QAPPA: QUANTIZATION-AWARE POWER, PERFORMANCE, AND AREA MODELING OF DNN ACCELERATORS

Ahmet Inci [1] **Siri Garudanagiri Virupaksha** [1] **Aman Jain** [1] **Venkata Vivek Thallam** [1] **Ruizhou Ding** [1] **Diana Marculescu** [1 2]

ABSTRACT

As the machine learning and systems community strives to achieve higher energy-efficiency through custom DNN accelerators and model compression techniques, there is a need for a design space exploration framework that incorporates quantization-aware processing elements into the accelerator design space while having accurate and fast power, performance, and area models. In this work, we present *QAPPA*, a highly parameterized quantization-aware power, performance, and area modeling framework for DNN accelerators. Our framework can facilitate the future research on design space exploration of DNN accelerators for various design choices such as bit precision, processing element type, scratchpad sizes of processing elements, global buffer size, device bandwidth, number of total processing elements in the the design, and DNN workloads. Our results show that different bit precisions and processing element types lead to significant differences in terms of performance per area and energy. Specifically, our proposed lightweight processing elements achieve up to $4.9\times$ more performance per area and energy improvement when compared to INT16 based implementation.

1 INTRODUCTION

Deep neural networks (DNNs) have achieved remarkable accomplishments across various applications ranging from image recognition (Tan & Le, 2019), object detection (Tan et al., 2020), to natural language processing (Devlin et al., 2019). However, the increasing model size and computational cost of these models become a challenging task for on-device machine learning (ML) endeavours due to the stringent performance per area and energy constraints of the edge devices. To this end, while machine learning practitioners focus on model compression techniques (Han et al., 2016; Ding et al., 2018; Chin et al., 2020), computer architects investigate hardware architectures to overcome the energy-efficiency problem and improve the overall system performance (Inci et al., 2020c;b;a; 2021).

As computing community hits the limits on consistent performance scaling for traditional architectures, there has been a rising interest on enabling on-device machine learning through custom DNN accelerators. As we deeply care about performance per area and energy-efficiency from a hardware point of view, tailored DNN accelerators have shown significant improvements when compared to CPUs and GPUs (Chen et al., 2016; Jouppi et al., 2017; Parashar et al., 2017; Gao et al., 2017). To better understand the trade-offs of various architectural design choices and DNN workloads, there is a need for a design space exploration framework that can rapidly iterate over various designs and generate power, performance, and area (PPA) results. To this end, in this work we present *QAPPA*, a quantization-aware power, performance, and area modeling framework for DNN accelerators.

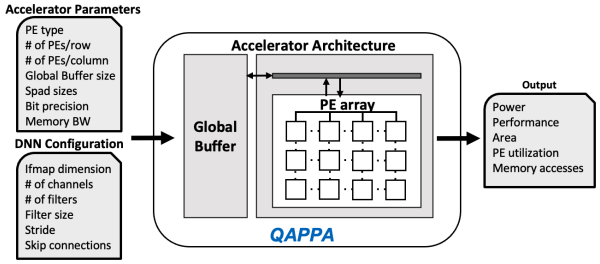

Figure 1. Schematic depicting *QAPPA* framework, with accelerator parameters and DNN configuration as inputs. The framework takes in accelerator parameters and layer-wise DNN configurations and generates power, performance, area results, and statistics on hardware utilization and memory accesses.

[1]Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA [2]Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, Texas, USA. Correspondence to: Ahmet Inci <ainci@andrew.cmu.edu>.

Proceedings of the 4^{th} MLSys Conference, San Jose, CA, USA, 2021.

This work makes the following contributions:

- We present QAPPA, a quantization-aware power, performance, and area modeling framework for DNN accelerators. Our framework can enable future research on design space exploration of DNN accelerators for various design choices such as bit precision, processing element types, scratchpad size of processing elements, global buffer size, device bandwidth, number of total processing elements in the the design, and DNN workloads.

- Our framework provides power, performance, and area results not just for a single hardware design point but for a range of different hardware designs as opposed to prior art (Qi et al., 2017; Cai et al., 2017). Therefore, it can be used to analyze trade-offs of various architectural design choices and DNN workloads at the same time.

The rest of the paper is organized as follows. In Section 2, we present a literature review on power and runtime models for CNNs and design space exploration frameworks for hardware accelerators. In Section 3, we describe the architectural details of the *QAPPA* framework and the details of our methodology for power, performance, and area modeling of DNN accelerators. In Section 4, we show experimental results demonstrating the efficiency of *QAPPA*'s power, performance, area models and the efficacy of lightweight processing elements to conventional designs in terms of performance per area and energy through a suite of case studies. Finally, Section 5 concludes the paper by summarizing the results.

2 Related Work

Prior art has proposed runtime and energy models for DNN workloads (Cai et al., 2017; Qi et al., 2017). However, these models have been implemented specifically for GPU platforms and thus they create an important limitation for a design space exploration of hardware architectures and potentially hardware/ML model co-design opportunities (Gupta & Akin, 2020; Yang et al., 2020). On the other hand, prior art has proposed tools and simulation methodologies for accelerator design. For example, SCALE-Sim (Samajdar et al., 2018) is a cycle accurate, systolic-array based DNN accelerator simulator. Similarly, Aladdin (Shao et al., 2014) is a pre-RTL power and performance accelerator simulator. Although these tools help to perform preliminary analysis on the design space for accelerators in different aspects, they do not incorporate quantization-aware processing elements and they do not generate RTL output based on the input hardware configuration which is an important impediment for enabling deployment of DNNs onto edge devices.

3 Methodology

In this section, we first explain the implementation details and architectural components of our *QAPPA* framework, as depicted in Figure 1. Next, we detail the lightweight processing elements (LightPE) that we implemented in our framework to provide a specialized processing element (PE) type for quantized DNN models. Finally, we explain our power, performance, and area modeling and design space exploration methodology.

3.1 QAPPA Framework

To enable comprehensive design space exploration for DNN accelerators for on-device machine learning, we implemented *QAPPA*, a highly parameterized spatial-array based DNN accelerator framework in RTL. Our framework enables hardware designers and machine learning practitioners to rapidly iterate over various accelerator designs and DNN configurations and better understand trade-offs of different architectural components of the design for dizzying requirements of deploying machine learning models to edge devices. Moreover, hardware designers can also use the automatically generated RTL code to follow the design synthesis flow.

As depicted in Figure 1, *QAPPA* framework is based on spatial-array based accelerators and utilizes row stationary dataflow which has been demonstrated to optimize the data movement in the storage hierarchy (Chen et al., 2016). *QAPPA* features a set of processing elements organized as a 2D array and a global buffer that stores input feature maps, filters, and activations. The number of PEs in each dimension can be tuned for different power, performance, and area requirements. In each PE, there are input feature map, filter, and partial sum scratchpads and a multiply-accumulate (MAC) unit which can be changed based on the desired bit precision. Each of these architectural components can be tuned in a flexible and automated manner to perform a comprehensive design space exploration for on-device edge accelerators.

Lightweight Processing Elements (LightPE)

To enrich the design space of hardware accelerators and achieve a better Pareto-frontier in terms of performance per area and energy-efficiency perspectives, we include LightPE implementations in our framework. LightPEs utilize 8 bits for activations and 4 bits and 8 bits for weights for LightPE-1 and LightPE-2 designs, respectively. As 4 bit and 8 bit quantization techniques for on-device machine learning became prevalent in various computing platforms, we provide these specialized quantization-aware PE types in our *QAPPA* framework to help hardware designers to enrich their design space and hopefully find better Pareto-frontiers.

Besides their low-precision benefits such as reducing the

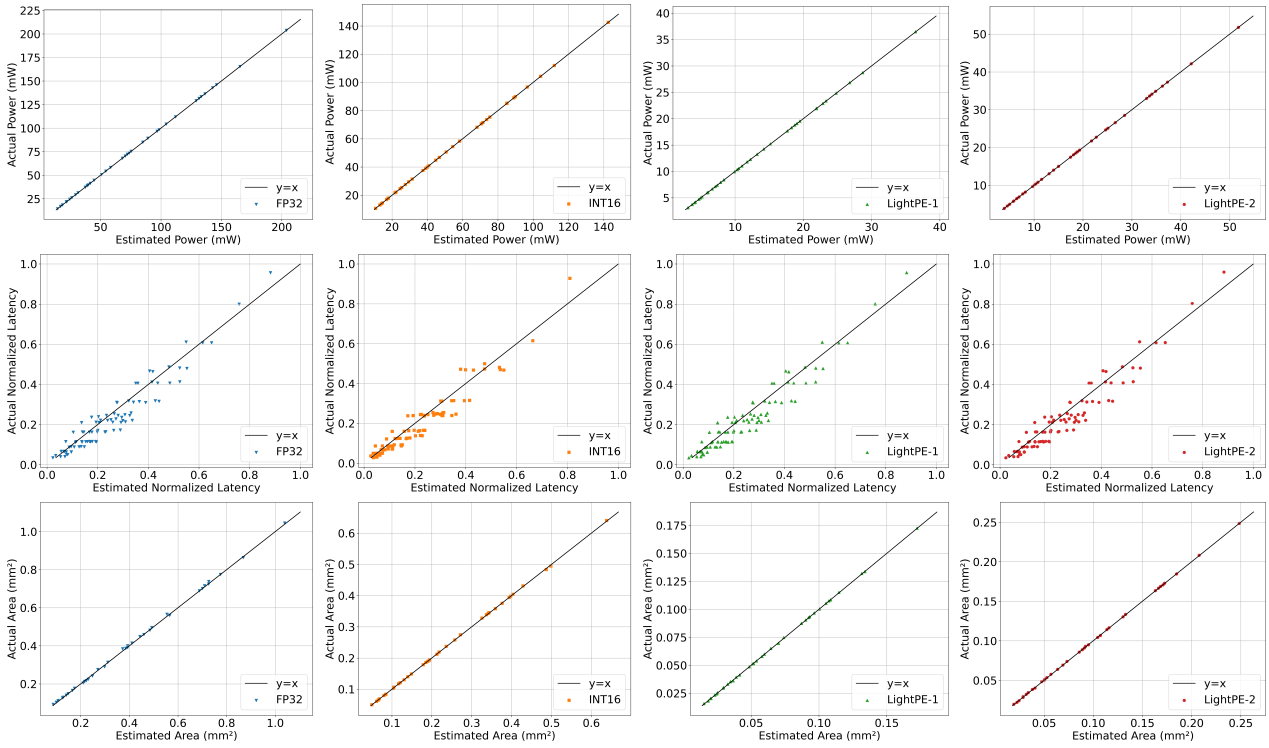

Figure 2. Power (top chart), performance (middle chart), and area (bottom chart) estimation results for various processing element types such as FP32, INT16, LightPE-1, and LightPE-2. Each data point corresponds to a different hardware configuration that can be achieved by using the corresponding processing element type. As it can be seen, the proposed polynomial model agrees closely with the actual values extracted from the synthesis tools.

storage requirements, LightPEs also replace the multiplications with more energy and area-efficient one shift or a limited number of shifts and add operations(Ding et al., 2018). Therefore, they also achieve significant power and area gains when compared to full-precision 32 bit floating point and 16 bit integer based designs with only slight accuracy degradation (Ding et al., 2018). As a results, LightPEs provide an enriched design space for hardware designers to analyze various trade-offs between performance per area and energy.

Power, Performance, and Area Modeling

To build our quantization-aware power, performance, and area models, we use various hardware and DNN configurations. Specifically, to cover this comprehensive design space of hardware accelerators, we run experiments by varying global buffer size, number of PEs per row and column in the 2D PE array, bit precision, and PE type (FP32, INT16, LightPE-1, and LightPE-2). Within each PE, we also vary individual scratchpad sizes for input feature map, filter scratch pad, and partial sum scratchpad.

We use Synopsys Design Compiler and the open-source FreePDK45 which is a commonly used process design kit (Stine et al., 2007) to synthesize our designs to obtain power, area, and initial timing results. We use Synopsys VCS RTL simulator to perform functional verification and collect timing information for various DNN configurations such as VGG-16 (Simonyan & Zisserman, 2014), ResNet-34, and ResNet-50 (He et al., 2016) that are implemented in our testbenches. After collecting power, area, and timing results from these tools, we use polynomial regression models and model selection techniques based on k-fold cross validation (Mosteller & Tukey, 1968) to tune the model parameters and fit the model.

4 RESULTS

In this section, we present power, performance, and area modeling results for each processing element type and perform a design space exploration on VGG-16 (Simonyan & Zisserman, 2014), ResNet-34, and ResNet-50 (He et al., 2016) design spaces to iterate through our framework to demonstrate the flexibility of *QAPPA* for future studies.

As detailed in Section 3, *QAPPA* framework provides power, performance, and area models that significantly speed up the design space exploration. Figure 2 shows the actual and estimated power, performance, and area results for each processing element type such as FP32, INT16, LightPE-1,

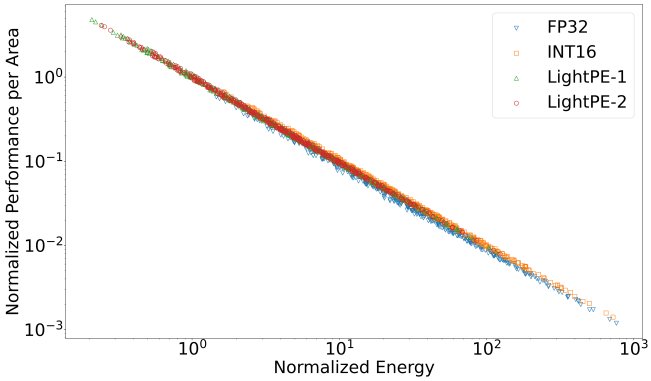

Figure 3. Normalized performance per area vs. normalized energy results with respect to the INT16 hardware configuration with the highest performance per area for VGG-16 design space

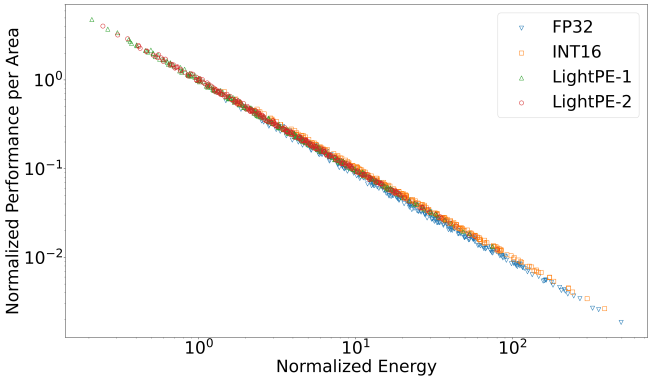

Figure 4. Normalized performance per area vs. normalized energy results with respect to the INT16 hardware configuration with the highest performance per area for ResNet-34 design space

and LightPE-2. Each data point in Figure 2 corresponds to a different hardware accelerator configuration in the comprehensive design space. As shown by the results, *QAPPA*'s PPA models achieve high correlation to the actual PPA values. Figure 2 also shows that the FP32 implementation has the highest area and power cost whereas LightPEs have the lowest area and power results when one processing element is considered. This shows the hardware-efficiency of LightPEs when compared to conventional PE implementations.

To show the efficacy of LightPEs to conventional PE designs, we perform design space exploration on VGG-16 (Simonyan & Zisserman, 2014), ResNet-34, and ResNet-50 (He et al., 2016) design spaces as shown in Figure 3-5. We show the normalized performance per area and normalized energy results for each PE type with respect to the baseline INT16 based implementation with the highest performance per area for the given design space.

Figure 3-5 shows that LightPE implementations consistently outperform conventional INT16 and FP32 implementations in both aspects, which proves their efficacy in terms of hardware-efficiency. Specifically, LightPE-1 and LightPE-2 achieve $4.9\times$ and $4.1\times$ more performance per area and $4.9\times$ and $4.2\times$ energy improvement on average when compared to the best INT16 hardware configuration, respectively. On the other hand, INT16 baseline implementation achieves $1.7\times$ more performance per area and $1.4\times$ energy improvement on average when compared to the best FP32 configuration. These conclusions hold for all models considered in this work VGG-16, ResNet-34, and ResNet-50, thereby showing that the benefits of using lower precision generalize across a variety of models. We conclude that different bit precisions and PE types can lead to significantly different performance per area and energy results which are two critical metrics for hardware designers and machine learning practitioners strive to improve upon.

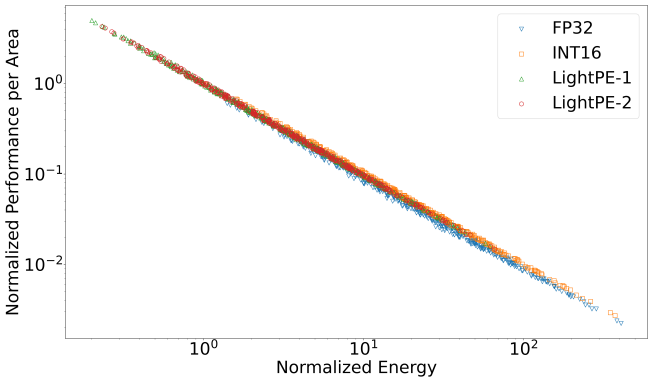

Figure 5. Normalized performance per area vs. normalized energy results with respect to the INT16 hardware configuration with the highest performance per area for ResNet-50 design space

5 Conclusion

In this work, we present *QAPPA*, a quantization-aware highly parameterized power, performance, and area modeling framework for DNN accelerators. Our framework can foster the future research on design space exploration of DNN accelerators for various design choices such as bit precision, processing element type, scratchpad size of processing elements, global buffer size, device bandwidth, number of total processing elements in the the design, and DNN workloads. Our results show that different bit precisions and processing element types lead to significant differences in terms of performance per area and energy. Specifically, LightPE-1 and LightPE-2 achieve $4.9\times$ and $4.1\times$ more performance per area and $4.9\times$ and $4.2\times$ energy improvement on average when compared to the best INT16 hardware configuration, respectively. Therefore, design space exploration of quantization-aware DNN accelerators merits a meticulous analysis that take these factors into account.

ACKNOWLEDGEMENTS

This research was supported in part by National Science Foundation grants CCF No. 1815899 and CSR No. 1815780.

REFERENCES

Cai, E., Juan, D.-C., Stamoulis, D., and Marculescu, D. Neuralpower: Predict and deploy energy-efficient convolutional neural networks. *arXiv preprint arXiv:1710.05420*, 2017.

Chen, Y., Emer, J., and Sze, V. Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks. In *ISCA*, Piscataway, NJ, USA, 2016. IEEE Press.

Chin, T.-W., Ding, R., Zhang, C., and Marculescu, D. Towards efficient model compression via learned global ranking. In *CVPR*, June 2020.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

Ding, R., Liu, Z., Blanton, R. D. S., and Marculescu, D. Lightening the load with highly accurate storage- and energy-efficient lightnns. *ACM Trans. Reconfigurable Technol. Syst.*, 11(3), December 2018. ISSN 1936-7406.

Gao, M., Pu, J., Yang, X., Horowitz, M., and Kozyrakis, C. Tetris: Scalable and efficient neural network acceleration with 3d memory. *SIGARCH Computer Architecture News*, 45(1):751–764, 2017. ISSN 0163-5964.

Gupta, S. and Akin, B. Accelerator-aware neural network design using automl, 2020.

Han, S., Mao, H., and Dally, W. J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding, 2016.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

Inci, A., Bolotin, E., Fu, Y., Dalal, G., Mannor, S., Nellans, D., and Marculescu, D. The architectural implications of distributed reinforcement learning on cpu-gpu systems. *arXiv preprint arXiv:2012.04210*, 2020a.

Inci, A., Isgenc, M. M., and Marculescu, D. DeepNVM++: cross-layer modeling and optimization framework of non-volatile memories for deep learning. *arXiv preprint arXiv:2012.04559*, 2020b.

Inci, A., Isgenc, M. M., and Marculescu, D. Cross-layer design space exploration of nvm-based caches for deep learning. *NVMW*, 2021.

Inci, A. F., Isgenc, M. M., and Marculescu, D. Deepnvm: A framework for modeling and analysis of non-volatile memory technologies for deep learning applications. DATE '20, 2020c.

Jouppi, N., Young, C., Patil, N., Patterson, D. A., and et al. In-datacenter performance analysis of a tensor processing unit. *ISCA*, 2017.

Mosteller, F. and Tukey, J. W. Data analysis, including statistics. In Lindzey, G. and Aronson, E. (eds.), *Handbook of Social Psychology, Vol. 2*. Addison-Wesley, 1968.

Parashar, A., Rhu, M., Mukkara, A., Puglielli, A., Venkatesan, R., Khailany, B., Emer, J., Keckler, S. W., and Dally, W. Scnn: An accelerator for compressed-sparse convolutional neural networks. *ISCA*, 2017.

Qi, H., Sparks, E. R., and Talwalkar, A. Paleo: A performance model for deep neural networks. In *Proceedings of the International Conference on Learning Representations*, 2017.

Samajdar, A., Zhu, Y., Whatmough, P., Mattina, M., and Krishna, T. Scale-sim: Systolic cnn accelerator simulator. *arXiv preprint arXiv:1811.02883*, 2018.

Shao, Y., Reagen, B., Wei, G.-Y., and Brooks, D. Aladdin: A pre-rtl, power-performance accelerator simulator enabling large design space exploration of customized architectures. *ISCA*, 2014.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

Stine, J. E., Castellanos, I., Wood, M., Henson, J., Love, F., Davis, W. R., Franzon, P. D., Bucher, M., Basavarajaiah, S., Oh, J., and Jenkal, R. Freepdk: An open-source variation-aware design kit. In *MSE'07*, 2007.

Tan, M. and Le, Q. EfficientNet: Rethinking model scaling for convolutional neural networks. volume 97, pp. 6105–6114. PMLR, 2019.

Tan, M., Pang, R., and Le, Q. V. Efficientdet: Scalable and efficient object detection. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10778–10787, 2020.

Yang, L., Yan, Z., Li, M., Kwon, H., Lai, L., Krishna, T., Chandra, V., Jiang, W., and Shi, Y. Co-exploration of neural architectures and heterogeneous asic accelerator designs targeting multiple tasks. In *DAC*, pp. 1–6, 2020.