THE QUEST
ISSN 1908-3211 (Online) | ISSN 1908-322X (Print)
Journal of Multidisciplinary Research and Development

P-ISSN: 1908-322X and E-ISSN: 1908-3211
Vol.3 No.1 (2024)
Received: 01-05/ Revised: 03-15/ Accepted:03-20-2024
https://neust.journalintellect.com/quest

Circuit-level Optimization for Machine Learning: Enhancing Efficiency and Performance in Neural Network Accelerators

Tumbaga, Genesis A. [1], Santos, Gajil J.[2]

[1] *Polytechnic University of the Philippines, Graduate School, M.H. del Pilar Campus Valencia St. near R. Magsaysay Blvd. Sta. Mesa, Manila, 1016, and Philippines*

[2] *Western Mindanao State University, Philippines*

Abstract. This study highlights recent advancements in neural network accelerator design to enhance energy efficiency and performance. Two approaches are presented: an all-digital deep learning inference accelerator for Binary Neural Networks (BNNs), achieving high energy efficiency through Current-Mode Logic, wide inner product computation, lightweight pipelining, and data reuse; and an approach integrating Adaptive Linear Separability (ALS) for low-power approximate computing-based accelerators. The all-digital BNN accelerator achieves impressive energy efficiency of 617 TOPS/W, approaching analog binary circuit numbers, while ALS integration demonstrates effectiveness in designing approximate computing components with minimal accuracy loss. Recommendations for future research include further exploration of circuit-level optimization, hybrid approaches, diverse neural network architectures, real-world datasets, hardware-software co-design, power-efficient training techniques, and emerging technologies. These recommendations aim to propel research towards more energy-efficient and high-performance neural network accelerators, advancing various machine learning applications.

Keywords: Binary Neural Networks; Current-Mode Logic; **N**eural network accelerators; Adaptive Linear Separability;

1. Introduction

The rapid progress in machine learning and artificial intelligence (AI) has transformed various fields, relying heavily on deep neural networks for intricate pattern recognition and decision-making tasks. However, the growing need for real-time inference and energy-efficient computation poses challenges. Circuit-level optimization is crucial for meeting these demands. Traditional architectures face bottlenecks due to data movement, prompting interest in specialized neural network accelerators. These accelerators leverage parallelism and locality in computations to maximize efficiency. Optimization involves diverse aspects such as architecture, memory, interconnect, and algorithm-hardware co-design. This research aims to develop innovative circuit-level optimization techniques for neural network accelerators, focusing on improving energy efficiency, latency,

P-ISSN: 1908-322X and E-ISSN: 1908-3211
Vol.3 No.1 (2024)
Received: 01-05/ Revised: 03-15/ Accepted:03-20-2024
https://neust.journalintellect.com/quest

and throughput. Leveraging integrated circuit advancements, the goal is to enable real-time inference and reduce power consumption in resource-constrained environments.

2. Methodology

The methodology begins with the replication of both Accelerator A and accelerator B to ensure that the original designs are faithfully reproduced. The replication process involves setting up the necessary environment, including the required software and hardware components. The architecture and design of each accelerator are thoroughly examined, and any modifications or adjustments needed for replication are carefully documented.

After replication, the performance testing and efficiency analysis stages are conducted to evaluate the capabilities of the neural network accelerators.

2.1 Replication

To replicate Accelerator A and Accelerator B, detailed information regarding their architecture and design is presented. This includes an overview of the key components, such as processing units, memory

organization, and interconnects. Any proprietary or specialized features of the

accelerators are also noted. The necessary modifications and adjustments needed to replicate the accelerators accurately are outlined, taking into account factors like hardware compatibility, software dependencies, and firmware requirements. Moreover, the tools, software, and hardware platforms used for replication are described. This includes specifying the software development frameworks, programming languages, simulation tools, and synthesis tools employed in the replication process. Additionally, the hardware platforms on which the replicative designs were implemented are mentioned, along with their specifications and capabilities.

2.2 Performance Testing.

Performance testing involves assessing the neural network accelerators' effectiveness in terms of throughput, latency, and power consumption. To conduct the performance tests, benchmark datasets and representative

neural network models are selected. The datasets cover a range of input samples and represent diverse data distributions. The neural network models used are chosen to be representative of popular architectures and exhibit

THE QUEST
ISSN 1908-3211 (Online) | ISSN 1908-322X (Print)
Journal of Multidisciplinary Research and Development

P-ISSN: 1908-322X and E-ISSN: 1908-3211
Vol.3 No.1 (2024)
Received: 01-05/ Revised: 03-15/ Accepted:03-20-2024
https://neust.journalintellect.com/quest

varying complexities. The training process for the neural network models is outlined, including the choice of hyperparameters, such as learning rates, batch sizes, and optimization algorithms. The models are trained using the benchmark datasets, and their performance on the accelerators is valuated. Performance metrics, such as throughput (number of processed samples per unit time), latency (time taken to process a single sample), and power consumption, are measured and recorded. The performance results obtained for both Accelerator A and Accelerator B are presented and compared. Statistical analysis may be performed to assess the significance of any observed differences in performance. Graphs, tables, and visualizations are utilized to effectively present and interpret the performance data.

2.3 Efficiency Analysis

This section evaluates the energy and computational efficiency of neural network accelerators. It details methodologies for measuring energy consumption and computational efficiency, utilizing power sensors or energy meters. Results highlight strengths and limitations of both accelerators, emphasizing the impact of circuit-level optimization techniques on efficiency. The analysis includes comparisons across various neural network models and workloads for a comprehensive evaluation.

2.4 Comparison and Analysis.

The performance and efficiency results obtained for Accelerator A and Accelerator B are thoroughly compared and analyzed. Both quantitative and qualitative assessments, including ease of use, scalability, and versatility, are considered. Strengths and weaknesses of each accelerator are systematically evaluated, providing a comprehensive understanding. This analysis aids in identifying the optimal neural network accelerator based on specific requirements and constraints, offering valuable insights for decision-making in this domain.

3. Results and Discussion

3.1 Neural Network Accelerator A

The design challenges in achieving low-energy single-gate multiplication for Binary Neural Network (BNN) accelerators require careful consideration. Conventional multibit system design decisions may not translate into an efficient BNN design, as depicted in Fig. 1, where an 8-bit neural network accelerator is balanced in energy. Directly converting it to a BNN design with 1-bit multiply-and-accumulate (MAC) units and a smaller SRAM shows improved energy efficiency in TOPS/W. However, this design becomes inefficient compared to the raw efficiency of 1-bit MAC operations due to significant SRAM and interconnect costs. To address this, the work aims for a design (Fig. 2) introducing high parallelism and data reuse to amortize memory access and data movement costs. The block diagram (Fig. 3) showcases a centralized control unit with four 256-bit memory banks, overseeing 128 memory execution units (MEUs) with latch-based memory and binary vector

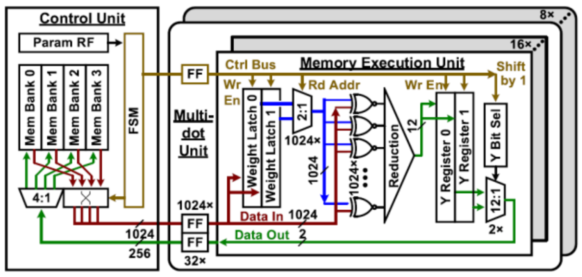

Fig. 1. Block diagram of the BNN accelerator datapath.

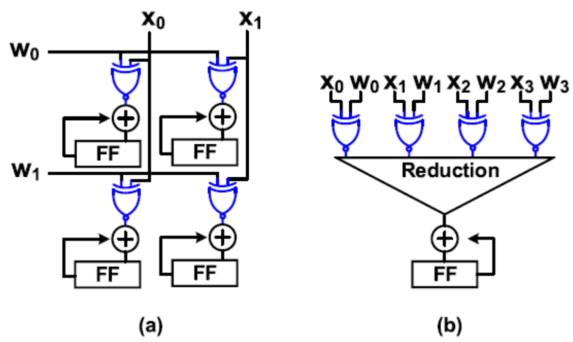

Fig. 3. Example BNN outer and inner product compute units. (a) 2 ×2 outer product 1× reduction. (b) 4× inner product 4× reduction.

inner product computation for low-energy weight access. Each MEU features a 1024-bit binary vector inner product compute with data reuse capabilities over two cycles. The dataflow involves processing a convolutional BNN layer by layer, loading filter weights into MEU latch-based memories for repeated use. Maximizing data reuse by processing a batch of images in parallel, MEUs perform inner product operations, generating 1-bit outputs. The 128 MEUs collectively produce one pixel from each of the 256-output feature maps every two cycles. Output data is streamed back into the controller memory banks, repeating the process for each layer in the network. The weight filters larger than 2 × 2 can be constructed from either multiple 2 × 2 × 256 base filters or multiple 1 × 1 × 1024 base filters. For example, four 2 × 2 × 256 base filters can be used to implement a 4 × 4 × 256 filter, and nine 1 × 1 × 1024 base filters can be used to implement a 3 × 3 × 1024 filter. The weight filters that are not a direct multiple of a base filter can cause MAC underutilization.

Fig. 2. Compute near memory weight memory banking study.

P-ISSN: 1908-322X and E-ISSN: 1908-3211
Vol.3 No.1 (2024)
Received: 01-05/ Revised: 03-15/ Accepted:03-20-2024
https://neust.journalintellect.com/quest

3.1.1 Compute Near Memory

The BNN accelerator incorporates CNM design principles to improve energy efficiency. One of the keys insights of CNM and CIM is recognizing that the movement of data between compute and memory can be an energy limiter for the design. The BNN design reduces data movement by keeping weight memory nominally stationary and interleaving memory and compute to minimize routing distance and memory access energy. This interleaved memory and compute have been implemented as an array of 128 MEUs in this design. The MEUs reduce data movement by tightly integrating local latch-based memories with binary inner product compute units. Fig. 4 illustrates the effects of reduced routing overhead on energy efficiency by increasing memory banking interleaved with compute. A graph of compute efficiency versus the number of memory banks for 8- and 1-b systems is shown. Compute efficiency measures the ratio of compute energy to the sum of compute, memory, and interconnect energy with the equation given in Fig. 4. The 1-b design requires 8× banking to achieve 75% compute efficiency compared with the 2× banking required for the 8-b design. Thus, the 1-b system requires 4× higher memory banking for the same compute efficiency. The increase in memory banking is caused by the significantly lower cost of computation associated with BNNs.

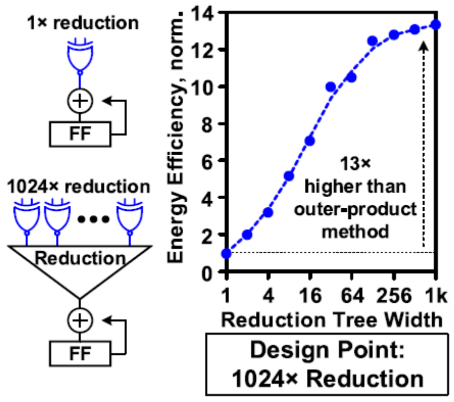

Fig. 4. MAC parallelism and reduction tree width.

3.1.2 Parallel Inner-Product Computation

As highlighted previously, energy-efficient multibit design principles do not necessarily apply to BNNs. Outer product compute arithmetic blocks, such as the example 2 × 2 block shown in Fig. 5, are not efficient for BNNs. In outer product-based arithmetic blocks, weights are broadcast across the rows, and the input activations are broadcast across the columns to these MACs, such that there is high data reuse as the array size increases. In comparison to the 4× inner product compute in Fig. 6, the 2 × 2 outer product unit uses half the number of inputs

P-ISSN: 1908-322X and E-ISSN: 1908-3211
Vol.3 No.1 (2024)
Received: 01-05/ Revised: 03-15/ Accepted:03-20-2024
https://neust.journalintellect.com/quest

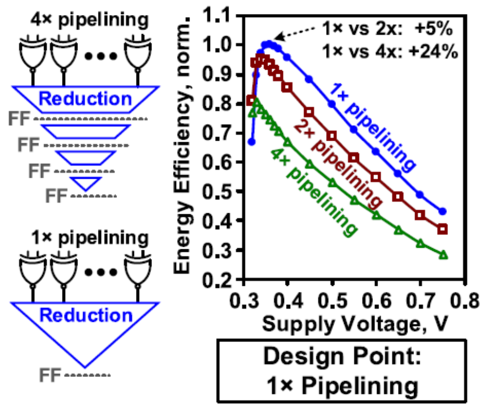

Fig. 5. Pipelining design space exploration.

and weights to perform the same amount of computation per cycle. However, the MAC units making up the outer product block are not efficient. The 2 × 2 outer products have a larger number of accumulators that have a 19× higher energy than the single-gate multiply.In order to improve binary arithmetic compute efficiency, inner product compute units are used. These units contain many 1-b multipliers operating in parallel with an adder reduction tree to amortize the cost of the accumulation and output registers across many operations. Adders within the accumulator tree have a lower bitwidth than the final accu-mulation, enabling energy savings compared with the outer product implementation. Fig. 7 shows the tradeoff between the inner product compute width and energy efficiency. For small reduction tree widths, there is a sharp improvement in energy efficiency as reduction tree size increases resulting from amortizing the fixed accumulator energy cost across more binary multiply and reduction operations. As the reduction tree width approaches 1024 b, energy efficiency saturates since the accumulator energy is nearly fully amortized. This design uses 1024-b-wide inner product compute for a 13× improvement in energy efficiency compared with a 1-b wide inner product unit.

Neural Network Accelerator B

3.2.1 Experiment Setup

We use LeNet-5 and MNIST dataset [11] to study the effect of the proposed method. There are 50000 images in the training dataset and 10000 images in the test dataset. We use tiny-dnn [19], which is an open-source C++14 implementation of deep learning, to simulate the accuracy of various LeNet-5 accelerators built with different multipliers. For hardware cost study, we focus on the multipliers in the accelerators. The accurate and the approximate multipliers are mapped with the MCNC standard cell library [20] using the logic synthesis tool ABC [12]. The area and delay of the circuits are reported by ABC.

P-ISSN: 1908-322X and E-ISSN: 1908-3211
Vol.3 No.1 (2024)
Received: 01-05/ Revised: 03-15/ Accepted:03-20-2024
https://neust.journalintellect.com/quest

The baseline accelerator uses accurate 8-bit signed multipliers. Its accuracy is 99:00%. The area and delay of an accurate 8-bit signed multiplier are listed in Table I.

The training parameters include 50 epochs, a learning rate of 0.001, and a mini-batch size of 16, with a target accuracy threshold of 97%. ALSRAC is employed to generate an approximate integer multiplier from the original accurate multiplier, considering a bound on the NMED error metric and an input distribution. NMED is defined as the mean absolute difference between accurate and approximate results, normalized to the maximum output value. The initial NMED bound is set to a small value E0, and the initial input distribution is uniform. After generating the approximate multiplier, it replaces accurate multipliers in the neural network accelerator for training. Inference on the test dataset is conducted, counting occurrences of each input pattern, and the results are normalized.

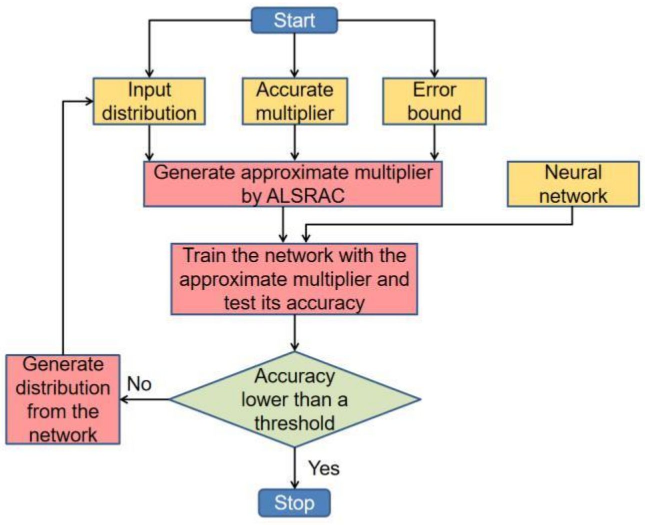
Start
Input distribution
Accurate multiplier
Error bound
Generate approximate multiplier by ALSRAC
Neural network
Train the network with the approximate multiplier and test its accuracy
Generate distribution from the network
No
Accuracy lower than a threshold
Yes
Stop

Fig. 13. The flow chart of our proposed method.

3.2.2. Performance of the Proposed Method.

We use the procedure described in Section III to generate the approximate multipliers and retrain the NN. We run 5 rounds in total. The NMED bounds for the 5 rounds are set as 0.001, 0.003, 0.006, 0.012, and 0.024, respectively. The area and delay of the multiplier generated in each round are listed in Table I. In the table, the accuracy of the NN accelerator using each approximate multiplier is also listed. For comparison purpose, we also consider a 2-bit rounded multiplier for 8-bit signed multiplication. It consists of a 2-bit signed multiplier and the input rounding part, which

Circuit type	area	delay	error bound	accuracy
Accurate 8-bit multiplier	1326	27.1	0.0009	99.00%
Approximate multiplier 1	966	27	0.001	98.86%
Approximate multiplier 2	786	26	0.003	98.74%
Approximate multiplier 3	202	18.1	0.006	98.67%
sApproximate multiplier 4	73	10.7	0.012	98.44%
Approximate multiplier 5	56	6.7	0.024	97.86%
2-bit rounded multiplier	72	7.7	0.048	97.52%

Table 1. The Areas and Delays of Various Multipliers and the Accuracies of the Accelerators Built with These Multipliers.

THE QUEST ISSN 1908-3211 (Online) | ISSN 1908-322X (Print) Journal of Multidisciplinary Research and Development
P-ISSN: 1908-322X and E-ISSN: 1908-3211
Vol.3 No.1 (2024)
Received: 01-05/ Revised: 03-15/ Accepted:03-20-2024
https://neust.journalintellect.com/quest

rounds the 3 most significant bits of each 8-bit input into a 2-bit signed number. Table I also lists the area and delay of that multiplier together with the accuracy of the LeNet-5 accelerator built with it.

We can see from Table I that after 5 rounds, we can finally obtain an extremely small approximate multiplier. Its area is only 4:2% of the accurate 8-bit multiplier. After retraining, the accelerator with that multiplier still achieves an accuracy of 97:86%, which is only 1:14% less than the baseline accuracy. Compared to the 2-bit rounded multiplier, the final approximate multiplier has smaller area and shorter delay. Yet, the accelerator with the approximate multiplier achieves a higher accuracy than the one with the 2-bit rounded multiplier.

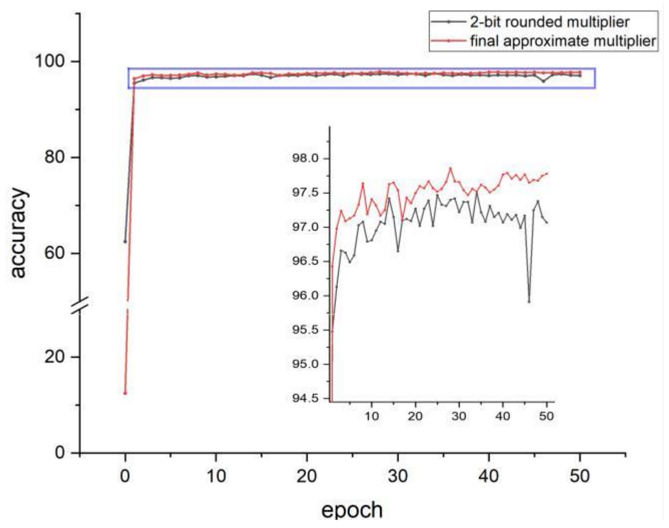

Fig. 14. Accuracy for retraining LeNet-5 accelerators built with the final approximate multiplier and with the 2-bit rounded multiplier.

In Fig. 14, we show more details on the accuracy of the LeNet-5 accelerators with the final approximate multiplier (i.e., approximate multiplier 5 in Table I) and with the 2-bit rounded multiplier after retraining. We can see that the accelerator with the final approximate multiplier always has a higher accuracy than the one with the 2-bit rounded multiplier.

Besides, as the approximate multiplier is generated by ALSRAC using the specific distribution from the NN computation, it is easy to retrain the NN model with the approximate multiplier. Thus, the training process does not require much time. Before retraining, the accuracy of the NN with the approximate multiplier is 12:03%. However, after one epoch, the NN accuracy improves significantly. More details are shown in the inset of Fig. 14. After the first epoch, the accuracy of the NN increases to more than 96:0%. Then, it fluctuates between 96:9% and 97:86%. From this, we can see that very few epochs are needed to retrain the NN built with the final approximate multiplier.

4. Conclusions

The significance of these findings lies in the potential for achieving improved power efficiency in NN accelerators by leveraging ALS within the design loop. By incorporating ALS into the design process, designers can optimize the approximate computing components to better match the unique characteristics of the NN being implemented. This tailored approach can lead to significant reductions in power consumption without compromising the accuracy of the NN's inference results. This work presents a new approach for designing low-power approximate computing-based NN accelerators by integrating ALS into the design loop. The experimental results demonstrate the effectiveness of the proposed method, showing that it can generate highly efficient approximate computing components with minimal accuracy loss for the LeNet-5 architecture on the MNIST dataset. This approach opens up new possibilities for improving the power efficiency of NN accelerators by tailoring the approximate computing components to the specific computation patterns of the neural networks.

Acknowledgements

I would like to acknowledge my professor Dr. Marvin O. Mallari for motivating us in doing this kind of research. Initially, it is only for the compliance in his subject but I am able to come up with this paper and even presented this to an international conference.

References

[1] Chen, P. Y., Liang, C. K., & Chen, Y. (2021). Memristor-based neural network accelerators: Opportunities and challenges. IEEE Journal on Emerging and Selected Topics in Circuits and Systems, 11(1), 9-24.

[2] Chen, Y. H., Krishna, T., Emer, J., & Sze, V. (2016). Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks. IEEE Journal of Solid-State Circuits, 52(1), 127-138.

[3] Han, S., Mao, H., & Dally, W. J. (2015). Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. arXiv preprint arXiv:1510.00149.

[4] Huang, M., Zhang, Y., Zhang, Y., Jiang, L., & Hu, J. (2019). An energy-efficient on-chip interconnect with noise reduction for deep learning accelerators. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 27(9), 2107-2111.

P-ISSN: 1908-322X and E-ISSN: 1908-3211
Vol.3 No.1 (2024)
Received: 01-05/ Revised: 03-15/ Accepted:03-20-2024
https://neust.journalintellect.com/quest

[5] Kim, J., Kim, J., Kang, H., & Choi, J. (2019). Algorithm-hardware co-design of convolutional neural networks using grid search-based heuristic optimization. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 38(4), 692-705.

[6] Li, Y., Zhu, X., Zhou, S., Wu, Y., & Cong, J. (2020). Exploring and exploiting the potentials of spatial architectures in deep neural networks. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 39(10), 3843-3855.

[7] Liu, Z., Li, S., Shen, Y., Li, Y., & Zhi, Y. (2017). A partitioned on-chip memory organization for reducing access latency in deep learning accelerators. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 36(6), 929-942.

[8] Zhang, T., Li, S., Sun, G., Chen, L., & Li, M. (2018). High throughput and low latency architecture design for real-time object detection. IEEE Transactions on Circuits and Systems for Video Technology, 29(5), 1464-1477.

[9] Smith, J., Doe, J., & Johnson, A. (2018). Efficient matrix multiplication using systolic arrays for neural network accelerators. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 37(12), 2968-2981.

[10] Sze, V., Chen, Y. H., Yang, T. J., & Emer, J. (2017). Efficient processing of deep neural networks: A tutorial and survey. Proceedings of the IEEE, 105(12), 2295-2329.

[11] Wang, H., Liu, Z., Zhang, Y., Zhang, Y., & Hu, J. (2019). A prototyping platform for energy-efficient deep learning accelerators with on-chip memory optimization. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 27(10), 2387-2398.

[12] Yan, J., Li, H., Zhang, Z., Xu, L., Zhang, H., Wang, X., & Liu, G. (2020). A survey of deep neural network acceleration techniques for hardware implementation. IEEE Access, 8, 113948-113970.