An Edge-Centric Scalable Intelligent Framework To Collaboratively Execute DNN

Jiashen Cao [*1] **Fei Wu** [*1] **Ramyad Hadidi** [1] **Lixing Liu** [1] **Tushar Krishna** [1] **Micheal S. Ryoo** [2] **Hyesoon Kim** [1]

Abstract

Deep neural networks (DNNs) help us to address several new problems. However, they are compute intensive and resource hungry. To deploy a DNN-based service, consumers need to either rely on cloud-based services or acquire high-performance and expensive systems. These approaches create a dependability on high-end infrastructures, computing systems, and availability of network. Moreover, easy-to-setup cloud services increase security risks and raise new privacy concerns. To overcome these challenges, which may limit the widespread usage of DNN-based applications, we are proposing an edge-centric scalable intelligent framework that enables the execution of DNNs. The computing nodes in this framework are several edge and Internet of Thing (IoT) devices that have already been installed in a place, such as cameras, smart thermostats, and smart locks. To address the inadequate power and computing resources of each of the edge devices, our framework, by utilizing concepts presented in previous studies [1, 2, 3, 4], realizes collaboration between these devices. This framework utilizes Docker virtualization service and Kubernetes cluster management service to provide real-time scalability for handling pressured or idle devices. In addition, the virtualization allows easy installation, minimum setup time, and flexibility in DNN models. To demo, we illustrate our work using up to 10 Raspberry Pis by showcasing DNN models execution, such as AlexNet and YOLO, in real time. Demo website: http://comparch.gatech.edu/hparch/sysml.

[*]Equal contribution

1 Introduction & Motivation

The widespread adaptation of deep neural network (DNN) to solve new problems enables us to tackle several ordinary tasks. Since DNN execution is inherently compute intensive and resource hungry, it requires a high-performance computing system or a cloud-based provider. These approaches lead to an increased risk of losing privacy as well as a strong dependency on the network connectivity. Moreover, for an ordinary user, investing in a high-performance and local system is not affordable. On the other hand, the ever-increasing number of Internet of Things (IoT) and edge devices[2] being integrated in our daily lives creates a great platform for DNN execution. In fact, the tasks of several of these IoT devices are monitoring, recording, and acting on gathered data in our environments which can be adapted by DNNs. However, the demanded computational power from resource-hungry DNN-based applications and manufacturing costs of IoT devices, limit the integration of DNNs in these devices. Furthermore, the computational demands are escalated because IoT devices must process raw data in real-time with time-sensitive constraints. Since ensuring many of these characteristics in processing complex raw data is computationally expensive, DNN-based applications are not performed on these devices.

Our previous studies [1, 2, 3, 4] enable collaborative execution of DNNs on resource-constrained devices such as edge and IoT devices. In summary, by exploiting various model- and data-parallelism techniques for convolution and fully-connected layers, we are able to speed up real-time performance of single-batch inferences for several visual models, such as AlexNet, VGG, Xception, C3D, and Yolo. Our studies propose to distribute the entire DNN model onto multiple resource-constraint devices so that we can reduce computation pressure and memory usage. Each device handles a certain task or, in other words, is in charge of a portion/entire layer of a DNN models. It is critical for us to find the most optimal model distribution among the large volume of possible sub-optimal choices. Currently, to distribute the tasks, our studies either use offline profiling [3, 2], or heuristics [1]. However, both of these methods are time-consuming procedures and must be performed manually or semi-manually before the deployment.

To improve the scalability and flexibility of our framework, in this demo, we integrate our framework with Docker virtualization service and Kubernetes cluster management ser-

[1]Georgia Institute of Technology, Atlanta, GA, USA [2]Google Brain, Mountain View, CA, USA. Correspondence to: Ramyad Hadidi <rhadidi@gatech.edu>.

[2]In this paper, we use IoT and edge devices interchangeably

vice. Our previous framework was not able to achieve real-time scalability because the framework could not adjust the service distribution in real time in an event when a certain service (i.e., task) becomes a bottleneck due to various reasons (i.e., network congestion, device might not be idle anymore). By adding Docker virtualization and Kubernetes management tool to our framework, we are able to scale up and scale down services in real time. We are also able to monitor the computation and memory usage of cluster nodes and manage the cluster effortlessly. In addition, with the virtualization features of Docker, we have the opportunity to extend our work to the heterogeneous systems (e.g., a mix of Raspberry Pis and Nvidia Jetson TX2) with adaption to different distribution setup.

Another issue we faced in our previous studies and framework is finding an optimal distribution for a given DNN model, which required us to fine-tune each distribution. To profile each unique distribution, we needed to put more than 30 minutes from code setup to execution. Moreover, since finding an optimal distribution is an NP-hard problem, it was nearly impossible to find the optimal distribution solution manually and with limited monitoring tools. Having a widely used framework (i.e., Kubernetes) that is able to efficiently monitor all performance metrics and test various distribution automatically creates the base infrastructure for extending our study, so that it can learn the optimal distribution. In the future, our system should be able to learn the optimal distribution by itself and use such distribution to optimize the real-time inference.

2 Demo Details & Audience Interactions

Although several features of our new framework are under active development, such as learning the most optimal distribution automatically or extending our work to heterogeneous systems, model distribution techniques and the collaborative execution of DNN models have been implemented in our new framework. Docker virtualization service and Kubernetes management tool also have been successfully integrated into our framework, which adds scalability and monitoring features during execution. We believe showcasing our work in SysML 2019 will enable us to share our finding with the community and receive constructive feedbacks and ideas.

In details, in our demo session, we will use up to 10 Raspberry Pi 3s that utilize our framework based on Docker and Kubernetes. Raspberry Pis will be connected with a wireless router/switch. Audience would be able to see performance metrics through Kubernetes dashboard and observe real-time performance of several devices. We will describe how each DNN model is distributed and the techniques we have used to ensure correctness of the execution. Moreover, we will present real-time image inference along with our model- and data-parallelism techniques. For models, we will execute one object detection model such as YOLO, SSD, or faster R-CNN, and one image recognition model such as AlexNet, VGG, or Xception.

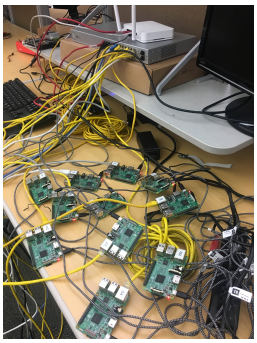

Figure 1. Prototype system: A Router, a switch, and 11 RPi3s.

Here is the summary of our demo:

- Deploy DNN models on up to 10 Raspberry Pi 3s.
- Use real-world models such as YOLO and VGG.
- Illustrate using real-time inputs for image recognition, object detection, and action recognition models.
- Describe and present data- and model-parallelism techniques in our models from our previous studies.
- Describe and illustrate Docker and Kubernetes procedures in our system.
- Demo Website: http://comparch.gatech.edu/hparch/sysml

References

[1] Hadidi, R., Cao, J., Ryoo, M., and Kim, H. Collaborative execution of deep neural networks on internet of things device. *arXiv preprint arXiv:1901.02537*, 2018.

[2] Hadidi, R., Cao, J., Ryoo, M. S., and Kim, H. Distributed perception by collaborative robots. *IEEE Robotics and Automation Letters (RA-L), Invited to IEEE/RSJ IROS'18, Madrid, Spain*, 3(4):3709–3716, Oct 2018.

[3] Hadidi, R., Cao, J., Woodward, M., Ryoo, M., and Kim, H. Musical chair: Efficient real-time recognition using collaborative iot devices. *arXiv preprint arXiv:1802.02138*, 2018.

[4] Hadidi, R., Cao, J., Woodward, M., Ryoo, M. S., and Kim, H. Real-time image recognition using collaborative iot devices. In *Proceedings of the 1st on Reproducible Quality-Efficient Systems Tournament on Co-designing Pareto-efficient Deep Learning*, ReQuEST '18. ACM, 2018.