. Open Neural Network Exchange (ONNX). https://github.com/onnx/onnx. Version: 1.16.2.
[4] Jan-Philipp Stauffert, Florian Niebling, and Marc Erich Latoschik. 2020. Latency and Cybersickness: Impact, Causes, and Measures. A Review. *Front. Virtual Real.* 1, 582204 (2020), 1–10. doi:10.3389/frvir.2020.582204.
[5] Changgang Zheng et al. 2024. In-network machine learning using programmable network devices: A survey. *IEEE Commun. Surv. Tutor.* 26, 2 (2024), 1171–1200. doi:10.1109/COMST.2023.3344351.