VSQ Accelerator

Team 20, Samuel Bruce and and Nicholas Dow

Problem and Motivation

The wide-scale adoption of machine learning algorithms in the form of Dense Neural Networks (DNNs) across various domains has led to demand for computational resources skyrocketing. Recently, transformer based architectures have become increasingly common for everyday tasks from natural language processing to CV. Transformer architectures have deterministic structure and require large amounts of computation, making them a target for specialized hardware which can improve performance for training and inference. One approach used to improve the efficiency of DNNs, in both space and computational complexity, is to reduce the representation size of the data in the model, called quantization. Quantization, however, can introduce rounding errors that impact the performance of models, and have been shown to impact transformer models more severely than CNNs.

We analyze a quantization based DNN accelerator and assess it's performance across several parameters. The accelerator impressively achieves **38.7 TOPS/W** at **0.67V** with only a **0.7%** loss of accuracy on the BERT-Base dataset. The accelerator achieves such low loss of accuracy and high throughput by using per-vector scaled quantization (VSQ), which employs an independent scale factor for each 64 element vector. These scale factors allow quantization down to 4-bit data representation with high accuracy. We begin by modeling this accelerator, replicating energy and area measurements documented in the paper sing a simulated CiMLoop environment. Then, using our software model, we investigate using the accelerator as a PE in a larger system and find an optimal system layout. Finally, we examine the effects of changing the technology used in the accelerator from the given 5nm technology to 40nm and 65nm components.

We made several insights in our analysis of the accelerator. Firstly, the workload shape greatly affects the buffer utilization in CiMLoop, especially for the A-Buffer. Smaller values for matrix **A**'s row-size compared to the row-size of matrix **B** can limit the usage of the A-Buffer space given the availability of the accumulator. This becomes a focus of the system architecture, utilitzing the accelerator as a PE. We found that, for this system, and 8×2 array of PE's along with a global PE with a 65,565-bit SRAM memory optimizes the workload for large, square matrices. Finally, we found that under these optimal conditions the amortized cost of the system architecture is relatively small, with the majority of the energy consumption and area usage of the system being taken by the accelerator PEs.

Related Work

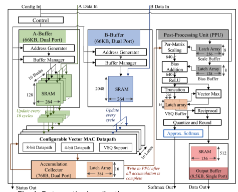
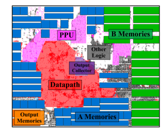


Fig. 1: Data-path visualization



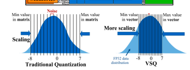


Fig. 2 (top): Die Layout, Fig 3. (bottom) VSQ Advantage

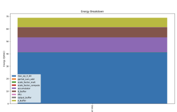
The primary work used for the basis of our model was accelerator architecture described in Keller et. al. [1]. The paper describes a unique hardware approach to achieving high energy efficiency and lower chip area footprint via smart quantization strategies and clever structuring of the memory hierarchy for high temporal and spatial reuse of data seen in Fig. 1.

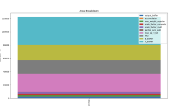
Their VSQ strategy involved having a scaling factor per vector of 64 4-bit values, allowing for the quantization of the original input and weight values that introduces less noise, as the range of values that the quantization had to compress was much smaller than that scaling the entire matrix, as demonstrated by Fig. 3. By having the factor be per-vector, additional hardware is needed to compute and apply the scaling during MACs, but this overhead is amortized over the entire vector and needs only an extra 8-bits of data per vector. The lower energy cost of 4-bit operations enabled by this quantization strategy increases efficiency drastically.

The PE datapath design also leverages data reuse strategies to achieve it's efficiency. The accelerator is innately parallel over the vector width and the 16 vector lanes. The lowest two loops of the loop nest are input stationary over the tile of A, and output stationary over the inner tiling loop. Each bank of A is equipped with a register that is written to every 16 MACs for high temporal reuse, and B is broadcasted across the vector lanes for high spatial reuse.

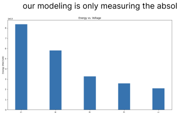
The final key part of the accelerator is specialized hardware for Transformer workloads, such as ReLU, Bias addition, Truncation, softmax, and an addition per-matrix quantization scaling factor for all data-paths.

Accelerator Modeling

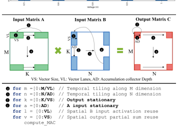


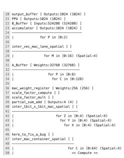


We modeled the area and energy of the accelerator to match the provided data as closely as possible. Note that the area here is less than the die plot size of ~150,000 um^2. This is because the die plot in Fig. 2 has unused area and is concerned with interconnects, while our modeling is only measuring the absolute space needed by the components.



Left here is the modeled efficiency in MACs/Watt as voltage is varied. Notice how, as voltage increases, efficiency decreases. While higher voltages improve OPS/S, a low voltage of 0.5V allows for over 80 TOPS/W.





Here is the mapping for Matrix Multiplication onto the accelerator. Left is the general tiling strategy, while on the right is the mapping found by CiMLoop in the software model. Note that the two mappings agree, and all buffers are full except for the accumulator. This is due to the larger representation size in the accumulator, and it is in fact full.

System Integration

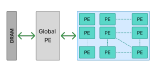
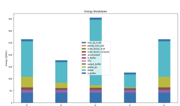
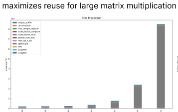
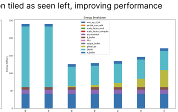


Fig. 4: PEs tiled within system layout

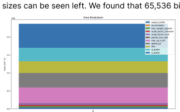


We put the accelerator into a system architecture, using it as the PE. We followed the general design on the left, with a Global PE in charge of small layer ops and a shared scratchpad. For the spatial PE array we did a parameter sweep and found (8,2) to be the best layout. We believe this is because large X fanout in the Input/Output shared dataspace maximizes reuse for large matrix multiplication tiled as seen left, improving performance



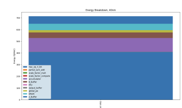


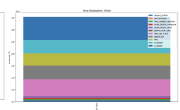
We also optimized scratchpad size for large, square MM. The graph on the right shows that increasing the size is beneficial until a critical point where cache thrashing is solved, then it has negative benefits as larger memories are less efficient. The space used for different sizes can be seen left. We found that 65,536 bits was the optimal size



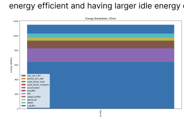
Here is the area of the final configuration, (8,2) spatial PEs and a 65,536 bit scratchpad. Note that the area is roughly 16x that of the single PE without the global PE, and just over 2,000,000 um^2 total. The energy of the final system can be found in the graph at the top of this section

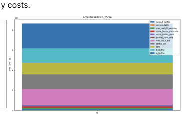
Technology Scaling





This is the model of our optimal accelerator (see left) technology scaled to 40nm components. Note that while area obviously scales when changing from 5nm to 40nm technology, the energy also increases dramatically due to the larger components being less energy efficient and having larger idle energy costs.





Here we further scale the model to use 65nm components. Again both the energy per computation and the area significantly increase. Note as well that for both 40nm and 65nm that, while energy and area increased, it did so uniformly. This is evidence of the way in which CiMLoop performs technology scaling approximations across it's components, and we might have gotten slightly different results if we explicitly modeled each component used in the different technology sizes.

References

1. Ben Keller, Rangharajan Venkatesan, Steve Dai, Stephen G. Tell, Brian Zimmer, Charbel Sakr, William J. Dally, C. Thomas Gray, and Brucek Khailany, "A 95.6-TOPS/W Deep Learning Inference Accelerator With Per-Vector Scaled 4-bit Quantization in 5nm," *IEEE Journal of Solid-State Circuits*, vol. 58, no. 4, April 2023.
2. Github Repository with CiMLoop Model: https://github.com/sgbruce/cimloop