Analysis of Quantization Across DNN Accelerator Architecture Paradigms

Tom Glint
IIT Gandhinagar, India
tom.issac@iitgn.ac.in

Chandan Kumar Jha
DFKI, Germany
chandan.jha@dfki.de

Manu Awasthi
Ashoka University, India
manu.awasthi@ashoka.edu.in

Joycee Mekie
IIT Gandhinagar, India
joycee@iitgn.ac.in

***Abstract*—Quantization techniques promise to significantly reduce the latency, energy, and area associated with multiplier hardware. This work, to the best of our knowledge, for the first time, shows the system-level impact of quantization on SOTA DNN accelerators from different digital accelerator paradigms. Based on the placement of data and compute site, we identify SOTA designs from Conventional Hardware Accelerators (CHA), Near Data Processors (NDP), and Processing-in-Memory (PIM) paradigms and show the impact of quantization when inferencing CNN and Fully Connected Layer (FCL) workloads. We show that the 32-bit implementation of SOTA from PIM consumes less energy than the 8-bit implementation of SOTA from CHA for FCL, while the trend reverses for CNN workloads. Further, PIM has stable latency while scaling the word size while CHA and NDP suffer 20% to 2× slow down for doubling word size.**

I. Introduction

Various DNN accelerators have been proposed to tackle the power and memory wall issues associated with processing Deep Neural Networks (DNN). They are used for both training and inferencing these DNNs. The focus of these accelerators is to optimize a set of the following metrics: accuracy, latency, energy, and area [1]. Traditionally, training is performed over a 32-bit IEEE 754 floating number system to achieve high accuracy. However, for inference, various accelerators adopt different number systems with various quantization and approximation levels to improve the abovementioned metrics. Table I from [2] shows an example of how these metrics change at the multiplier level for different quantization levels and different number systems. It also shows how accuracy changes with quantization. However, these benefits in metrics for the multiplier do not directly translate to the system level, as the flow and storage of data also dictate the overall system latency and energy. Further, there exist different DNN accelerator paradigms [1], which have contrasting energy and latency consumption profiles. Based on the primary storage location of the data and the site for computation [3], digital DNN accelerators can be mainly classified into Conventional Hardware Accelerators (CHA), Near Data Processors (NDP), and Processing-in-Memory (PIM) paradigms. This work analyzes the change in latency and energy for SOTA architectures from each of these paradigms, for CNN and fully connected workloads, to gauge the degree of benefits from quantization as higher quantization, especially for deeper networks, leads to accuracy loss [1], [2].

This work is supported through grants received from Science and Engineering Research Board (SERB), Government of India, under SERB-CRG grant CRG/2018/005013, SERB-MATRICS grant MTR/2019/001605, and SERB-SUPRA grant SPR/2020/000450, and funds received for YFRF Visvesvaraya PhD fellowship from MEITY and Semiconductor Research Corporation (SRC) through contracts 2020-IR-3005 and 2020-IR-2980, and is partially supported through Ashoka University startup and Huawei Technologies India grants.

II. State-of-the-art Accelerators

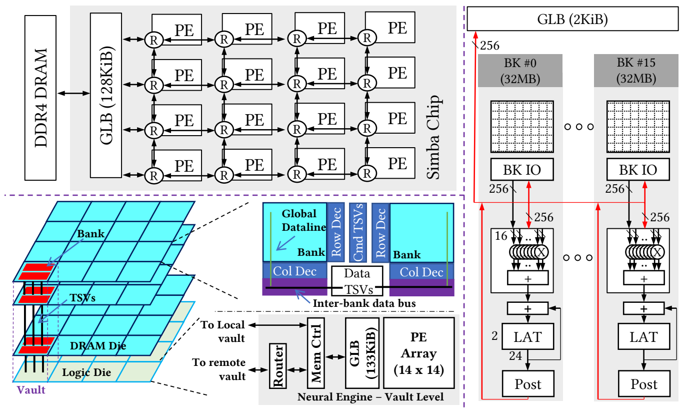

Fig. 1. Top Left: Simba Architecture [4] (CHA); Right: AiM Architecture [5] (PIM); Bottom Left: Tetris Architecture [6] (NDP)

We use the following standard to determine the best architecture within each paradigm: (i) highest peak performance (TOPS), (ii) post-layout or hardware realized architectures, and (iii) comparable output quality results to the traditional hardware. Based on these, we select Simba [4], Tetris [6], and AiM [5] for CHA, NDP, and PIM, respectively.

Simba is a chiplet-based architecture that operates at 2 GHz with 1024 Multiply-Accumulate (MAC) units, with data for computation fetched from external main memory. The 1024 MACs are spread across 16 Processing Elements (PEs) as vectorized MACs, as shown in Fig. 1. At each PE, there are buffers for storing input activations, filters, and outputs of DNNs. These buffers are used to exploit the data reuse property of DNNs where the same input word is convolved with different filters, and similarly, the same weight word is used across different input words. Thus the chip can efficiently cache the data required for processing and avoid the cost of fetching data from external memory repeatedly. We have modeled the external memory of Simba as a single DDR4 memory with 25GBps bandwidth. However, any access to external memory is energy intensive.

Aim uses PIM paradigm based on GDDR6 DRAM and has 32 banks of memory across two dies, with each bank equipped with a vector MAC with 16 multipliers operating at 1 GHz and operating on BFloat16. Since data is fetched from the bank and used for compute in the periphery, access energy is low. When processing, the input activation is transferred to the Global Buffer (GLB) and is used as a common operand across 16 Vector MACs, where each vector MAC accumulates the products into a single word. Single words from 16 banks are then collected together to form a new row of data which is written back to any of the banks. This spatial arrangement is similar to organization inside the PE of Simba and vastly

TABLE I

SYNTHESIZED METRICS FOR LATENCY, ENERGY, AREA, AND ACCURACY ASSOCIATED WITH DIFFERENT MULTIPLIERS FROM [2] FOR 65 NM

Multiplier	Abbrv.	Type	Bit Width	Bit Arrangment	Latency (ps)	Energy (pJ)	Area (um^2)	LeNet Acc.	ResNet Acc.
IEEE 754 Multiplier(32,8)	FP32	Floating Point	32	(N,es)	1299	22.50	13830.84	98.49%	82.21%
IEEE 754 Multiplier(16,5)	FP16	Floating Point	16	(N,es)	979	4.55	3095.28	98.49%	82.02%
IEEE 754 Multiplier(8,5)	FP8	Floating Point	8	(N,es)	334	0.22	362.88	93.28%	23.54%
Array Multiplier	AM16	Integer	16	-	1780	8.81	6015.24	98.55%	53.18%

reduces reads and writes. Further, the MAC units in AiM are manufactured using a DRAM process and are, therefore, less energy efficient and slower than CMOS chips.

Tetris tries to combine the advantage of both Simba and Aim by stacking DRAM on top of traditional logic chip and connecting them using Through-Silicon-Vias (TSVs), which provide high bandwidth and low energy for data transfer. The logic chip is, however, limited in area and TDP due to stacking. In Tetris, the logic layer has 3136 MAC units spread across 16 vaults operating at 500 MHz for limiting power.

III. EXPERIMENTAL SETUP AND RESULTS

We extend and use Timeloop [7] based infrastructure to model the three SOTA architectures, in great detail, with hardware measured values [4]–[6]. We use a 45nm model, and obtain energy and area values for buffers from CACTI. The architectures are scaled for 8, 16, and 32-bit computation. For Simba and Tetris, the available memory bandwidth from DRAM and the operational frequency is kept same as that of original work during scaling. However, for AiM, the array width is also scaled according to the word's width while keeping the count of multipliers constant. The workloads are AlexNet (AN), MobileNet (MN), ResNet (RN), VGG (VN), DLRM (DN), BERT (BN), LSTM (LM). Further, the results show the average layer metric, separated by CNN and FCL.

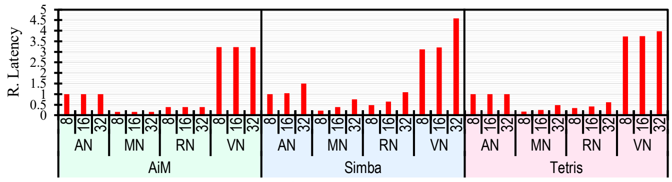

Fig. 2. CNN: Relative latency of each SOTA architecture compared to 8-bit implementation on AN. On average, Simba slows down by 23% and 94% for each doubling of word size, whereas for Tetris, the slow down is only 16% and 76%. Due to architectural construction, AiM does not suffer from slow down for increase in word size.

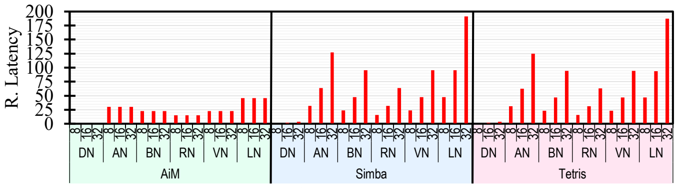

Fig. 3. FCL: Relative latency of each SOTA architecture compared to 8-bit implementation on DN. The latency of AiM does not increase, similar to CNN. For Simba and Tetris, due to a change in effective memory bandwidth, the latency becomes 2× and 4× for each doubling of word size.

Latency: From Fig. 2 and Fig. 3, changing word size does not affect AiM architecture as word size and row width is co-related. For Simba and Tetris, an increase in word size results in a slowdown as effective bandwidth to DRAM reduces. Nevertheless, Tetris scales better than Simba for CNN. Simba and Tetris degrade equally for FCL, as data reuse in FCL is negligible.

Energy: From Fig. 4 and Fig. 5, AiM consumes 4-6× energy for inferencing CNN than Simba and Tetris. However, for FCL,

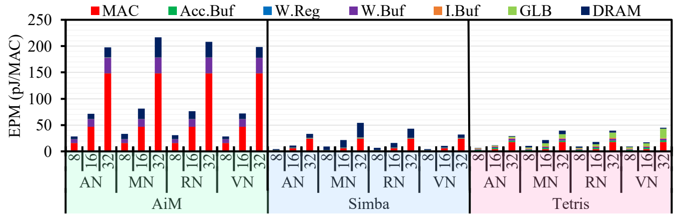

Fig. 4. CNN: Energy per MAC (EPM) shows the amortized energy spent per computation at each component (pJ/MAC Op). For AiM and Simba, the energy increases 2.5× and 6.5× for each doubling, whereas in Tetris, the energy roughly doubles for each doubling of word size. The trends are due to the quadratic increase in energy for MACs operating with high frequency (AiM and Simba) while Tetris is operating at low frequency resulting in a near linear increase - similar to memory access energy.

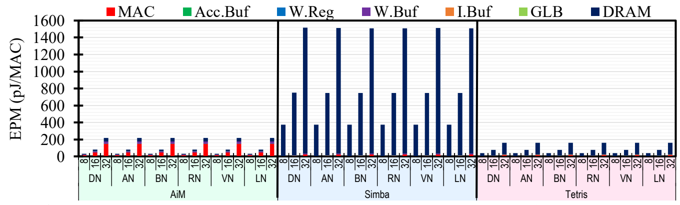

Fig. 5. FCL: EPM shows that energy increases to 2× and 4× for each doubling of word size for Simba and Tetris as transfer energies dominate whereas energy increases to 2.5× and 6× for AiM as compute energy dominates. However, the AiM and Tetris energies are comparable, with Tetris being slightly lower

Simba consumes 6-9× energy than AiM and Tetris. As a result, 8-bit AiM consumes more energy than 32-bit Simba for CNN, while 32-bit AiM consumes less energy than 8-bit Simba for FCL. Tetris scales better for both CNN and FCL in terms of energy.

IV. CONCLUSION

We model and observe the impact of quantization at 8, 16, and 32 bits for SOTA designs from digital DNN accelerator paradigms. We observe that AiM architecture does not slow down due to scaling word size. However, we observe that 32-bit Simba architecture uses similar energy to 8-bit AiM for CNN inference, whereas 32-bit AiM architecture uses less energy than 8-bit Simba for FCL inference.

REFERENCES

[1] Y. Chen *et al.*, "A survey of accelerator architectures for deep neural networks," *Engineering*, vol. 6, no. 3, pp. 264–274, 2020.

[2] T. Glint *et al.*, "Hardware-software codesign of dnn accelerators using approximate posit multipliers," in *ASP-DAC*, 2023.

[3] M. He *et al.*, "Newton: A dram-maker's accelerator-in-memory (aim) architecture for machine learning," in *MICRO*, 2020, pp. 372–385.

[4] B. Zimmer *et al.*, "A 0.32–128 tops, scalable multi-chip-module-based deep neural network inference accelerator with ground-referenced signaling in 16 nm," *JSSC*, 2020.

[5] S. Lee *et al.*, "A 1ynm 1.25v 8gb, 16gb/s/pin gddr6-based accelerator-in-memory supporting 1tflops mac operation and various activation functions for deep-learning applications," in *ISSCC*, vol. 65, 2022, pp. 1–3.

[6] M. Gao *et al.*, "Tetris: Scalable and efficient neural network acceleration with 3d memory," in *ASPLOS*, 2017, pp. 751–764.

[7] A. Parashar *et al.*, "Timeloop: A systematic approach to dnn accelerator evaluation," in *ISPASS*. IEEE, 2019, pp. 304–315.