Proceedings of the 5th International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/121/2025.19490

Hardware Implementation of Convolutional Neural Networks

Yulin Guo[1,a,*]

[1]*Detroit Green Technology Institute, HuBei University of Technology, WuHan, 430070, China*
a. g1812514770@outlook.com
**corresponding author*

Abstract: Convolutional Neural Networks (CNNs) have broad application prospects in computer vision, image processing, and other fields. However, their characteristics, such as high computational speed and large data volume, pose significant challenges for hardware implementation. This project aims to improve computational efficiency and reduce energy consumption by summarizing hardware-based convolutional neural network algorithms to meet real-time requirements. Additionally, it will summarize research methods for accelerating convolutional computation, pooling, and fully connected layers using different hardware platforms such as ASIC, FPGA, and GPU. This paper focuses on optimizing data flow, parallel processing, and memory architecture to reduce computation latency and energy consumption. To ensure network accuracy, methods such as quantization and pruning are employed to reduce model size and computational complexity. Literature indicates that custom hardware designs for convolutional neural networks significantly enhance performance and energy efficiency compared to traditional software implementations. This review aims to provide an efficient hardware acceleration method for practical deep learning algorithms and promote development in fields such as smart terminals and edge computing.

Keywords: Convolutional Neural Networks, Hardware Implementation, FPGA, GPU.

1. Introduction

Convolutional Neural Networks (CNNs) are an emerging machine learning method with significant application prospects in image processing and computer vision. With the rapid development of deep learning technology, hardware-based convolutional networks have become a current research hotspot. Although CNNs perform excellently in computer vision and image processing, their large computational and data requirements often lead to efficiency and speed bottlenecks in software implementations [1][2]. Therefore, utilizing hardware to enhance the performance of convolutional neural networks is particularly important. For instance, CNNs involve many matrix operations and convolution calculations, which are time-consuming for conventional CPUs. By integrating specialized hardware such as ASIC, FPGA, and TPU, the parallel performance of networks is greatly improved, accelerating both training and inference speeds. Furthermore, the computational demands of convolutional neural networks often require substantial energy, especially in large-scale applications, while dedicated hardware can effectively reduce network power consumption, thereby enhancing system energy efficiency and making it more suitable for mobile terminals and edge devices.

© 2025 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/121/2025.19490

In applications such as autonomous driving and intelligent monitoring, the scenarios for convolutional neural networks often require high real-time computing capabilities[3]. Hardware design can effectively reduce network latency and utilize efficient computational architectures to meet real-time processing needs. Overall, this project aims to embed convolutional neural networks into various terminals (such as embedded systems and edge computing) to expand their application fields and promote the widespread use of deep learning.

2. Current Research Status and Development Trends

2.1. Domestic Research Status

Tsinghua University has independently developed the intelligent hardware platform Thinker, which aims to achieve dynamic optimization for various deep learning tasks using a configurable computing framework.[4] The design philosophy of Thinker is to improve energy efficiency and computational performance by flexibly configuring hardware resources to adapt to different application scenarios. The project's features include: using multiple general-purpose computing modules (PES) in the computation module to support different computational tasks such as convolution, fully connected layers, and pooling. The storage module integrates high-performance memory modules, effectively connecting various computing modules through special data channels to reduce data transfer latency. The unique controller module realizes dynamic allocation of computational resources and flexibly adjusts the working modes of each module. The adaptive computing architecture dynamically configures computing resources based on the requirements of different deep learning tasks, enabling efficient operation of different types of computations on the same chip.In the data interaction section, thinker has implemented two 144KB local buffers, which can implement pingpong operations for calculation and storage. Each local buffer is divided into 48 entries that can provide 48x16 bits of data bandwidth. For two 16x16 matrices, 32 entries are sufficient. ports corresponding to the remaining 16 entries provide additional data read and write channels after PE array reconstruction. And from the algorithm description, thinker also provides a convolutional data decompression method, can achieve the convolution data supply with a smaller number of ports.

2.2. International research status

Among numerous recent international studies, Microsoft Project Brainwave stands out, aiming to provide high-performance, low-latency hardware acceleration solutions for deep learning inference using Field Programmable Gate Arrays (FPGAs)[5]. Due to their flexibility and reconfigurability, FPGAs are suitable for addressing different deep learning models and real-time inference requirements. The design philosophy of Project Brainwave is to use FPGAs as hardware accelerators to dynamically deploy and optimize the inference process of various deep learning models. Through DMA technology, data can be directly transferred between memory and computing units, reducing CPU involvement and minimizing data transfer latency. A layered data flow architecture is designed to separately manage computation and data transfer, optimizing data transfer paths and enhancing overall computational efficiency. The dynamically reconfigurable computing resources provide a unified hardware abstraction layer, allowing different models to switch seamlessly on the same FPGA, improving resource utilization. An automatic optimization mechanism converts and quantizes deep learning models to make them more suitable for FPGA hardware execution, reducing latency. It also supports various deep learning models, achieving an efficient data flow architecture.

DOI: 10.54254/2755-2721/121/2025.19490

3. Key theories and technologies

3.1. Quantization and fixed point operations

A real-time CNC (Computer Numerical Control) mechanical fault detection solution is proposed, using a Binary Weight Convolutional Neural Network (BNN) to identify vibration signals, achieving an accuracy of 95.07% on an FPGA at a working frequency of 130 MHz. By converting operations to fixed-point calculations, computational speed is significantly improved, and memory usage is reduced.[6] This research not only focuses on model accuracy but also considers the hardware implementation costs, meeting the needs of industrial applications and enabling real-time monitoring and self-prevention of mechanical failures, thereby reducing maintenance costs and downtime. By quantizing and optimizing the network structure, hardware resource usage is minimized, making it suitable for deployment in resource-constrained environments.

A new YOLO CNN model (tiny-YOLO-Inception-ResNet2) combines the Inception-ResNet module, optimizing computational efficiency [7]. The study investigates the impact of varying the number of computing units on image detection time, finding that increasing computing units can linearly enhance performance. When using 32-bit floating-point numbers, it maintains detection accuracy comparable to software implementations. This model optimizes computational speed while preserving high accuracy, making it suitable for real-time image processing applications. By removing batch normalization operations, the model's computational complexity is simplified, reducing power consumption, and is applicable in scenarios requiring efficient image recognition, such as drones, with broad application potential.

3.2. Unified computing unit and floating-point calculation

A convolutional neural network for image recognition, consisting of 7 layers, achieves a prediction accuracy of 99.54% [8]. By employing a model-driven design approach, the network architecture is simplified and optimized on an FPGA, enabling fixed-point format calculations. This research demonstrates the efficiency of FPGAs in image recognition, suitable for applications requiring rapid processing, such as medical, autonomous driving, and quality control. By optimizing the network structure and reducing the number of filters, although accuracy decreases, it remains within an acceptable range for practical applications. The flexibility of hardware implementation allows the network to be adjusted according to different needs, showcasing good adaptability.

3.3Logarithmic and exponential computing units and Microarchitecture design

A multi-sensor fire detection system based on fuzzy logic and convolutional neural networks (CNNs) can process monitoring videos in real-time and detect fires [9]. The system integrates multiple sensors (smoke, flame, and temperature) to enhance detection accuracy and classifies video streams using CNNs, developing a web and mobile-based real-time alert notification system that promptly informs users and fire departments in the event of a fire. The system effectively handles noisy data by combining fuzzy logic and CNNs, improving fire detection accuracy and reliability. Using multiple sensors overcomes traditional sensors' limitations in detecting fires over large areas, providing a more comprehensive fire monitoring solution. The system's navigation feature helps firefighters quickly reach fire scenes, enhancing response speed, especially in areas with unclear addresses.

3.3. Multiple parallel MAC unit

An efficient CNN hardware architecture employs approximate arithmetic operations to reduce the complexity of multiplication and addition operations, thereby improving processing speed and reducing power consumption. By using Dynamic Range Unbiased Multipliers (DRUM) and

DOI: 10.54254/2755-2721/121/2025.19490

approximate adders, a CNN accelerator suitable for ASIC implementation is designed [10]. This accelerator maintains high accuracy while reducing chip area by 15% and significantly lowering power consumption, making it suitable for resource-constrained environments. The use of approximate arithmetic leverages CNNs' tolerance for small errors, allowing hardware design optimization without significantly sacrificing accuracy. The flexibility of this architecture enables it to adapt to different CNN models, showcasing broad application potential.

3.4. GSM module and multi-sensor data fusion

A dedicated computing core is designed to handle bipolar morphological convolutions, reducing the need for multiplication operations and thereby lowering hardware costs [11]. The bipolar morphological model significantly simplifies calculations (using addition and maximum operations), making it suitable for hardware implementation. Although the bipolar morphological model slightly lags behind classical models in latency, it excels in hardware resource utilization and power consumption.

4. Limitations and future development directions

Overall, despite the excellent performance and energy efficiency of FPGAs and other hardware accelerators, there are still resource limitations when implementing complex CNNs. Particularly when processing large-scale models, hardware resource constraints can lead to performance bottlenecks. Although researchers have optimized models, balancing high accuracy with reduced computational complexity remains a challenge. Existing optimization methods often require trade-offs between accuracy and speed, making it difficult to find the optimal balance. In practical applications, further reducing energy consumption is also a significant concern, especially in mobile devices and embedded systems, where improving energy efficiency is crucial.

Future research should focus on developing more efficient model compression techniques, such as pruning, quantization, and knowledge distillation, to reduce computational demands and storage requirements while maintaining high accuracy. By combining the advantages of FPGA, GPU, and CPU, a heterogeneous computing architecture that collaborates between software and hardware should be constructed to achieve higher computational performance and flexibility. When designing new hardware accelerators, greater attention should be paid to improving energy efficiency, particularly in mobile and edge computing devices, where developing low-power, high-performance solutions will be an important direction.

By conducting in-depth research and exploration of these limitations, future work can further advance convolutional neural networks and their hardware acceleration technologies to meet the growing application demands.

5. Conclusion

This article explores the hardware implementation and optimization of convolutional neural networks (CNNs) in various applications, covering fire detection, CNC mechanical fault diagnosis, image recognition, and more.

The multi-sensor fire detection system based on fuzzy logic and CNNs integrates sensor data and video processing to enhance early fire detection and alarm capabilities. This system overcomes the limitations of traditional methods by fusing multiple fire features. The FPGA-based YOLO subclass CNN model focuses on real-time image processing, emphasizing the balance between improving computational speed and accuracy in hardware implementation. Research indicates that using 32-bit floating-point calculations can maintain high detection accuracy.

DOI: 10.54254/2755-2721/121/2025.19490

Focusing on CNC mechanical fault detection, a binary weight CNN hardware accelerator is proposed, utilizing vibration signals for fault identification. This method reduces hardware overhead through quantization operations and achieves real-time monitoring on FPGA. The efficient CNN hardware architecture employs approximate arithmetic operations to lower power consumption and area while maintaining high accuracy. Research shows that using approximate multipliers and adders has minimal impact on CNN performance [6].

The FPGA-based CNN image recognition system demonstrates high-precision image recognition achieved through model-driven design. Although network complexity is reduced during optimization, a high recognition rate is still maintained. The hardware implementation of classical and bipolar morphological models proposes a new computing core that replaces multiplication with addition and maximum operations, significantly reducing hardware costs. Research results indicate that the bipolar morphological model is competitive in speed and area but slightly lags behind classical models in latency [11].

Overall, these studies showcase the diversity and flexibility of CNNs in hardware implementation, emphasizing the importance of optimizing performance and reducing costs in various application scenarios. They provide more efficient and reliable solutions by combining advanced algorithms and hardware design, demonstrating significant practical application value.

References

[1] *Tan, Y. (2024). Image classification VGG16 hardware implementation and sensor-based side channel security research. Heilongjiang University.*

[2] *Ji, M. (2024). Research on convolutional neural network accelerators based on configurable parallelism and automatic optimization of network architecture. Jilin University.*

[3] *Shuen, Z., Gong, Z., & Zhao, D. (2024).Traffic signs and markings recognition based on lightweight convolutional neural network. The Visual Computer, 40(2), 559-570.*

[4] *Yin, S., Ouyang, P., Tang, S., et al. (2018). Thinker: Reconfigurable hybrid neural network computing chip. Artificial Intelligence, (02), 34-45.*

[5] *Chung, E., et al. (2018). Serving DNNs in real time at datacenter scale with Project Brainwave. IEEE Micro, 38(2), 8-20.*

[6] *Chung, C.-C., Liang, Y.-P., Chang, Y.-C., & Chang, C.-M. (2023). A binary weight convolutional neural network hardware accelerator for analysis faults of the CNC machinery on FPGA. In 2023 International VLSI Symposium on Technology, Systems and Applications (VLSI-TSA/VLSI-DAT) (pp. 1-4).*

[7] *Markov, N. G., Zoev, I. V., & Mytsko, E. A. (2022). FPGA hardware implementation of the YOLO subclass convolutional neural network model in computer vision systems. In 2022 International Siberian Conference on Control and Communications (SIBCON) (pp. 1-4).*

[8] *Akimova, J. E., & Budanov, D. O. (2023). Hardware implementation of a convolutional neural network. In 2023 International Conference on Electrical Engineering and Photonics (EExPolytech) (pp. 72-75).*

[9] *Olegovich, T. M., Maksimovich, A. D., & Evgenievna, L. E. (2021). Hardware implementation of classical and bipolar morphological models for convolutional neural network. In 2021 International Conference Engineering and Telecommunication (En&T) (pp. 1-5).*

[10] *Elbtity, M. E., Son, H.-W., Lee, D.-Y., & Kim, H. (2020). High speed, approximate arithmetic based convolutional neural network accelerator. In 2020 International SoC Design Conference (ISOCC) (pp. 71-72).*

[11] *Sowah, R. A., Apeadu, K., Gatsi, F., Ampadu, K. O., & Mensah, B. S. (2020). Hardware module design and software implementation of multisensor fire detection and notification system using fuzzy logic and convolutional neural networks (CNNs). Journal of Engineering, 2020, Article ID 3645729.*