Towards Energy-Efficient Neural Network Training on the Cloud for Effective Inference on IoT/Edge Devices

Yasutaka Wada[1,a)] Ken’ya Onai[1] Musashi Aoto[1]

Abstract: IoT/Edge devices need to be low-power, and it is required to enhance their computational power by employing hardware accelerators like FPGAs and by offloading heavy workloads to the cloud side. However, maintaining the cloud environments at a low power is challenging because of their unstable workloads with virtualization. This paper explains our idea and strategy to realize energy-efficient deep learning computation on virtualized cloud platforms and IoT/edge devices. We propose to utilize cloud servers to provide sufficient computational resources for neural network training and its model optimizations. Then, IoT/edge devices can focus on inference tasks while accelerating the tasks with FPGAs. Based on this strategy, we are developing a framework to minimize the power consumption of virtualized cloud servers considering the difference in computational workloads between deep learning training tasks and High-Level Synthesis tasks.

Keywords: Deep Learning, Cloud, Virtualization, Energy Efficiency, Edge, IoT

1. Introduction

IoT/Edge devices are now widely used, and there are significant demands upon them as they perform essential roles in establishing a smart and sustainable society. However, sufficient computational capabilities of these devices need to be prepared to fulfill such requirements. At the same time, these devices are required to be low-power. To solve such a conflict, typically a certain type of accelerator hardware, like GPUs and FPGAs, are utilized to obtain both low-power and high performance. The FPGA is one of the most promising hardware accelerators because of its flexibility and effective performance with lower power consumption.

Currently, many IoT/edge devices are required to make some inferences, namely deep learning computations, based on the input from the sensors equipped on them, to obtain effective autonomous driving systems, intelligent robot operations, etc. Parameters for deep learning-based methods need to be continuously updated based on the neural network training results. Training the neural networks requires us to provide sufficient computational power as well as a massive amount of learning data; thus, this training cannot be achieved on the IoT/edge side. Therefore, we need to establish a software development workflow or environments to obtain cooperation between the cloud and IoT/edge. However, offloading the training tasks to cloud servers makes the situation worse with regards to total power consumption, including the cloud servers and IoT/edge devices.

To tackle these problems, we are developing a power-performance optimization framework for virtualized cloud platforms. This framework assumes that IoT/edge devices with FPGAs offload their neural network training and optimization to cloud servers, and obtain the training and optimization results via the internet. Our framework attempts to minimize the power consumption of the virtualized cloud servers considering the difference in computational workloads between the deep learning training tasks and High-Level Synthesis tasks.

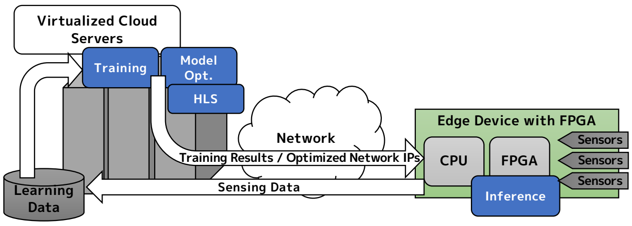

Fig. 1 Expected Task Offloading Workflow to Cloud Servers

2. Expected Task Offloading Workflow from IoT/Edge Devices to Cloud Servers

This section describes our expected workflow to offload deep learning computations and optimizations from IoT/edge devices to cloud servers. Fig. 1 shows our expected workflow of offloading heavy computations to cloud servers from IoT/edge devices. This workflow is based on our previous work [1–3].

For neural network development, training, and optimization, Python-based environments (like Keras [4] and Tensorflow [5]) are significantly popular, thus we assume that they are being used. Once the users develop their deep learning-based applications with Keras, the applications can be used for training and model optimizations as they are. After the training and optimization, the users can also obtain IP logics of the network by using HLS (High-Level Synthesis) tools [6] on the server. To utilize the HLS tools, the Keras-based program must be translated into C/C++ using a translation tool [7].

The IoT/edge devices download the IP logics and trained network parameters from the servers and apply them to their FPGAs.

[1] School of Information Science, Meisei University, 2-1-1 Hodokubo, Hino, Tokyo 191–8506, Japan
[a)] yasutaka.wada@meisei-u.ac.jp

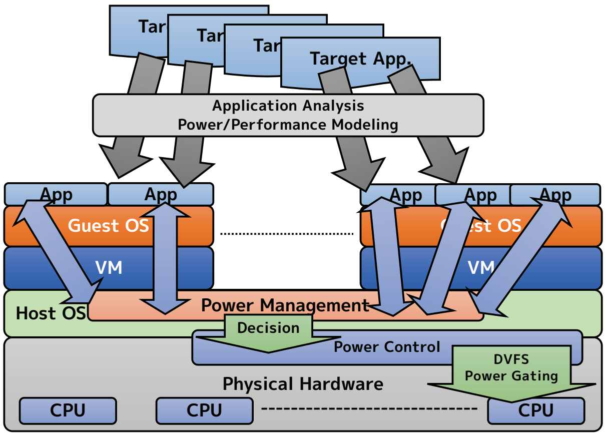

Fig. 2 The Proposed Power-Performance Optimization Framework

We can dynamically reconfigure the FPGAs with these updated logics and parameters by partial reconfiguration functionalities. While operating the devices, we can also capture sensing data with the sensors equipped on the devices. This sensing data can also be shared with the cloud servers, and then used for additional neural network training to obtain more accurate training results.

To minimize the neural network structure and size, we also propose to utilize multiple single-functioned small neural networks in parallel in the workflow. Dividing heavy inference tasks into multiple small networks makes it possible to shorten the overhead for the training, HLS, and partial reconfiguration [1, 2].

3. Power-Performance Optimization on Virtualized Cloud Servers

This section explains our software framework overview to optimize the power performance on virtualized cloud platforms. Fig. 2 shows the relationship between the applications, virtual machines, and host machines.

3.1 Power-Performance Modeling of Applications

To apply power-performance optimization, first, the structure of the given applications must be analyzed. Second, we also need to understand how the hardware reacts to DVFS (Dynamic Voltage/Frequency Scaling) and power gating operations. By utilizing these pieces of information, power-performance models for each combination of the application and the hardware can be developed. Based on the models, we can insert the API calls to apply DVFS into applications.

In this framework, we propose extending the PomPP Library and Tools [8,9] to analyze the applications and develop the power-performance modeling of these applications on the server. Most of the neural network computations should have similar characteristics because their primary computations employ matrix-matrix multiplications or matrix-vector multiplications. Furthermore, HLS tasks are CPU intensive and it is relatively easy to analyze their workloads.

3.2 Power-Performance Control on Virtualized Servers

In virtualized platforms, the physical hardware cannot be controlled directly from the guest OS. Moreover, as the applications run on virtual machines are isolated, it is difficult to obtain precise information about them. Application analysis framework, like the PomPP Tools mentioned in the previous section, can resolve the latter problem. However, in the application of power-performance optimization with hardware control like DVFS, safe interfaces are required to request the host OS to apply DVFS to the physical hardware. Furthermore, on the virtualized platforms, multiple virtual machines run on the same physical machine, and the conflict among the power-performance optimization requests from various virtual machines need to resolved based on their situations.

To apply power-performance optimization on such a virtualized platform, we are developing APIs/libraries that allow the virtual machines to send requests to the host OS by extending virtualized system software like KVM [10] and Qemu [11]. In our strategy, the host OS captures and summarizes the requests from the various virtual machines and decides how to control the physical hardware. For example, if the user requests that applications not be slowed down, the host OS selects the highest frequency among the requests given by the virtual machines.

4. Conclusions

This paper describes our strategy to realize energy efficient neural network training on virtualized cloud platforms, assuming the cooperation with IoT/edge devices equipped with FPGAs. The IoT/edge devices offload neural network training tasks to cloud servers for fast training. In this strategy, our framework applies power-performance optimization for each task (deep learning application), and the task sends the requests for the guest OS to apply DVFS based on the optimization result. The host OS gathers and summarizes the request from the guest OSs and then controls the physical hardware.

Currently, we are developing the power-performance control APIs/libraries between the guest OSs and the host OS. As additional future work, we would like to extend the proposed framework to handle more complex applications and to utilize more detailed application analysis information given by a parallelizing compiler.

Acknowledgments This work was patially supported by JSPS KAKENHI Grant Number 17K12665 and 18K19786.

References

[1] Aoto, M. et al.: Towards the Improvement of Training Efficiency and Image Recognition Accuracy for an FPGA Controlled Mini-Car by Offloading Neural Network Training, *Proc. of FPT2019* (2019).

[2] Aoto, M. et al.: FPGA Implementation of High-speed and High-accuracy Image Recognition Using Multiple Single-Function Neural Networks, *IEICE Tech. Rep.*, Vol. 119, No. 208, pp. 57–62 (2019).

[3] Aoto, M. et al.: An FPGA based Autonomous Driving Car Design using Multiple Simple Neural Networks for Decision Making, *Proc. of HEART2019* (2019).

[4] Keras: The Python Deep Learning library: https://keras.io/.

[5] TensorFlow: https://www.tensorflow.org/.

[6] Vivado Design Suite: https://www.xilinx.com/products/design-tools/vivado.html.

[7] Keras2cpp: https://github.com/gosha20777/keras2cpp.

[8] He, Y. et al.: Simple DSL for Power-Performance Modeling with Segmented Linear Models, *ICPP2019, Poster Session* (2019).

[9] PomPP Library and Tools: https://github.com/pompp/pompp_tools.

[10] Kernel-based Virtual Machine: https://www.linux-kvm.org/.

[11] QEMU: https://www.qemu.org/.