POMMEL: Exploring Off-Chip Memory Energy & Power Consumption in Convolutional Neural Network Accelerators

Alexander Montgomerie-Corcoran, Christos-Savvas Bouganis
Dept. of Electrical and Electronic Engineering, Imperial College London, UK
{alexander.montgomerie-corcoran15, christos-savvas.bouganis}@imperial.ac.uk

Abstract—**Reducing the power and energy consumption of Convolutional Neural Network (CNN) Accelerators is becoming an increasingly popular design objective for both cloud and edge-based settings. Aiming towards the design of more efficient accelerator systems, the accelerator architect must understand how different design choices impact both power and energy consumption. The purpose of this work is to enable CNN accelerator designers to explore how design choices affect the memory subsystem in particular, which is a significant contributing component. By considering high-level design parameters of CNN accelerators that affect the memory subsystem, the proposed tool returns power and energy consumption estimates for a range of networks and memory types. This allows for power and energy of the off-chip memory subsystem to be considered earlier within the design process, enabling greater optimisations at the beginning phases. Towards this, the paper introduces POMMEL, an off-chip memory subsystem modelling tool for CNN accelerators, and its evaluation across a range of accelerators, networks, and memory types is performed. Furthermore, using POMMEL, the impact of various state-of-the-art compression and activity reduction schemes on the power and energy consumption of current accelerations is also investigated.**

Index Terms—**Convolutional Neural Networks, Power Modelling, Machine Learning Acceleration.**

I. Introduction

The increasing demand for CNN solutions for a range of applications has brought forth challenges for the hardware they are being deployed on. With greater numbers of parameters and operations for state-of-the-art CNN models, there is an increased demand for computational power and subsequently, demand for energy and power. This demand for energy and power has an impact on edge-based platforms, where there are strict constraints on both due to the limited resources. Moreover for data-center settings, power-efficient hardware is becoming increasingly more desirable in order to reduce running costs and environmental impact. This has lead to the development of CNN-specific accelerators, which aim to meet the performance demands for various applications and systems, offering advances on the energy and power efficiency of traditional compute platforms such as CPUs and GPUs.

However, despite improvements made to efficiency by the accelerator chips, the off-chip memory component of these CNN accelerator systems still has a significant impact on total power consumption. Off-chip memory access is a requirement for nearly all accelerators, despite recent trends of trying to reduce off-chip memory accesses by keeping more data on-chip. Off-chip memory access is driven by the large memory requirements of modern CNN models, with networks such as EfficientNet-v2 [1] requiring over 60MB to store parameters alone. The problem is further amplified as off-chip memory can have a significant impact on overall power and energy consumption, often outweighing that of the accelerator itself. For example, the fpgaConvNet accelerator has shown that off-chip memory power can contribute 50% of the total system power [2]. As the off-chip memory subsystem is significant in terms of power and energy consumption, its utilisation must be understood in order to improve the efficiency of the entire system.

Towards reducing power and energy consumption of the off-chip memory in a CNN accelerator system, compression and activity reduction schemes for off-chip memory transactions are employed increasingly often. Compression schemes have the ability to reduce time spent utilising the memory bus and DRAM chips, whilst activity reduction schemes impact the dynamic power consumed on the memory bus. However, the effectiveness of such schemes varies across memory types, accelerators, and networks. To choose an optimal scheme or inform the design of improved coding schemes, a better understanding is needed of the features of accelerators and memory types, relating to power and energy.

In this work, an open-source tool, POMMEL[1], is proposed for investigating power and energy consumption for CNN accelerator systems. This tool is used to undertake an investigation into the properties of CNN accelerators and how this affects power consumption for different memory types. Further to this, the impact of popular compression and activity coding schemes on power and energy consumption is also investigated.

The background of this work is covered in Section II, related work is discussed in Section III and an overview of the power modelling tool is given in Section IV. Then common memory types are evaluated in Section V, the accuracy of the tool is evaluated in Section VI, CNN accelerator systems are evaluated in Section VII and finally the impact of different coding schemes are evaluated in Section VIII. The paper is concluded in Section IX.

[1]https://github.com/AlexMontgomerie/pommel

II. Background

CNN Accelerators are used to accelerate the inference of CNN models. Many different CNN Accelerator architectures have been proposed [3]–[7] to address different accelerator settings and requirements. CNN accelerators typically fall into two categories: streaming architectures and systolic arrays, with the latter being much more popular due to their flexibility. Systolic array accelerators can generally be categorised by the type of dataflow they employ [8], which relates to data reuse within the processing engines of the array. These accelerator systems typically come in the form of an Application Specific Integrated Circuit (ASIC) or Field Programmable Gate Array (FPGA) platform that accelerates the convolution layers within CNN models, as well as external DRAM in order to store the large feature-maps and weights that cannot fit into the accelerator's on-chip memory.

The two main components of the memory subsystem are the IO components [9], such as chip interconnects and IO interfaces, and the DRAM chips themselves [10], [11]. Considering the IO power, there are four main sources of power consumption: IO dynamic power, termination power, interconnect power and PHY power [9]. IO dynamic power is dissipated across load capacitances for the IO interface. Termination power is the power consumed by the on-chip terminations. Interconnect power is dissipated across the off-chip interconnects. PHY power summarises the background power consumed by the IO peripherals. The main parameters that affect the IO power components for a given memory system is the bandwidth utilisation of the memory bus as well as the activity along the address and data lines. DRAM power consumption captures the power consumed by cells within DRAM chip. The memory commands (`ACT`, `PRE`, `READ`, `WRITE`) sent to memory have different energy profiles and more frequently commands are sent to the device, the greater the power that is consumed.

Compression schemes have been employed to reduce memory subsystem power consumption of feature-map transfers in several CNN Accelerators [3], [12], [13]. These compression schemes mostly exploit the sparsity in CNN feature-maps by employing methods such as Run Length Encoding (RLE) and Compressed Sparse Row (CSR) representation. Further to feature-map compression, weights compression has also been explored for CNN Accelerators [12]. In particular, Huffman Coding is commonly used to reduce on-chip memory usage.

Activity reduction techniques have also been explored in the design of CNN Accelerators. In activity reduction coding schemes, the objective is to reduce the average number of transitions along a bus, which reduces the dynamic power consumed. For a memory bus, address and data lines have the largest impact on dynamic power and so are typically the focus for activity reduction schemes. More general activity reduction schemes have been proposed such as Bus-Invert (BI) [14] and Probability Based Mapping (PBM) [15]. There is also the Differential Encoding of Feature-maps (DEF) [16] coding scheme which reduces activity for CNN applications.

III. Related Work

The impact of off-chip memory power has been targeted in recent works, and many CNN accelerator works acknowledge the impact of off-chip memory power on their system [12], [17]. The EYERISS accelerator paper [17] in particular has an evaluation of energy for DRAM accesses, for different dataflow types, however this investigation is done for a specific memory subsystem. Other works have explored modelling the power consumption of off-chip memory within the accelerator system, with opportunities for design space exploration [2], [18]. All of these works consider the memory subsystem as fixed, with few opportunities to optimise for off-chip memory power and energy explicitly.

A number of works model memory power consumption [9]–[11], [19], with the purpose to improve on vendor power estimation models, and allow for a high-level investigation of off-chip memory power consumption. The DRAMPower [10] tool focuses on improving the power estimation of DRAM chips themselves through more detailed power models of the DRAM chip. The VAMPIRE [11] tool takes into account variations in DRAM chips to produce a more accurate model based on empirical results. These previous tools focus on power consumption from the DRAM chips perspective without considering the interfacing power, which is a significant aspect of the memory subsystem. The CACTIO-IO [9] modelling tool focuses on this aspect, providing general models of IO power for different memory types and configurations. In this work, the CACTI-IO [9] and the DRAMPower [10] power modelling tools are utilised in order to model IO and DRAM power respectively.

The purpose of this work is to extend the use of power modelling tools for the memory subsystem and build on the insights of previous accelerator works. By modelling the memory transactions of given accelerator and network pairs and gathering power readings through existing power modelling tools, this work enables CNN accelerator designers to rapidly evaluate their memory subsystem power consumption and enable informed decisions on the overall design of the architecture.

IV. Power Modelling Tool

In order to evaluate the power consumption of the memory subsystem in CNN accelerator platforms from a high-level description, as well as the impact of various compression and coding schemes, a power modelling framework needs to be in place. This section details the design of this tool, which we have coined POMMEL. The tool performs off-chip memory power estimation of each partition (i.e. computational stage) in the CNN accelerator's execution by creating a memory access trace for the given accelerator description and performing power estimation on this trace. The tool focuses on the feature-map access as it dominates in terms of the utilisation of the memory bus, due to the computational bottlenecks as well as the large size of feature-maps that are usually 100x larger than the weights in a layer, such as in ResNet [20], VGG [21], AlexNet [22] and MobileNet-v2 [23].

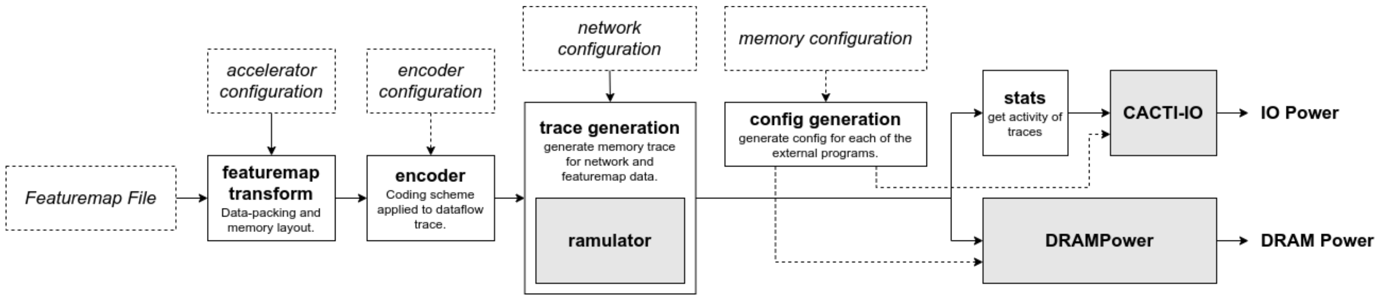

Fig. 1: The POMMEL memory power modelling tool-flow. This represents the flow for each partition for the CNN accelerator's execution of a network.

The tool accepts three configuration files, which are described in Table I. The *accelerator configuration* file describes the key parameters with regards to the memory subsystem. Within it, the feature-map layout describes how the feature-map words occur in memory (row-major or channel-major). The burst-size refers to the memory access burst by the DMA on the accelerator. The *memory configuration* file follows the same specification as is in the DRAMPower tool [10], as the power model parameters are specified in this file. The *network configuration* file specifies the feature-maps as well as the bandwidth for each partition executed on the device. A partition refers to what part of the CNN computation graph is being executed on the accelerator at a time. Systolic array architectures typically execute a single convolution layer in each partition. These network descriptions can be hand-written or generated from SCALE-Sim [24].

Configuration File	Parameters
Accelerator	Word-length
	Burst Size
	Clock Frequency
	Featuremap Layout
Memory	Type
	Number of Chips
	Ranks
	Banks
	Rows
	Columns
	Data Width
Network (per partition)	Bandwidth
	Featuremap Path

TABLE I: Parameters for different configuration files in the POMMEL modelling tool.

In addition to the configuration files, sample feature-maps are also taken into account. These are specified in a feature-maps file which contains a sample of the input and output of each layer in its respective network for a sample of relevant inputs. The tool utilises the Neural Network Distiller Tool [25] to create quantised feature-maps from the ImageNet [22] dataset. Optionally, an encoding scheme configuration file can be specified. This file contains relevant parameters for each feature-map in the network.

An overview of the toolflow is shown in Figure 1. In this figure, the dashed boxes represent the user configuration files that were previously outlined, the white solid-line boxes represent components of the tool-flow, and the grey boxes represent external tools utilised in this framework. The tool takes as an input the feature-maps file. It iterates over each partition in the network emulating the execution of the CNN model on the given accelerator.

feature-map transform: The first stage transforms the feature-map to represent the contents of off-chip memory. This manipulates the out-of-order feature-map into a list of address and data pairs. The words of the feature-map are packed into a memory-bus width word. Then they are organised into a row or channel major format based on the accelerator configuration.

encoder: There is an optional stage next to encode the transformed feature-maps. Based on a encoder configuration, the transformed feature-maps are further manipulated by the specified compression or activity reduction scheme. These encoders affect the data as well as the addresses.

trace generation: Having transformed the feature-maps to represent off-chip memory content, the next step is to generate the traces that represent the memory accesses. Using the utilised bandwidth described in the network configuration file as well as the burst size for data accesses, the memory addresses are then transformed into memory commands. This stage utilises the RAMULATOR tool [26] to help generate accurate commands for the trace.

stats: From the generated memory access trace, metrics are then extracted to be used by the power estimation tools. In particular, the CACTI-IO tool requires the average switching activity as well as the average utilised bandwidth across the memory bus, for both reading and writing.

config generation: The final stage of the tool is to generate the configuration files needed by the DRAM and IO modelling tools. This stage takes information from the POMMEL memory configuration file alongside the metrics generated from the previous stage to modify the DRAMPower and CACTI-IO configuration files. This then allows these tools to be executed in order to get the DRAM and IO power estimations.

Type	Data Width	Clock Freq. (MHz)	Bandwidth (GB/s)
DDR3	32	1332	5.3
DDR3L	32	1066	4.3
DDR4	32	1866	7.5
LP-DDR2	64	1066	8.5
LP-DDR3	32	1600	6.4

TABLE II: Memory configurations.

V. Evaluation of Memory Types

Having established a tool for modelling off-chip memory power/energy consumption, in this section the POMMEL tool will be used to evaluate different memory types that are commonly used for CNN accelerators, and explores their properties. Throughout the evaluation sections, we will refer to different memory configurations based on the type of memory. These different memory configurations are summarised in Table II. All the memory configurations share the same capacity of 1GB. This capacity is sufficient for this investigation, as the largest network (VGG11 [21]) requires less than 318 MB of storage for both parameters and feature-maps.

The first investigation that is performed is exploring how the choice in type of memory affects power consumption. To do this, two metrics are extracted from the established power modelling tools which show how power varies with both bandwidth and activity. For off-chip memory systems, these are the two main variables that affect power consumption. The static power constant is extracted also. These metrics are derived using the model given in Equation 1:

$$\mathrm{P}_{total} = \mathrm{P}_{static} + k_{bw} \cdot b + k_{act} \cdot a \cdot b \qquad (1)$$

where P_{total} is the total memory subsystem power in mW, P_{static} is the static power component in mW, b is the bandwidth utilised by the memory bus in gigabytes per second (GB/s), a is the activity of the words sent along the data-lines of the memory bus in average transitions per bit ($\frac{\mathrm{T}}{\mathrm{b}}$) where T is average transitions and b is bit, k_{bw} is the bandwidth coefficient in GB/s/mW, and k_{act} is the activity coefficient in GT/s/mW. The activity for the data is calculated out of context of the actual rate at which they are sent and thus needs to be scaled by the bandwidth utilised on the memory bus. It is also worth noting that the activity for address lines is not taken into account, as CNN Accelerators usually access data from sequential addresses, and so this parameter is typically constant across all accelerators. The coefficients for this model are derived using linear regression for a uniform set of power predictions generated by the modelling tool. Coefficients for the various memory types are presented in Table III.

DRAM Type	Static Power (mW)	Bandwidth Coefficient (GB/s/mW)	Activity Coefficient (GT/s/mW)
DDR3	768.4	253.7	5.3
DDR3L	268.0	230.7	4.5
DDR4	151.7	171.5	3.1
LP-DDR2	288.5	142.9	20.3
LP-DDR3	157.4	144.1	13.7

TABLE III: Static power and bandwidth and activity coefficients for different memory types.

The results show that the memory types exhibit different properties when it comes to bandwidth and activity. Comparing the values for the coefficients themselves, bandwidth has a much more significant impact on power than activity. And out of the memory types, both DDR3 and DDR3L are the most sensitive to changes in bandwidth compared to the other configurations. In terms of sensitivity to activity, both LP-DDR2 and LP-DDR3 are the most sensitive, suggesting that IO dynamic power can have a significant power contribution for these memory types compared to other memories. DDR3 has the largest static power term by far, suggesting that any optimisations on activity and bandwidth will proportionally have less of an effect on power consumption. DDR4 and LP-DDR3 have the lowest static power values, making them ideal for computationally-bounded accelerator systems where static power dominates due to the low memory bandwidth utilisation.

VI. Evaluation of Power Modelling Accuracy

It should be noted that these power estimations are derived from existing power modelling tools. To evaluate the actual accuracy of these modelling tools, an investigation is undertaken to compare actual readings against the model predictions. A number of readings are taken for a DDR3-based memory system (part `MT41J256M8HX-15E`) on the ZC702 Xilinx FPGA development board.

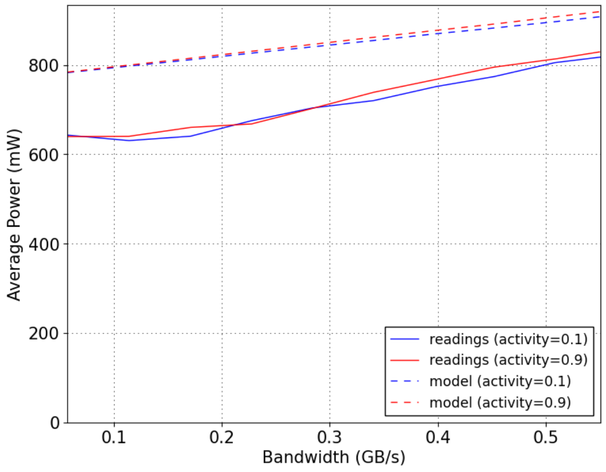

Fig. 2: Comparison between model and actual readings for memory power consumption across different bandwidths. Data activity of both 0.1 and 0.9 are compared.

The comparison of the model predictions and actual readings are presented in Figure 2. In this figure, the model predictions and actual readings for the total memory power consumption are compared for varying bandwidth usage, for low (0.1) and high (0.9) data activity values. The results show that the model assumes a larger static power than what is actually measured, at around 150 mW. Moreover, the actual memory power readings are more sensitive to bandwidth changes compared to the model prediction (seen by the steeper gradient). Nevertheless, the model predictions and actual readings have a similar trend.

Table IV presents the coefficients generated from the memory power model in Equation 1 for the power predictions and actual readings. The actual memory system consumes 140mW more power per GB/s increase in utilised bandwidth and 170mW greater static power. Both the model and actual readings demonstrate similar sensitivity with respect to the

DRAM Type	Static Power (mW)	Bandwidth Coefficient (GB/s/mW)	Activity Coefficient (GT/s/mW)
model	768.4	249.0	5.3
actual	594.0	390.5	5.3

TABLE IV: Comparison of static power and bandwidth and activity coefficients for modelled and actual readings of a `MT41J256M8HX-15E` DDR3 based memory system.

activity, shown by the activity coefficient. In conclusion, even though these power modelling tools exhibit a bias in the power estimation, the uncertainty in their predictions is low, making them valuable for design space exploration.

VII. Evaluation of CNN Accelerator Systems

In this section, the investigation focuses on the power and energy consumption of state-of-the-art CNN accelerators when combined with a specific off-chip memory type. This work utilises the SCALE-SIM tool [24] to generate layer-wise memory bandwidth usage estimations for the accelerators under investigation. We generate configuration files to emulate the behaviour of three research accelerators: ShiDianNao [5], EYERISS [3] and SCNN [27], as well as for a commercial accelerator, the Tensor Processing Unit (TPU) [7], using the SCALE-SIM tool. A batch size of 8 is used across all accelerators. The AlexNet [22], ResNet-18 [20], VGG11 [21], and MobileNet-v2 [23] networks have been used as case studies. Table V shows the type of dataflow and wordlength used across these systems, as well as the required memory bandwidth and measured activity for the execution of ResNet18 [20].

Accelerator	Wordlength	Dataflow	Performance (GOP/s)	Avg. Activity (T/b)
ShiDianNao	16	OS	1.22	0.26
EYERISS	16	RS	2.62	0.30
SCNN	16	IS	3.19	0.39
TPU	8	OS	51.22	0.22

TABLE V: Properties of the various accelerators for the ResNet18 [20] CNN model.

As the available bandwidth and performance for the TPU configuration is considerably higher compared to the research devices, a separate elaboration on the TPU will follow. Using the proposed tool, the power and energy consumption across the research accelerators for various memory types is shown in Figure 3. The figure illustrates variations in average power and energy consumption across the memory types for each accelerator architecture. The results show that the accelerators have similar power consumption profiles, however due to their differences in performance, the lower performance accelerators consume more energy in total. The strong similarity in power consumption profiles is due to the low bandwidth requirements of these smaller research accelerators, meaning that static power consumption dominates in all cases. Furthermore, the obtained results show that both DDR4 and LP-DDR3 have

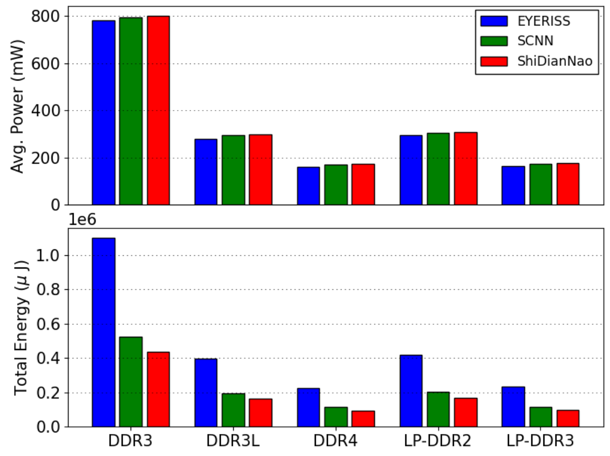

Fig. 3: Comparison of power and energy for different accelerators and memory types for the ResNet18 [20] CNN model.

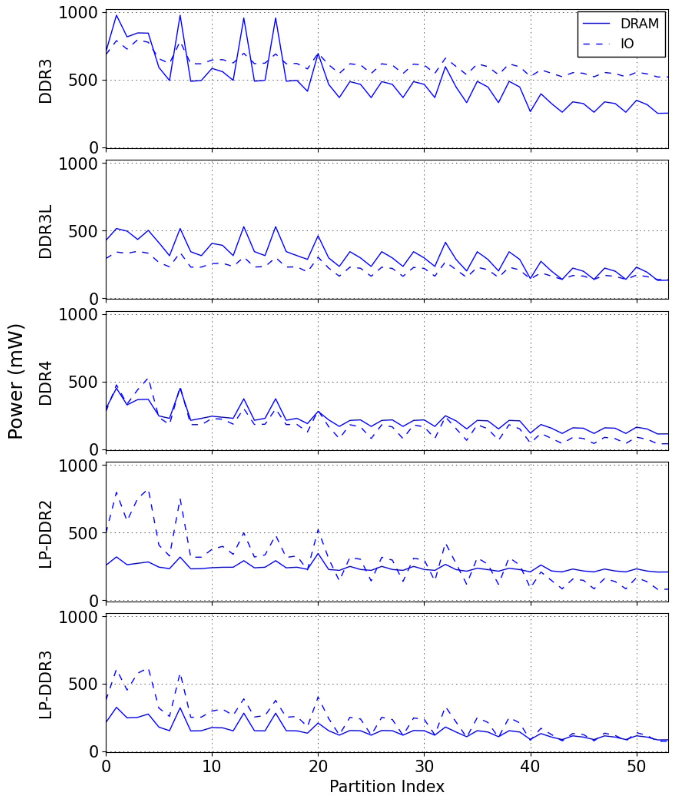

Fig. 4: Comparison of IO and DRAM power for different memory types for a TPU accelerator configuration running the MobileNet-v2 [23] CNN model.

the lowest power and energy consumption across all the accelerators.

Figure 4 illustrates the IO and DRAM power as a function of the partition of the computation for the TPU configuration, when the accelerator is combined with different memory types. The results show that DRAM power and IO power vary across partition indices and memory types. For LP-DDR3 and LP-DDR2, the IO power dominates in the earlier layers

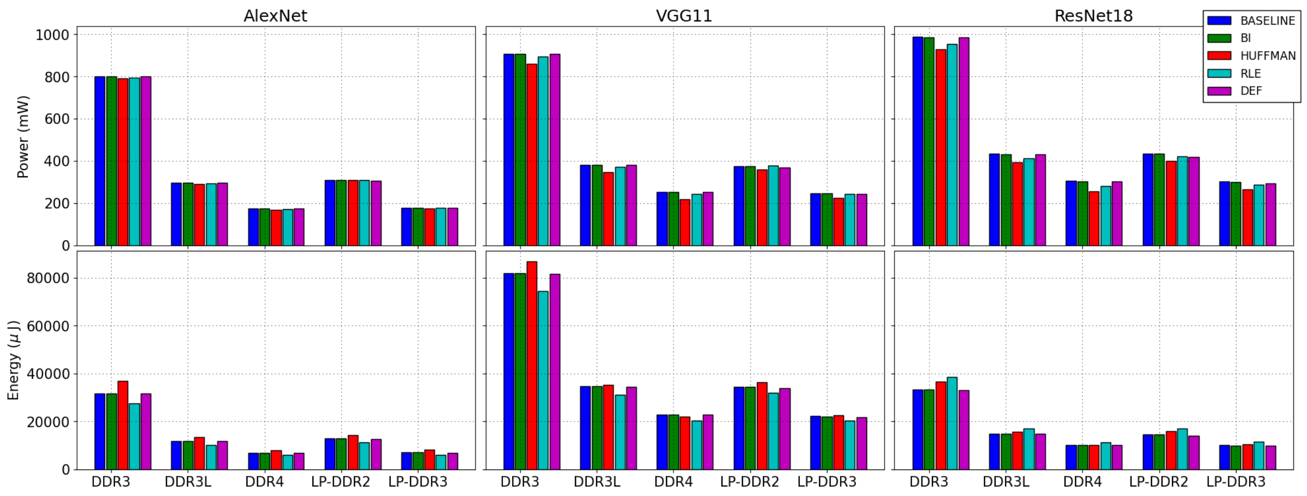

Fig. 5: Comparison of impact of different coding schemes for a TPU accelerator configuration for different networks and memory types.

and reduces further down the network. DDR3 is dominated by static IO power towards the later partition indices in the network. Spikes in power consumption can be observed across the partitions, which is due to the types of layer and whether they are bounded by memory or computation bandwidth. The results highlight the significance of this tool as it provides detailed power consumption per partition, allowing for fine-grain optimisation of power consumption in CNN accelerators.

VIII. Impact of Compression and Activity Reduction Schemes

A popular method for reducing memory power consumption in existing accelerators is to employ compression and activity reduction schemes. This section investigates the impact of compression and coding schemes mentioned in Section II. In particular, the coding schemes Bus-Invert (BI), Huffman, Run-Length Encoding (RLE) and Differential Encoding of Featuremaps (DEF) are investigated. Results with no coding scheme applied are referred to as baseline results. It should be noted that these compression and coding schemes only have a noticeable impact on high-performance accelerators due to the large static power that exists in memory power consumption. We will therefore focus on the TPU accelerator configuration due to its high performance.

Figure 5 illustrates the power and energy impact of various coding schemes and memory pairs for a TPU accelerator configuration running AlexNet [22], VGG11, and ResNet18. The results show that despite the high performance of the TPU accelerator configuration, that only modest energy and power savings (14% energy savings at most) can be achieved by employing these compression and coding schemes. Out of all the schemes, the compression schemes (Huffman and RLE) provide the best reduction in power. These results are in agreement with the model coefficients seen in Table III, where we see that all the accelerators have a greater sensitivity to bandwidth reduction than activity reduction. Moreover, the reductions in power do not necessarily lead to reductions in energy, and in fact for AlexNet [22], the system's energy consumption using the Huffman compression scheme is increased despite the slight reduction in average power.

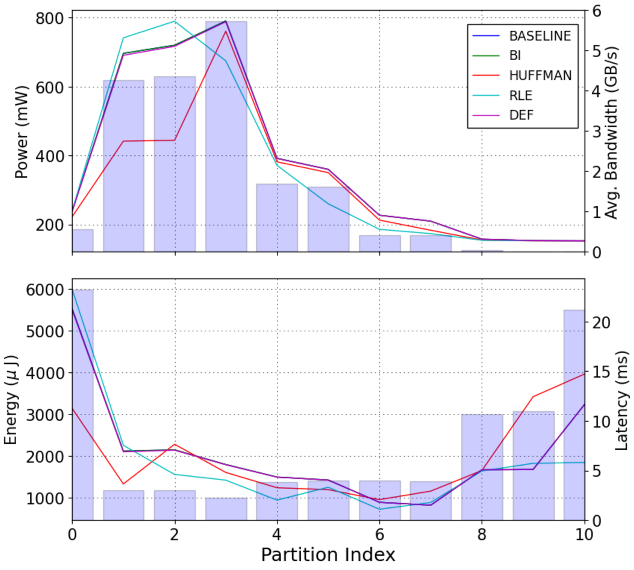

Fig. 6: Partition-level analysis of power and energy for a TPU accelerator configuration for the VGG11 [21] network for different coding schemes. The left axis represents the values for the line plots and the right axis for the bar plots.

Figure 6 provides a partition-level analysis on the impact of the above schemes in power and energy reduction. The results show that Huffman and RLE lead to reductions in power particularly around the early partition indices, where the accelerator exhibits high memory bandwidth. Towards the later partitions, these differences are less prominent due to the low bandwidth requirements. This also leads to the accelerator spending more time in those later partitions, and therefore to consume more energy as opposed to earlier layers.

The above analysis demonstrates that the evaluated activity reduction schemes have minimal impact on the power and energy consumption of the system, whereas the compression schemes have a greater impact on the power and energy reduction as they reduce bandwidth utilisation. Moreover, optimisation on layers where the accelerator spends a higher proportion of its execution time has a larger impact on energy reduction, whereas layers with high bandwidth requirements have more potential for power reduction.

IX. Conclusion

This paper presents the POMMEL off-chip memory power modelling tool for CNN accelerators, which enables CNN accelerator designers to bring off-chip memory power and energy consumption earlier into the design process. The tool integrates a number of widely reputed tools for off-chip power modelling, and its accuracy is evaluated using real-world benchmarks. Using POMMEL, the paper investigates the impact of power and energy consumption of state-of-the-art CNN accelerators for various off-chip memory types as well as power reduction approaches, and draws conclusions on their potential.

References

[1] Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le, "Self-Training With Noisy Student Improves ImageNet Classification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 687–10 698.

[2] A. Montgomerie-Corcoran, S. I. Venieris, and C. Bouganis, "Power-Aware FPGA Mapping of Convolutional Neural Networks," in *2019 International Conference on Field-Programmable Technology (ICFPT)*, Dec. 2019, pp. 327–330.

[3] Y. Chen, J. Emer, and V. Sze, "Eyeriss: A Spatial Architecture for Energy-Efficient Dataflow for Convolutional Neural Networks," in *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, Jun. 2016, pp. 367–379.

[4] T. Chen, Z. Du, N. Sun, J. Wang, C. Wu, Y. Chen, and O. Temam, "DianNao: A small-footprint high-throughput accelerator for ubiquitous machine-learning," in *Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS '14. New York, NY, USA: Association for Computing Machinery, Feb. 2014, pp. 269–284.

[5] Z. Du, R. Fasthuber, T. Chen, P. Ienne, L. Li, T. Luo, X. Feng, Y. Chen, and O. Temam, "ShiDianNao: Shifting vision processing closer to the sensor," in *2015 ACM/IEEE 42nd Annual International Symposium on Computer Architecture (ISCA)*, Jun. 2015, pp. 92–104.

[6] S. I. Venieris and C.-S. Bouganis, "fpgaConvNet: A Toolflow for Mapping Diverse Convolutional Neural Networks on Embedded FPGAs," in *NIPS 2017 Workshop on Machine Learning on the Phone and Other Consumer Devices*, 2017.

[7] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers, R. Boyle, P. Cantin, C. Chao, C. Clark, J. Coriell, M. Daley, M. Dau, J. Dean, B. Gelb, T. V. Ghaemmaghami, R. Gottipati, W. Gulland, R. Hagmann, C. R. Ho, D. Hogberg, J. Hu, R. Hundt, D. Hurt, J. Ibarz, A. Jaffey, A. Jaworski, A. Kaplan, H. Khaitan, D. Killebrew, A. Koch, N. Kumar, S. Lacy, J. Laudon, J. Law, D. Le, C. Leary, Z. Liu, K. Lucke, A. Lundin, G. MacKean, A. Maggiore, M. Mahony, K. Miller, R. Nagarajan, R. Narayanaswami, R. Ni, K. Nix, T. Norrie, M. Omernick, N. Penukonda, A. Phelps, J. Ross, M. Ross, A. Salek, E. Samadiani, C. Severn, G. Sizikov, M. Snelham, J. Souter, D. Steinberg, A. Swing, M. Tan, G. Thorson, B. Tian, H. Toma, E. Tuttle, V. Vasudevan, R. Walter, W. Wang, E. Wilcox, and D. H. Yoon, "In-datacenter performance analysis of a tensor processing unit," in *2017 ACM/IEEE 44th Annual International Symposium on Computer Architecture (ISCA)*, Jun. 2017, pp. 1–12.

[8] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient Processing of Deep Neural Networks: A Tutorial and Survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, Dec. 2017.

[9] N. P. Jouppi, A. B. Kahng, N. Muralimanohar, and V. Srinivas, "CACTI-IO: CACTI With OFF-Chip Power-Area-Timing Models," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 7, pp. 1254–1267, Jul. 2015.

[10] K. Chandrasekar, B. Akesson, and K. Goossens, "Improved Power Modeling of DDR SDRAMs," in *2011 14th Euromicro Conference on Digital System Design*, Aug. 2011, pp. 99–108.

[11] S. Ghose, A. G. Yaglikçi, R. Gupta, D. Lee, K. Kudrolli, W. X. Liu, H. Hassan, K. K. Chang, N. Chatterjee, A. Agrawal, M. O'Connor, and O. Mutlu, "What Your DRAM Power Models Are Not Telling You: Lessons from a Detailed Experimental Study," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 2, no. 3, pp. 1–41, Dec. 2018.

[12] M. Capra, B. Bussolino, A. Marchisio, M. Shafique, G. Masera, and M. Martina, "An Updated Survey of Efficient Hardware Architectures for Accelerating Deep Convolutional Neural Networks," *Future Internet*, vol. 12, no. 7, p. 113, Jul. 2020.

[13] J. J. Zhang, P. Raj, S. Zarar, A. Ambardekar, and S. Garg, "CompAct: On-chip Compression of Activations for Low Power Systolic Array Based CNN Acceleration," *ACM Transactions on Embedded Computing Systems*, vol. 18, no. 5s, pp. 47:1–47:24, Oct. 2019.

[14] M. R. Stan and W. P. Burleson, "Bus-invert coding for low-power I/O," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 3, no. 1, pp. 49–58, Mar. 1995.

[15] S. Ramprasad, N. Shanbhag, and I. Hajj, "A coding framework for low-power address and data busses," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 7, no. 2, pp. 212–221, Jun. 1999.

[16] A. Montgomerie-Corcoran and C. Bouganis, "DEF: Differential Encoding of Featuremaps for Low Power Convolutional Neural Network Accelerators," in *2021 26th Asia and South Pacific Design Automation Conference (ASP-DAC)*. ACM, 2021.

[17] Y. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks," *IEEE Journal of Solid-State Circuits*, 2017.

[18] E. Cai, D.-C. Juan, D. Stamoulis, and D. Marculescu, "NeuralPower: Predict and Deploy Energy-Efficient Convolutional Neural Networks," in *Asian Conference on Machine Learning*. PMLR, Nov. 2017, pp. 622–637.

[19] J. Lucas and B. Juurlink, "MEMPower: Data-Aware GPU Memory Power Model," in *Architecture of Computing Systems – ARCS 2019*, M. Schoeberl, C. Hochberger, S. Uhrig, J. Brehm, and T. Pionteck, Eds. Cham: Springer International Publishing, 2019, vol. 11479, pp. 195–207.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 770–778.

[21] K. Simonyan, "Very Deep Convolutional Networks for Large-Scale Image Recognition." in *International Conference on Learning Representations*, 2015.

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *NIPS*, 2012.

[23] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.

[24] A. Samajdar, Y. Zhu, P. Whatmough, M. Mattina, and T. Krishna, "SCALE-Sim: Systolic CNN Accelerator Simulator," *arXiv:1811.02883 [cs]*, Feb. 2019.

[25] N. Zmora, G. Jacob, L. Zlotnik, B. Elharar, and G. Novik, "Neural Network Distiller: A Python Package For DNN Compression Research," *arXiv:1910.12232 [cs, stat]*, Oct. 2019.

[26] Y. Kim, W. Yang, and O. Mutlu, "Ramulator: A Fast and Extensible DRAM Simulator," *IEEE Computer Architecture Letters*, vol. 15, no. 1, pp. 45–49, Jan. 2016.

[27] A. Parashar, M. Rhu, A. Mukkara, A. Puglielli, R. Venkatesan, B. Khailany, J. Emer, S. W. Keckler, and W. J. Dally, "SCNN: An accelerator for compressed-sparse convolutional neural networks," in *2017 ACM/IEEE 44th Annual International Symposium on Computer Architecture (ISCA)*, Jun. 2017, pp. 27–40.