Machine Learning aided Computer Architecture Design for CNN Inferencing Systems

Christopher A. Metz
Supervisor: Rolf Drechsler
Institute of Computer Science, University of Bremen
Bremen, Germany
cmetz@uni-bremen.de

***Abstract*—Efficient and timely calculations of Machine Learning (ML) algorithms are essential for emerging technologies like autonomous driving, the Internet of Things (IoT), and edge computing. One of the primary ML techniques used in such systems is Convolutional Neural Networks (CNNs), which demand high computational resources. This requirement has led to the use of ML accelerators like GPGPUs to meet design constraints. However, selecting the most suitable accelerator involves Design Space Exploration (DSE), a process that is usually time-consuming and requires significant manual effort.**
Our work presents approaches to expedite the DSE process by identifying the most appropriate GPGPU for CNN inferencing systems. We have developed a quick and precise technique for forecasting the power and performance of CNNs during inference, with a MAPE of 5.03% and 5.94%, respectively. Our approach empowers computer architects to estimate power and performance in the early stages of development, reducing the necessity for numerous prototypes. This saves time and money while also improving the time-to-market period.

***Index Terms*—Energy Efficiency, Power and Performance Estimation, Machine Learning**

I. Introduction

Advancements in Machine Learning (ML) and Artificial Intelligence (AI) have yielded impressive results. To ensure fast and efficient calculations, AI accelerators are increasingly being utilized. The latest developments include GPGPUs designed specifically for ML training and inferencing, which can consume up to 700 watts per GPGPU. As most High-Performance Computing (HPC) systems come equipped with multiple GPUs per machine, the power consumption of ML and AI systems presents new challenges [1]–[5].

An example of extreme power consumption in HPC can be seen in the Summit, a supercomputer with 27,648 NVIDIA Volta GPUs that consume 13 million watts [6]. However, by implementing power savings of 5%, significant cost savings of up to 1 million dollars can be achieved [7]. Smaller Internet of Things (IoT) devices may experience increased power consumption when performing ML inferencing. For example, object recognition on an Nvidia Jetson TX1 can use up to 7 watts of power, but offloading the same task to the cloud can reduce power consumption to 2 watts [8]. Hence, offloading ML and AI workloads to the cloud can be a promising power-saving solution for ML-enabled IoT applications. However, the feasibility of offloading ML workloads to the cloud depends on available bandwidth. In cases where offloading is not possible, local execution may be necessary. These challenges make it difficult for computer architects to design appropriate ML inferencing systems due to the wide range of design options available.

There are different ways to explore design space for ML inferencing computer architecture, but two main approaches stand out: 1) simulation and 2) ML-based predictors. However, both have their drawbacks. For example, simulators like GPGPU-Sim or GPUocelot run GPU applications on CPUs for simulation, which leads to significantly slower simulations than on real devices due to CPUs not having the same high parallelization ability as GPUs. ML-based predictors aim to provide fast and accurate estimations, but most require specific configuration and profiling of the application on a real GPGPU first to collect performance counters. Since performance counters are not standardized across all Nvidia GPUs, it is possible that the required counter is unavailable or is collected differently than in the original approach, making it impossible to apply the approach or result in inaccurate results [8]. However, none of these approaches consider the option to offload the ML workloads to cloud or edge systems.

This work was partly supported by the Data Science Center of the University of Bremen (DSC@UB)

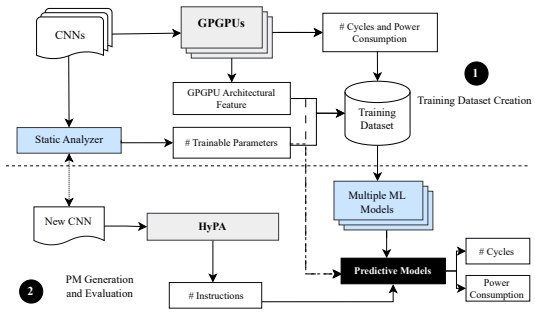


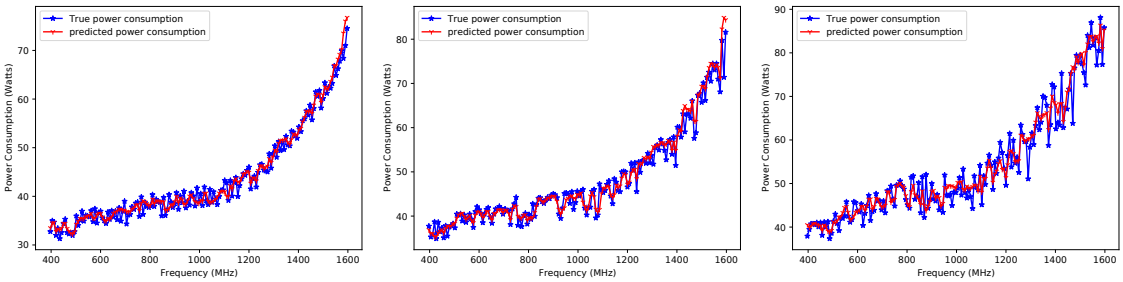
Fig. 1. Methodology for estimating performance and power has been adapted from [2].

This work addresses the limitations of simulation and current ML-based predictors for predicting the power and performance of ML inferencing on GPGPUs. Our study presents recent approaches to overcome these obstacles. Our contributions mainly include the following:

- We developed several predictive models available for predicting the power and performance of CNNs when running on GPGPUs [1]–[5].
- We developed a hybrid PTX Analyzer that can collect runtime-dependent features for power and performance estimation without executing on real GPUs. This solution should also overcome the slow execution time of simulators [8].

II. Power and Performance Estimation

Figure 1 briefly overview our process for estimating power and performance when developing ML models. To ensure accurate results, we train multiple machine learning models (e.g., K-Nearest Neighbor, Decision Tree, Random Forest Tree) for each specific task (i.e., power or performance prediction), which helps improve each model's accuracy. As predictions must be made during the early design stages, we focus on not runtime-dependent features. This includes utilizing



(a) Efficientnetb0 (b) Efficientnetb1 (c) Efficientnetb2

Fig. 2. Comparison of predicted and real power consumption for three CNNs with different frequencies between 397MHz and 1590MHz on the Nvidia V100S GPGPU [5]

hardware specifications such as the size and factor of the GPGPU, the number of cores, the frequency, and the available memory. Additionally, we consider features that describe the ML application (e.g., neural networks) that consist of varying layers and neurons. This approach allows us to develop ML models that are both effective and efficient [1]–[5].

Additionally, we have created a new tool called *Hybrid PTX Analyzer* (HyPA) to account for the intricacies of the compiled ML Model. This tool lets us determine the exact number of executed instructions in the PTX without running the code on physical devices. To achieve this, we simulate critical code sections such as loops or if-statements to construct an accurate control flow graph that encompasses all necessary instructions. Thus, we can also consider runtime-dependent features without executing GPU applications on actual devices [8].

III. Experimetal Results.

In the following, we present power and performance estimation results of ML model inferencing on GPGPUs based on [1]–[5]. The power prediction for various frequencies of the Nvidia V100S is depicted in fig. 2 [5]. In our studies, the Random Forest Trees achieve a *Mean Absolute Percentage Error* (MAPE) of 5.03% and a R^2-Score of 0.9561 for the power prediction for different CNNs at different frequencies. This methodology can also be applied to other GPUs, enabling the creation of predictive models for each GPU that can forecast power usage for different Neural Networks at varying frequencies.

The performance prediction (i.e., number of cycles) for different Neural Networks is illustrated in fig. 3. As the results demonstrate, the K-Nearest Neighbors Algorithm achieved a MAPE of 5.94% [2].

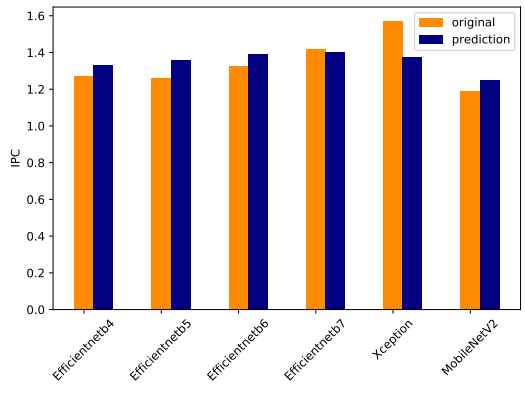


Fig. 3. Prediction results for number of cycles [2].

The results show that our methodology allows the generation of fast and accurate predictive models for estimating power and performance. This is beneficial for computer architects in navigating the design space and identifying the optimal GPGPU.

IV. Conclusion and Future Work

In our upcoming projects, we aim to incorporate optimization techniques to search for the best GPGPU to enhance ML model inference while considering factors such as limited power supply and desired performance. Additionally, we intend to devise approaches to discern whether offloading would adhere to the constraints or if executing locally would be more advantageous. We have developed a REST API for offloading ML workloads and are currently studying the power and performance characteristics at various bandwidths and latencies.

We plan to merge power and performance prediction for GPGPUs with the findings from our offloading analysis. This will help us identify the most suitable GPGPU for local execution. As a result, computer architects and system designers will be able to reduce the number of prototypes they need to build.

References

[1] C. A. Metz, M. Goli, and R. Drechsler, "Early Power Estimation of CUDA-Based CNNs on GPGPUs: Work-in-Progress," in *Proceedings of the 2021 International Conference on Hardware/Software Codesign and System Synthesis*, ser. CODES/ISSS '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 29–30.

[2] ——, "Fast and Accurate: Machine Learning Techniques for Performance Estimation of CNNs for GPGPUs," in *2023 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, 2023, pp. X–Y.

[3] ——, "Pick the Right Edge Device: Towards Power and Performance Estimation of CUDA-based CNNs on GPGPUs," *CoRR*, vol. abs/2102.02645, 2021.

[4] ——, "ML-based Power Estimation of Convolutional Neural Networks on GPGPUs," in *2022 25th International Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS)*, 2022, pp. 166–171.

[5] ——, "Towards Neural Hardware Search: Power Estimation of CNNs for GPGPUs with Dynamic Frequency Scaling," in *Proceedings of the 2022 ACM/IEEE Workshop on Machine Learning for CAD*, ser. MLCAD '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 103–109.

[6] F. Foertter, "Summit GPU Supercomputer Enables Smarter Science," https://developer.nvidia.com/blog/summit-gpu-supercomputer-enables-smarter-science/, 2018, accessed: 22.03.2022.

[7] J. Guerreiro, A. Ilic, N. Roma, and P. Tomás, "GPU Static Modeling Using PTX and Deep Structured Learning," *IEEE Access*, vol. 7, pp. 159 150–159 161, 2019.

[8] C. A. Metz, C. Plump, B. J. Berger, and R. Drechsler, "HyBrid PTX Analysis for GPU accelerated CNNs inferencing aiding Computer Architecture Design," in *2023 Forum on Specification & Design Languages (FDL)*, 2023, accepted for publication.