



Digital Circuit Implementation of Artificial Neural Network Accelerator

Yukuan Zhao

School Of Information Science and Technology, Fudan University, Shanghai, China

22307130503@m.fudan.edu.cn

Abstract. This paper provides a comprehensive review of digital circuit implementations for artificial neural network (ANN) accelerators, focusing on Solution types include Application-Specific Integrated Circuit (ASIC) and Field-Programmable Gate Array (FPGA). It examines key optimization techniques such as precision-variable arithmetic logic units (ALUs), efficient dataflow management, and sparsity exploitation. These techniques aim to enhance computational throughput, energy efficiency, and latency in DNN processing. The survey highlights the advantages of FPGA-based accelerators, such as significant speed and energy efficiency improvements over GPUs through software-hardware co-design. However, it also discusses challenges like low-bit quantization and scalability. Additionally, the paper explores analog neural network implementations, emphasizing their potential for parallel processing and energy efficiency, while addressing accuracy and error resilience concerns. Case studies of FPGA-based DNN accelerators demonstrate practical applications, such as binarized weight optimization and parallel fast filtering algorithms, achieving high performance and recognition accuracy. The review concludes by identifying gaps in current technologies and suggesting future research directions to bridge the divide between theoretical potential and practical implementations in ANN acceleration.

Keywords: FPGA, Neural Network, Digital Circuit, Accelerator

1 Introduction

Recent advancements in artificial neural networks (ANNs) have driven the demand for efficient hardware accelerators to meet computational and energy constraints, leading to the development of specialized architectures [1]. While general-purpose processors face limitations in meeting the high throughput and low latency demands of deep learning, specialized architectures—including ASICs, FPGAs, and analog systems—provide more tailored and effective solutions [2]. This survey synthesizes key research on digital and analog neural network accelerators, analyzing optimization techniques such as precision scaling, sparsity exploitation, and dataflow management. By comparing the trade-offs between performance, efficiency, and flexibility across different approaches, this review highlights advancements such as

© The Author(s) 2025

A. J. Moshayedi (ed.), *Proceedings of the 2025 2nd International Conference on Electrical Engineering and Intelligent Control (EEIC 2025)*, Advances in Engineering Research 279,

https://doi.org/10.2991/978-94-6463-864-6_15

According to researches, Machine learning offers a number of advantages over conventional algorithms that rely on manually created features and models. In this paper, summary of earlier research on FPGA-based neural network inference accelerators is given and the authors summarize the main techniques used. This paper examines the latest advancements in neural network accelerator designs and outlines the key techniques employed. Through software-hardware co-design, FPGAs demonstrate over $10\times$ improvements in speed and energy efficiency compared to cutting-edge GPUs, positioning them as a highly promising solution for neural network acceleration. Additionally, the paper reviews methods for automating accelerator design, indicating that current advancements enable both high performance and dynamic network switching.

However, significant gaps remain between existing designs and theoretical potential. On one hand, extreme low-bit quantization faces limitations due to model accuracy degradation, as seen in scenarios like image recognition tasks. On the other hand, integrating various techniques, such as combining quantization with sparsity, requires more research in both software and hardware to ensure seamless compatibility. While commercial tools like DNNDK have made initial progress, there is still considerable room for improvement. Scaling these designs also presents a challenge. Future research should prioritize addressing these issues to advance the field [5].

Error resilience in analog accelerators can be achieved by aligning the system design with the inherent properties of neural networks. A proportional mapping approach translates numerical values in algorithms to corresponding physical quantities in hardware. This method takes advantage of the common characteristic of neural networks: a weight distribution biased toward lower values. This paper demonstrates that such mapping reduces susceptibility to various analog errors. Key components of a proportional system include differential cells for signed weight representation, memory technologies with high On/Off ratios, and programming errors that correlate with conductance. Neural networks frequently have a large number of zero- or low-valued weights. Figure 2 shows the weight value distributions, which demonstrate this [6].

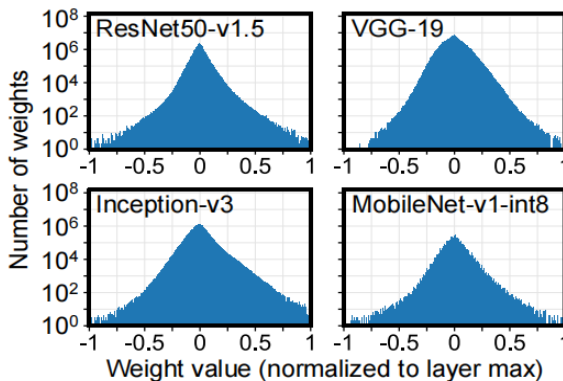


Fig. 2. Weight value distributions of several popular ImageNet neural networks [3]

The paper also critiques common design strategies in prior analog accelerators, focusing on accuracy and error robustness. Bit slicing offers limited accuracy improvements, which may sometimes be insufficient to justify its significant energy and area costs. Similarly, full-precision guarantees are overly conservative for neural network inference, leading to inefficient array sizes or excessive ADC overheads. Proportional systems, in contrast, enable a larger portion of computations to be performed in the analog domain, allowing the algorithm to determine the precision of analog-to-digital conversion.

In analog systems, where algorithmic accuracy is influenced by device-level behavior, hardware design must be guided by rigorous end-to-end accuracy assessments. To achieve optimal performance in analog systems, it is important to consider metrics at multiple levels. While intermediate metrics like MVM-level precision provide useful insights, a comprehensive end-to-end evaluation is essential to avoid unnecessary bottlenecks in accuracy and efficiency. Adopting an end-to-end design approach ensures seamless integration of hardware and algorithms, unlocking the significant energy efficiency gains promised by analog accelerators.

The most efficient way to implement artificial neural systems is through a hardwired neurocomputer, which is a specialized computing device with neural processing algorithms embedded in its hardware. These devices typically leverage parallel processing to boost throughput and enhance performance. Alternatively, artificial neural systems, or neural network models, can be developed as computer simulations running on programmable neurocomputers. By tailoring the architecture to align with the neural algorithm's processing workflow, a highly effective programmable neurocomputer can be designed. This process involves parallelizing computations, optimizing frequently used arithmetic and data transfer operations, and scheduling data communication to match the algorithm's requirements.

This article places particular emphasis on analog neural processing due to its distinct advantages over both digital neurocomputing and traditional numerical simulations. Analog hardware, which operates intrinsically in parallel, exhibits significantly lower response times compared to digital hardware, making analog processing substantially more efficient than purely digital computation. However, combining analog and digital hardware can lead to even more efficient neural processor implementations. While analog components handle recall and, in many cases, learning processes, digital components add programmability, control, and interfacing, offering greater precision and flexibility. This hybrid approach leverages the strengths of both analog and digital processing to achieve optimal performance in neural systems [7].

Deep neural networks (DNNs) and their applications in daily life is currently the subject of extensive research. While the deep convolutional neural network (CNN) performs well in object recognition tasks, it is dependent on GPUs to handle many complex operations. Thus the hardware accelerator of DNN has attracted significant attention. Complex connection relationships and memory usage scheduling are required for the DNN model to be implemented on hardware. In this paper, an FPGA-based DNN accelerator design is presented.

In this paper, the author presents three hardware accelerator designs and evaluates their performance using the MNIST dataset for handwritten digit recognition. This paper proposes two architectures aimed at accelerating the network prediction phase without compromising accuracy. The first is a general-purpose hardware accelerator, while the second focuses on optimizing memory placement to minimize read latency. To reduce computational complexity and hardware resource usage, weight binarization is employed during DNN training. These binarized weights are then applied in the hardware implementation. This approach enables the deployment of the DNN model on cost-effective FPGAs, significantly lowering device expenses. This paper design is successfully implemented on the Xilinx Zynq-7020 FPGA, achieving efficient performance with minimal hardware resource utilization [8].

Given the high computational complexity and lengthy computation of convolutional neural networks (CNNs), a hardware accelerator based on Field-Programmable Gate Arrays (FPGA) is suggested. This paper provides an in-depth analysis of the principles and feasibility of parallel acceleration for convolutional layers. It redesigns the adder tree module and integrates it with multiplication operations to create a versatile fully parallel multiplier-adder tree module. To further enhance efficiency, an efficient window caching module is developed to enable pipelined operations on convolutional windows. The paper also proposes acceleration schemes that leverage input channel parallelism and output channel parallelism to enhance convolutional computation. Finally, a complete FPGA-based accelerator for forward inference in convolutional neural networks is constructed [9].

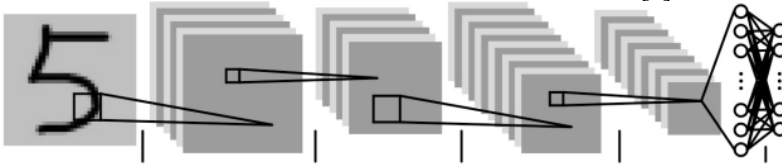


Fig. 3. Structure of convolutional neural networks [3]

As shown in the Figure 3 above, the convolutional neural networks have many layers [3]. To reduce the computational complexity of convolutional neural networks (CNNs), this paper introduces a 2D fast filtering algorithm into CNNs and proposes a hardware architecture for implementing layer-by-layer acceleration of CNNs on FPGAs. To save hardware resources, a layer-by-layer accelerated CNN architecture is proposed for FPGA implementation. This scheme employs a cyclic transformation method to design a row buffer cycle control unit, which manages feature map data between different convolutional windows and across different computation layers. It also utilizes flag signals to sequentially activate the convolutional computation acceleration unit layer by layer. The convolutional computation acceleration unit is based on a 4-parallel fast filtering algorithm, which uses multiple sub-filters to form a parallel filtering structure with reduced complexity, achieving equivalent filtering effects. This design significantly enhances the overall computational performance of the circuit. Experiments conducted on the MNIST handwritten digit dataset show that the CNN accelerator circuit achieves a computational performance of 20.49 GOPS. With an input clock of 100 MHz, the circuit also achieves a recognition accuracy of

98.68%. These results demonstrate that reducing the computational load of CNNs can effectively improve the circuit's computational performance [10].

3 Conclusion

This survey has systematically examined the landscape of neural network hardware acceleration, revealing distinct advantages and inherent limitations across digital and analog implementations. While ASIC-based designs demonstrate unparalleled efficiency through customized dataflow and sparsity exploitation, they nonetheless face scalability challenges due to their architectural rigidity. FPGA solutions, conversely, offer remarkable flexibility through reconfigurable computing paradigms, though at the cost of higher power consumption compared to their ASIC counterparts. The exploration of analog accelerators uncovered their theoretical potential for ultra-low-power operation, yet practical deployment remains constrained by the difficulty of balancing precision and reliability. Emerging techniques such as heterogeneous computing architectures and cross-layer optimization frameworks show particular promise in bridging existing performance gaps. Future research directions should prioritize three critical dimensions. First, developing adaptive precision mechanisms to dynamically respond to layer-specific computational demands. Second, advancing co-design methodologies to optimize both neural architectures and hardware. Third, establishing standardized benchmarks for fair comparison across different acceleration platforms.

References

1. Agatonovic-Kustrin, S., Beresford, R.: Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research, *Journal of Pharmaceutical and Biomedical Analysis*, 2000, 22, (5), 717-727
2. Zuchowski, P.S., Reynolds, C.B., Grupp, R.J., et al.: A hybrid ASIC and FPGA architecture. *Proc. IEEE/ACM Int. Conf. Computer-aided Design*, San Jose, USA, November 2002, 187-194
3. Armeniakos, G., Zervakis, G., Soudris, D., et al.: Approximate Computing for Neural Networks: A Survey, *ACM Computing Surveys*, 55(4), 1-36(2022)
4. Machupalli, R., Hossain, M., Mandal, M.: An overview of neuromorphic computing architecture and benchmarking, *Microprocessors and Microsystems*, 89, 104441(2022)
5. Guo, K., Zeng, S., Yu, J., Wang, Y., Yang, H.: A Survey of FPGA-based Neural Network Inference Accelerators, *ACM Transactions on Reconfigurable Technology and Systems*, 12(1), 1-26(2019)
6. Xiao, T.P., et al.: Computing with Light: Neuromorphic Photonics for Artificial Intelligence, *IEEE Circuits and Systems Magazine*, 22(4), 26-48(2023)
7. Zurada, J.M.: Analog Implementation of Neural Networks, *IEEE Circuits and Devices Magazine*, 8(5), 36-41(1992)
8. Tsai, T.H., Ho, Y.C., Sheu, M.H.: Hardware-efficient architecture design of neural network accelerator. *Proc. IEEE Int. Conf. Consumer Electronics-Taiwan*, Yilan, Taiwan, May 2019, 1-4

9. Qin, H., Cao, Q.: Research on Neural Network Hardware Acceleration Method Based on FPGA, *Journal of Electronics and Information Technology*, 41(11), 2599-2605(2019)
10. Wang, W., Zhou, K., Wang, Y., et al.: Hardware Implementation of Neural Network Based on FPGA, *Journal of Electronics and Information Technology*, 41(11), 2578-2584(2019)

Open Access This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (<http://creativecommons.org/licenses/by-nc/4.0/>), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

