

A Review on FPGA-Based Optimization and Applications of Deep Learning Convolutional Neural Networks

Zhongyu Wang¹, Qi Li¹, Yifei Ge¹ and Lin Meng^{2,*}

¹Graduate School of Science and Engineering, Ritsumeikan University, 1-1-1 Noji-higashi, Kusatsu, Shiga, 525-8577, Japan

²College of Science and Engineering, Ritsumeikan University, 1-1-1 Noji-higashi, Kusatsu, Shiga, 525-8577, Japan

Abstract

In today's era of rapid technological advancement, Convolutional Neural Networks (CNNs) have demonstrated superior performance in many fields. As a key component of deep learning, CNNs have proven to be highly effective across various applications. Deploying CNNs on Field Programmable Gate Array (FPGA) is a challenging task due to the computational and storage requirements. This paper provides a comprehensive review of CNNs deployment on FPGA, covering the history of CNNs and explaining the key layers. A survey is conducted on FPGA optimization methods, and FPGA optimization methods are summarized by category. Optimizations for software deployment as well as hardware design have been made to improve computing on FPGA, further unlocking the potential of deploying CNNs on resource-constrained devices. Additionally, this review delves into examples of applications under power consumption constraints. Overall, this review offers significant reference value for researchers to understand CNNs architectures, explore FPGA acceleration methods, and application prospects.

Keywords

FPGA, CNNs, Deep Learning, Hardware Acceleration

1. Introduction

With the rapid development of the big data industry and the arrival of the internet of things era, the amount of global data has shown explosive growth in recent year. This surge in data provides a solid foundation and rich resources for the development [1] of artificial intelligence (AI). In this context, deep learning, as a core technology and an important research direction for realizing AI, has received widespread attention and rapid development. In particular, deep learning models based on neural networks has become a hotspot for research due to their superior ability in processing complex data and recognizing patterns [2].

The 6th International Symposium on Advanced Technologies and Applications in the Internet of Things (ATAIT 2024), August 19-22, 2024, Kusatsu, Shiga, Japan

*Corresponding author.

✉ gr0684hx@ed.ritsumei.ac.jp (Z. Wang); gr0517rs@ed.ritsumei.ac.jp (Q. Li); gr0607pe@ed.ritsumei.ac.jp (Y. Ge); menglin@fc.ritsumei.ac.jp (L. Meng)

ORCID 0009-0007-8498-0745 (Z. Wang); 0000-0002-1963-5263 (Q. Li); 0000-0003-3479-9802 (Y. Ge); 0000-0003-4351-6923 (L. Meng)

© 2024 Copyright for this paper by its authors.
Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

Among many deep learning models, convolutional neural networks (CNNs) have attracted the attention of a large number of research institutes by virtue of their high accuracy and excellent performance in the fields of image recognition, speech processing, etc. CNNs are able to automatically extract and learn features from data through layer-by-layer convolution and pooling operation, which significantly improves the effectiveness of various real-world applications. From cultural heritage protection to medical image analysis and natural language processing, CNNs play a crucial role in practical applications in various fields, driving the continuous progress of AI technology.

However, the size of neural networks is rapidly expanding as the accuracy requirements and complexity increase in real-world applications [3]. This increase in data volume imposes significant computational demands, challenging the capabilities of many traditional hardware platforms. At the same time, many application scenarios impose stringent requirements on the performance, low power consumption and real-time performance of hardware devices. In addition, how to achieve low power consumption while maintaining high performance for large-scale deep learning neural networks has become a major challenge [4]. These demands have motivated researchers and engineers to continuously explore new computing platforms to meet these demanding requirements.

Common platforms for accelerating deep learning include central processing units (CPUs), graphics processing units (GPUs), field programmable gate arrays (FPGA), and application-specific integrated circuits (ASICs) FPGA is flexible and configurable integrated circuits that internally consist of multiple programmable logic blocks [5]. Each block contains logic gates that perform computational tasks. The modules within the FPGA are connected together through a programmable interconnect network and signals are transmitted through I/O modules [6]. The user can change the functions of the logic blocks and the interconnection paths between the modules to customize circuit functions and data flow routes.

FPGA can perform a large number of parallel operations at the same time and have lower latency through direct hardware implementation of functions as compared to CPUs. FPGA can reconfigure their internal logic as per the requirement whereas the hardware architecture of CPUs is fixed. In addition FPGA have a higher energy efficiency ratio and are more effective in processing and applying real-time data [7].

FPGA has a higher energy-efficiency ratio compared to GPUs. FPGA is more energy-efficient when performing specific tasks, and its hardware can be highly optimized for specific applications. At the same time, FPGA is flexible and can be reconfigured to suit different tasks and application requirements, making them particularly suitable for development, testing and systems that require frequent upgrades.

FPGA has higher flexibility compared to ASICs, which has fixed hardware functions that cannot be modified after fabrication. ASICs have longer design, fabrication, and testing cycles, with high upfront development and fabrication costs. FPGA can be modified frequently during the design validation phase to minimize design errors and risks. Whereas ASIC designs, once finalized, can be costly to modify, and FPGA has a more low risk.

To provide researchers with a better understanding of CNNs and FPGA acceleration technology, this paper introduces the architecture of CNNs and the acceleration and optimization directions of FPGA. This paper also introduces specific application directions, providing researchers with a reference for understanding application areas.

2. The History and Development of CNNs

2.1. Origin and development

CNNs are type of deep learning model that is particularly suitable for processing image and video data [8]. Their design was inspired by research into the biological visual cortex in the 1980s. Researchers observed that the visual system of mammalian brains effectively processes complex visual information through highly structured layers. Based on these observations, scientists began designing artificial neural networks to mimic the brain's processing mechanisms [9].

The concept of CNNs can be traced back to the "Neocognitron" model proposed by Kuni-hiko Fukushima in 1980 [10]. Fukushima's model had a multi-layer structure that transmitted information layer by layer for image recognition. However, due to the lack of modern backpropagation training methods, its recognition performance for complex tasks was not very good [11].

In 1989, Yann LeCun and his team developed LeNet-5 [12], a convolutional neural network for handwritten digit recognition, marking a significant advancement in the practical application of CNNs. LeNet-5 effectively improved the performance of image recognition by combining convolutional layers, pooling layers, and fully connected layers with backpropagation algorithms for training [13]. It was successfully applied in the postal code recognition system of the United States Postal Service.

In the 21st century, with the advancement of big data and computing power, CNNs have undergone further development. In 2012, AlexNet [14] achieved breakthrough results in the ImageNet competition, demonstrating the powerful potential of deep learning. In the AlexNet network, the ReLU activation function and Dropout regularization were utilized to significantly reduce the error rate of image classification. This achievement showcased the remarkable potential of deep learning and sparked rapid development in the field [15].

Building upon AlexNet, many teams continued to optimize CNNs architectures, resulting in the development of models such as VGGNet, GoogLeNet, ResNet and YOLO [16].

- VGGNet (2014) : Proposed by the Visual Geometry Group [17] at the University of Oxford, multiple small 3x3 convolutional kernels are used instead of large ones, increasing the depth of the network while reducing the number of parameters and improving its capabilities.
- GoogLeNet (2015) : Proposed by Google [18], it adopts the inception module, which uses multiple convolutional kernels of different sizes and pooling operations to capture multi-scale features in the same layer, improving the feature extraction capability. A 1x1 convolutional kernel is used in the descending operation to improve computational efficiency.
- ResNet (2015) : Proposed by Microsoft Research, ResNet introduces a residual block design that skips one or more layers and directly adds the inputs to the outputs. This approach effectively addresses the problem of vanishing gradients that occurs during training, enabling the network to be trained at greater depths.
- YOLOv1 (2016) : Proposed by Joseph Redmon et al, YOLOv1 [19] treats the object detection problem as a single regression problem, directly predicting the object's boundaries

and categories from the input image. This approach significantly improves detection speed [16]. YOLOv1 uses full-image prediction, fully integrating the input image with global contextual information, thereby enhancing detection accuracy.

Table 1
Comparison of CNNs model architectures and performance metrics

Year	Architecture Name	Content	Parameter	Error Rate
1989	LeNet-5	Spatial Exploitation	0.06M	MNIST: 0.95
2012	AlexNet	Spatial Exploitation	60M	ImageNet: 16.4
2014	VGGNet	Spatial Exploitation	138M	ImageNet: 7.3
2014	GoogLeNet	Spatial Exploitation	4M	ImageNet: 6.7
2015	ResNet	Depth + Multi-Path	25.6M	CIFAR-10: 6.43
2016	YOLOv1	Unified Detection + Real-time Performance	62.4M	VOC-2012: 42.1

The development history of CNNs is shown in Table 1, including the content, parameter, and error rate. From Neocognitron to modern deep convolutional networks, the development of CNNs reflects the tremendous progress in artificial intelligence and deep learning technologies.

2.2. Structure and Principle

Multiple layers are combined, and together they form CNNs, with each layer serving a different purpose. In this subsection, we provide a brief overview of the key layers required to build a CNNs.

- **Input Layer:** The starting point of a neural network is the input layer, responsible for receiving raw data and passing it on to subsequent layers. In image processing tasks, the input layer typically receives a matrix of pixels, including the height, width, and number of color channels of an RGB image.
- **Convolutional Layer:** Convolutional operations can be thought of as sliding a convolutional kernel over the input data, extracting local features, and computationally generating a feature map. Convolutional layers can gradually extract higher-level features from captured low-level features, such as edges, after multiple layers of stacking.
- **Activation Functions:** activation functions introduce nonlinear transformations that allow the network to learn to represent complex nonlinear relationships. Activate the ReLU function in this process sigmoid and tanh are commonly used. Activation functions are typically applied to the outputs of convolutional and fully connected layers, significantly enhancing the performance of the model.
- **Pooling Layer:** Pooling, also known as subsampling, the pooling layer gradually reduces the spatial size of representations through downsampling operations. This reduction in size helps decrease the number of parameters and computation, lowering computational complexity and memory usage while preserving critical information.
- **Fully Connected Layer:** A fully connected layer whose neurons are fully connected to all activations of the previous layer, It expands the feature map of the previous layer into a one-dimensional vector and generates the output through matrix multiplication.

- **Output Layer :** The output layer is the last layer of the neural network and is responsible for generating the final prediction, usually using Softmax or Sigmoid functions to convert the output of the network into a probability distribution and generate the final prediction. Figure 1 gives an example of CNNs architecture.

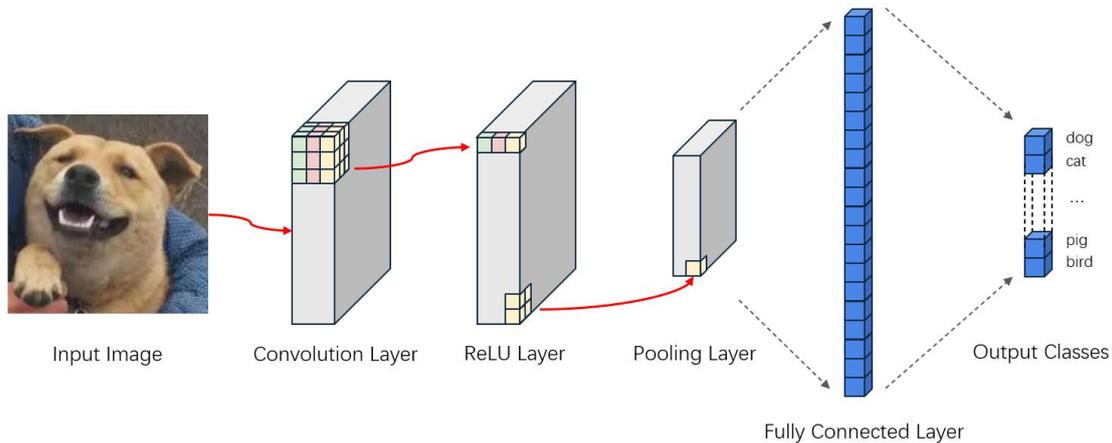


Figure 1: A sample CNNs architecture designed for image classification.

In order to meet the growing computational demands with limited energy consumption. The next step will explore the deployment of CNNs on FPGA to improve inference and training efficiency.

3. FPGA acceleration technology

As models deepen, the computational process involves a large amount of computation and data, which significantly increases the system burden. Using more energy-efficient FPGA for model deployment has become a promising option. Improving operational efficiency and reducing energy consumption are key focuses of current research. This section will introduce FPGA optimization techniques through software optimization and hardware optimization. Figure 2 shows the article structure.

3.1. Software Optimization

This section presents the software optimization aspect, primarily focusing on the optimization of CNNs network models and algorithms.

3.1.1. Optimization of network structure

In CNNs [20], meta-heuristic refers to an advanced optimization strategy or algorithm used to find the global optimal solution rather than being limited to locally optimal solutions.

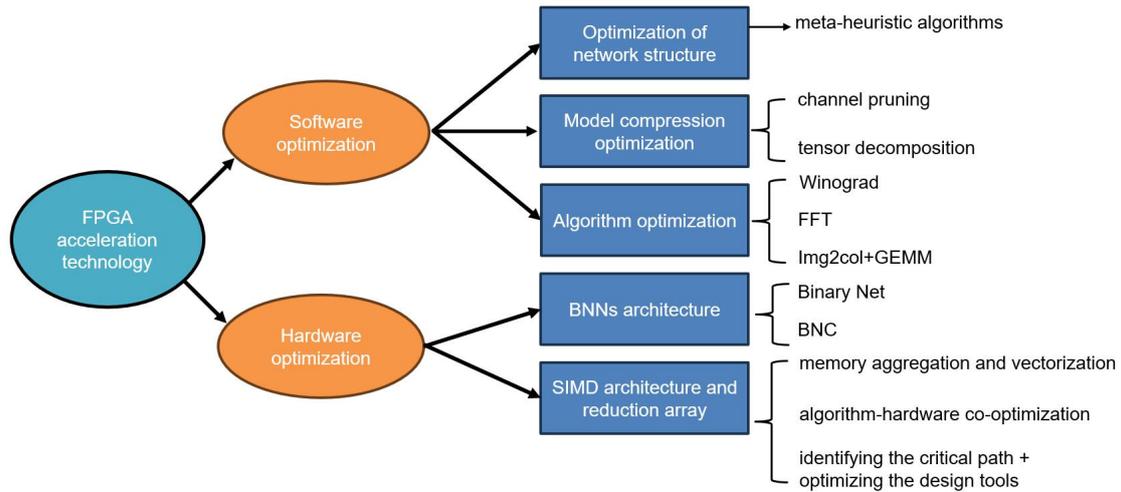


Figure 2: FPGA acceleration technology article structure.

In the literature, L. M. Rasdi Rere et al. compared the effectiveness of three meta-heuristic algorithms—Simulated Annealing, Differential Evolution, and Harmony Search in optimizing CNNs [21]. The computational time of the optimized CNNs increased compared to the original CNNs. These optimization effect analyses and research outlooks demonstrate the potential and future directions of metaheuristic algorithms in optimizing neural networks and deep learning architectures.

Traditional gradient-based backpropagation methods, although widely used, still have some problems such as local optima, being computationally expensive, and dependence on a continuous cost function. Mehrdad Kaveh et al. summarized the strengths and weaknesses of metaheuristic (MH) algorithms in deep learning and CNNs optimization and proposed future research directions in integrating MH algorithms and deep learning, especially the potential of hybrid MH algorithms [22]. MH algorithms have a significant advantage in the optimization of neural networks and deep learning architectures, and there are many directions worth exploring in future research.

3.1.2. Model compression optimization

Compressing and optimizing the model can greatly reduce computational and storage costs. In this section, we will review [23] the optimization techniques [24] for model compression on CNNs.

Li et al. proposed a Collaborative Compression (CC) method that combines channel pruning and tensor decomposition to compress CNNs by simultaneously learning the sparsity and low-rank nature of the model [25]. The compression sensitivity of each layer is analyzed by constructing a model of the relationship between information loss and compression rate. The results showed that the CC method significantly outperforms existing methods. For example, it achieved a 52.9% reduction in FLOPs on ResNet-50, while decreasing the Top-1 accuracy by only 0.56%.

Luis Balderas et al. proposed a method for optimizing CNNs architectures and named it OCNNsA [26]. The method uses techniques such as pruning and knowledge refinement to identify and retain the most important filters. It used Principal Component Analysis, Frobenius Paradigm and Coefficient of Variation to determine the importance of the filters, thus simplifying the model with minimal loss of accuracy. The method performed well and significantly reduced the number of parameters while maintaining high accuracy. The method reduced computational cost and energy consumption, ensuring that the neural network remains effective and accurate when deployed on resource-limited devices.

By compressing the CNNs [27], power consumption can be significantly reduced, and computational complexity, memory usage, and storage space consumption can be decreased. This improves the deployability and environmental friendliness of the model [23].

3.1.3. Algorithm optimization

When deploying CNNs in FPGA, the algorithms need to be optimized to improve performance and resource utilization. The use of Winograd algorithm and Fast Fourier Transform (FFT) can significantly reduce computational complexity and improve computational efficiency [28].

Wang et al. used the Winograd algorithm to design an efficient FPGA accelerator that significantly reduced the number of multiplication operations [29]. Wang proposed the Sparse Winograd-ReLU algorithm, which combines the MBM coding format and the Scatter-Compute-Gather method to significantly optimize the FPGA performance and energy efficiency of the CNNs accelerator. By addressing the irregularities of sparse data, the application efficiency of the Sparse Winograd algorithm is improved, and its scalability is enhanced for applications in embedded systems and high-performance computing environments.

He et al. proposed a fast convolutional algorithm based on FFT pruning, which removed redundant addition operations through an intelligent pruning method and reduces more than 50% of the addition operations compared to existing algorithms, while maintaining better numerical accuracy [30]. An efficient reconfigurable architecture was designed to support convolutional operations with different convolutional kernel sizes, achieving a throughput of 200.6 GOPS on a Xilinx ZC706 FPGA with a 61% improvement in resource efficiency.

The two-step process of converting convolution operations into matrix multiplication is known as *Im2col* and GEMM.

Ye et al. proposed a unified FPGA-based acceleration design that selects the optimal algorithm (*im2col* or frequency-domain convolution) for each layer of convolutional operations [31]. The optimal algorithm was selected through a performance model, reducing latency by 3.4x to 6.7x compared to the CPU implementation. The computational efficiency of homomorphic crypto convolution is significantly improved.

By performing algorithmic optimizations, model performance can be greatly improved and deployment flexibility can be increased.

3.2. Hardware design optimization

Hardware design optimization plays a vital role in the accelerator FPGA. In this section hardware design optimization techniques will be reviewed.

3.2.1. BNNs Architecture

Binarized Neural Networks (BNNs) are a special type of neural network architecture in which weights and activation values are restricted to binary values.

Tang et al. explored the challenges of deploying CNNs in an edge computing environment [32]. The study compared the accuracy of different BNNs methods on the CIFAR-10 dataset. The BinaryNet method showed a significant decrease in accuracy due to the loss of information caused by the binarization of activation values. Using methods like BNC, which quantized only the weights, accuracy can be kept high and close to that of a full-precision network. FP-BNN achieved high performance in the number of operations per second (GOP/s) but has relatively high power consumption. The study also pointed out that network sparsity can be utilized to skip unnecessary computations, leading to significant power savings. Finding a balance between efficiency and resourced consumption is crucial for implementing BNNs on FPGA efficiently.

3.2.2. SIMD Architecture and Reduction Array

Single Instruction Multiple Data (SIMD) architecture is a parallel computing architecture that can operate on multiple data elements with a single instruction. Optimizing SIMD architecture can significantly improve computational performance when dealing with large datasets. Reduction Array Optimization is a technique commonly used in parallel computing to combine multiple elements of an array into a single result.

Wang et al. accomplished the computational optimization of the SIMD approach through memory aggregation and vectorization [33]. By using group convolution and a new channel shuffling process, the memory footprint of each device was reduced, and inter-device synchronization is eliminated. A parallel FPGA accelerator for ShuffleNet was designed based on this approach. In terms of time consumption, I-ShuffleNet reduced the inference time by nearly half compared to the original ShuffleNet. Regarding resource utilization, RAM usage was reduced from 1707 to 1265, and memory utilization was reduced by 34% when using two FPGA devices.

Ni et al. proposed an algorithm-hardware co-optimization approach to achieved CNNs acceleration on FPGA for remote sensing image processing via SIMD architecture [34]. The improved YOLOv2, VGG-16, and ResNet-34 networks were deployed on an AMD-Xilinx VC709 evaluation board, and the experimental results showed that the throughput of the improved YOLOv2 is 386.74 GOPS, that of the VGG-16 was 344.44 GOPS, and that of the ResNet-34 was 182.34 GOPS, significantly outperformed existing related work. The model optimization technique proposed by Ni significantly reduced the hardware resource requirements of the model and improved the energy efficiency of the system through operations such as operation fusion and depth-first mapping.

Zhang et al. presented two strategies to improved the frequency performance of shrinking array-based CNNs on FPGA by identifying the critical path and optimizing the design tools [35]. To addressed the critical path problem in the shrinking array design, the length of the DSP chain was reduced at the front end of the FPGA design, and layout constraints were imposed at the back end by optimizing the front end and the back end separately. The optimized design achieved a frequency of 290 MHz on the VGG16 network, an improvement of about 50% over

the unoptimized baseline version. This directly improved throughput by 27.6% and reduces processing latency by 21.6%.

With hardware optimization techniques, the efficiency and performance of deploying CNNs on FPGA can be dramatically improved, allowing them to be used in multiple scenarios.

3.3. Summary

The FPGA acceleration techniques in the third section are primarily realized through software optimization and hardware optimization. This paper posits that the flexibility and reconfigurability of FPGA makes it highly potential candidates for resource-constrained applications requiring high performance, particularly in fields such as object detection, anomaly detection, and identification of ancient documents. Future research should focus on balancing the high performance and low energy consumption of FPGA to achieve broader applications.

4. Applications

There are many practical applications [36] for deploying CNNs on FPGA by utilizing their low power consumption and parallel computing features. This subsection will be organized into three parts. Figure 3 gives sample of applications. [37].

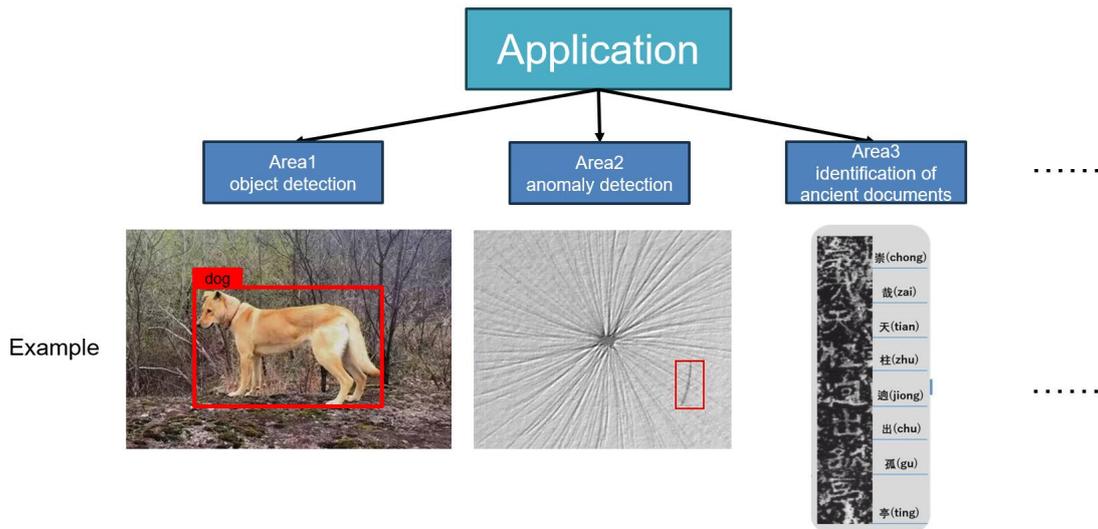


Figure 3: A sample of applications.

4.1. Object detection

Equipping CNNs on FPGA for object detection [38] is an efficient solution in scenarios with high requirements for real-time [39] and power consumption [40].

RB et al. designed a system for eye detection in long-distance iris recognition [41]. The system used an AMD/Xilinx Zynq UltraScale+ MPSoC to implement a Tiny YOLO-v3 network. At a distance of 2 meters, the system could accurately capture and process eye images of a person in motion. This system, which improved biometric accuracy and speed, was suitable for security surveillance and identity verification.

Harada et al. proposed an object detection system implemented on a Xilinx ZCU102 FPGA, utilizing the YOLOv3 network for real-time object detection in an autonomous driving system [42]. In actual road testing, the system operated stably in various complex scenarios, achieving a real-time detection rate of 99.8% and adapting to different lighting and weather conditions.

4.2. Anomaly detection

Anomaly detection by piggybacking CNNs on FPGA is an effective approach.

Wess et al. designed an abnormality detection system for electrocardiograms [43]. The system used principal component analysis for feature dimensionality reduction and a multilayer perceptron for classification. The design significantly reduced the required hardware resources and computational latency through segmented linear approximation of the activation function and a fixed-point implementation method. With the optimized design, the neural network achieved an average classification accuracy of 99.82% on the MIT-BIH arrhythmia database.

A system for fire detection using multispectral imagery was designed by Coca et al [44]. A modeling anomaly detector was implemented to detect weather-induced disasters and natural hazards on a Xilinx Zynq UltraScale+ XCZU9EG multiprocessor system-on-chip (MPSoC) device. The accelerator excelled in recognizing fire scenes acquired by Sentinel-2 in the Spanish and French regions, enabling rapid generation of early warnings and interventions when hazardous events are imminent.

4.3. Identification of Ancient Documents

Ancient document [45] identification is a complex but valuable task [46]. Deployment using FPGA can enhance the efficiency of document digitization and preservation [47].

Rizk et al. presented a reconfigurable capsule network hardware accelerator designed for processing ancient text symbols with sparse annotations [48]. The system used an improved capsule network architecture that preserved the spatial relationships of image entities through a dynamic routing algorithm, making it particularly suitable for dealing with small datasets and sparse annotations. Processing high-dimensional matrix operations through a parallel structure significantly reduced computational latency and improved throughput. Experiments on the Phoenician ancient text dataset show that the accelerator achieved a high accuracy of 0.9891 and a low loss value of 0.021. In a comparison with GPUs, the FPGA hardware accelerator achieved a 2x reduction in latency, allowing it to outperform GPUs in handling capsule network inference tasks. The design was particularly suitable for decoding ancient texts with sparse annotations, enabling efficient character segmentation and detection by maintaining the positional and gestural information of the characters.

5. Conclusion

This paper reviews the optimization and application of CNNs on FPGA and introduces the history and development of CNNs. This paper also focuses on enhancing FPGA performance through software and hardware optimization techniques, thereby enabling the efficient deployment of CNNs on resource-constrained devices. In addition, the paper explores the performance of FPGA in practical applications such as object detection, anomaly detection, and identification of ancient documents.

This paper posits that due to the large data volume, high computational demand, and frequent memory access of CNNs, deploying FPGA systems on accelerators remains challenging. FPGA has broad application prospects in scenarios requiring low power consumption and high performance. As the complexity and scale of deep learning models continue to increase, the advantages of FPGA in handling large-scale data and complex models will become more apparent. Future research should focus on further balancing the high performance and low energy consumption of FPGA to achieve broader applications.

References

- [1] Y. Hu, Y. Liu, Z. Liu, A survey on convolutional neural network accelerators: Gpu, fpga and asic, in: 2022 14th International Conference on Computer Research and Development (ICCRD), IEEE, 2022, pp. 100–107.
- [2] Z. Li, H. Li, L. Meng, Model compression for deep neural networks: A survey, *Computers* 12 (2023) 60.
- [3] Y. Wu, Review on fpga-based accelerators in deep learning, in: 2023 IEEE 6th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), volume 6, IEEE, 2023, pp. 452–456.
- [4] Y. Xu, M. Wang, X. Yue, R. ISHIBASHI, Z. Wang, L. Meng, Underwater robot gripper model design using 4d printing, *The papers of technical meeting on control, IEE Japan 2023* (2023) 1–5.
- [5] S. Zhenling, M. Lin, Development of a deep learning model using chatgpt and its acceleration in cpu, *The papers of technical meeting on control, IEE Japan 2023* (2023) 1–6.
- [6] A. Shawahna, S. M. Sait, A. El-Maleh, Fpga-based accelerators of deep learning networks for learning and classification: A review, *IEEE Access* 7 (2018) 7823–7859.
- [7] C. Wang, Z. Luo, A review of the optimal design of neural networks based on fpga, *Applied Sciences* 12 (2022) 10771.
- [8] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, L. Farhan, Review of deep learning: concepts, cnn architectures, challenges, applications, future directions, *Journal of big Data* 8 (2021) 1–74.
- [9] Y. Ma, Y. Cao, S. Vrudhula, J.-s. Seo, Optimizing loop operation and dataflow in fpga acceleration of deep convolutional neural networks, in: *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, 2017, pp. 45–54.
- [10] K. Fukushima, Neocognitron: A self-organizing neural network model for a mechanism

- of pattern recognition unaffected by shift in position, *Biological cybernetics* 36 (1980) 193–202.
- [11] M. A. Arshad, S. Shahriar, A. Sagahyroon, On the use of fpgas to implement cnns: A brief review, in: *2020 International Conference on Computing, Electronics & Communications Engineering (iCCECE)*, IEEE, 2020, pp. 230–236.
 - [12] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, L. D. Jackel, Backpropagation applied to handwritten zip code recognition, *Neural computation* 1 (1989) 541–551.
 - [13] S. I. Venieris, A. Kouris, C.-S. Bouganis, Toolflows for mapping convolutional neural networks on fpgas: A survey and future directions, *ACM Computing Surveys (CSUR)* 51 (2018) 1–39.
 - [14] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, *Advances in neural information processing systems* 25 (2012).
 - [15] P. Xiyuan, Y. Jinxiang, Y. Bowen, L. Liansheng, P. Yu, A review of fpga-based custom computing architecture for convolutional neural network inference, *Chinese Journal of Electronics* 30 (2021) 1–17.
 - [16] X. Wang, H. Li, X. Yue, L. Meng, A comprehensive survey on object detection yolo, *Proceedings* <http://ceur-ws.org> ISSN 1613 (2023) 0073.
 - [17] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, *arXiv preprint arXiv:1409.1556* (2014).
 - [18] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.
 - [19] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, in: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.
 - [20] H. Li, L. Meng, Hardware-aware approach to deep neural network optimization, *Neuro-computing* 559 (2023) 126808.
 - [21] L. R. Rere, M. I. Fanany, A. M. Arymurthy, Metaheuristic algorithms for convolution neural network, *Computational intelligence and neuroscience* 2016 (2016).
 - [22] M. Kaveh, M. S. Mesgari, Application of meta-heuristic algorithms for training neural networks and deep learning architectures: A comprehensive review, *Neural Processing Letters* 55 (2023) 4519–4622.
 - [23] E. L. Denton, W. Zaremba, J. Bruna, Y. LeCun, R. Fergus, Exploiting linear structure within convolutional networks for efficient evaluation, *Advances in neural information processing systems* 27 (2014).
 - [24] Q. Li, H. Li, L. Meng, Deep learning architecture improvement based on dynamic pruning and layer fusion, *Electronics* 12 (2023) 1208.
 - [25] Y. Li, S. Lin, J. Liu, Q. Ye, M. Wang, F. Chao, F. Yang, J. Ma, Q. Tian, R. Ji, Towards compact cnns via collaborative compression, in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 6438–6447.
 - [26] L. Balderas, M. Lastra, J. M. Benítez, Optimizing convolutional neural network architecture, *arXiv preprint arXiv:2401.01361* (2023).
 - [27] S. I. Young, W. Zhe, D. Taubman, B. Girod, Transform quantization for cnn compression,

- IEEE Transactions on Pattern Analysis and Machine Intelligence 44 (2021) 5700–5714.
- [28] M. R. Vemparala, A. Frickenstein, W. Stechele, An efficient fpga accelerator design for optimized cnns using opencl, in: International conference on architecture of computing systems, Springer, 2019, pp. 236–249.
 - [29] X. Wang, C. Wang, J. Cao, L. Gong, X. Zhou, Winonn: Optimizing fpga-based convolutional neural network accelerators using sparse winograd algorithm, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 39 (2020) 4290–4302.
 - [30] L. He, X. Xie, J. Lin, Z. Wang, Efficient fpga design for convolutions in cnn based on fft-pruning, in: 2020 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS), IEEE, 2020, pp. 27–30.
 - [31] T. Ye, S. R. Kuppannagari, R. Kannan, V. K. Prasanna, Performance modeling and fpga acceleration of homomorphic encrypted convolution, in: 2021 31st International Conference on Field-Programmable Logic and Applications (FPL), IEEE, 2021, pp. 115–121.
 - [32] L. Zhang, X. Tang, X. Hu, T. Zhou, Y. Peng, Fpga-based bnn architecture in time domain with low storage and power consumption, Electronics 11 (2022) 1421.
 - [33] J. Wang, W. Tong, X. Zhi, Model parallelism optimization for cnn fpga accelerator, Algorithms 16 (2023) 110.
 - [34] S. Ni, X. Wei, N. Zhang, H. Chen, Algorithm–hardware co-optimization and deployment method for field-programmable gate-array-based convolutional neural network remote sensing image processing, Remote Sensing 15 (2023) 5784.
 - [35] J. Zhang, W. Zhang, G. Luo, X. Wei, Y. Liang, J. Cong, Frequency improvement of systolic array-based cnns on fpgas, in: 2019 IEEE International Symposium on Circuits and Systems (ISCAS), IEEE, 2019, pp. 1–4.
 - [36] Y. Wang, S. Cang, H. Yu, A survey on wearable sensor modality centred human activity recognition in health care, Expert Systems with Applications 137 (2019) 167–190.
 - [37] P. M. Bhatt, R. K. Malhan, P. Rajendran, B. C. Shah, S. Thakar, Y. J. Yoon, S. K. Gupta, Image-based surface defect detection using deep learning: A review, Journal of Computing and Information Science in Engineering 21 (2021) 040801.
 - [38] J. Ren, A. Wang, H. Li, X. Yue, L. Meng, A transformer-based neural network for gait prediction in lower limb exoskeleton robots using plantar force, Sensors 23 (2023) 6547.
 - [39] Z. Li, L. Meng, Deep spiking neural networks for image classification, International Journal of Human Factors Modelling and Simulation 8 (2023) 21–35.
 - [40] Y. Ge, Z. Li, X. Yue, H. Li, Q. Li, L. Meng, Iot-based automatic deep learning model generation and the application on empty-dish recycling robots, Internet of Things 25 (2024) 101047.
 - [41] C. A. Ruiz-Beltrán, A. Romero-Garcés, M. González-García, R. Marfil, A. Bandera, Fpga-based cnn for eye detection in an iris recognition at a distance system, Electronics 12 (2023) 4713.
 - [42] K. Harada, K. Kanazawa, M. Yasunaga, Fpga-based object detection for autonomous driving system, in: 2019 International Conference on Field-Programmable Technology (ICFPT), IEEE, 2019, pp. 465–468.
 - [43] M. Wess, P. S. Manoj, A. Jantsch, Neural network based ecg anomaly detection on fpga and trade-off analysis, in: 2017 IEEE International Symposium on Circuits and Systems (ISCAS), IEEE, 2017, pp. 1–4.

- [44] M. Coca, M. Datcu, Fpga accelerator for meta-recognition anomaly detection: Case of burned area detection, *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* (2023).
- [45] X. Yue, Z. Wang, R. Ishibashi, H. Kaneko, L. Meng, An unsupervised automatic organization method for professor shirakawa's hand-notated documents of oracle bone inscriptions, *International Journal on Document Analysis and Recognition (IJ DAR)* (2024) 1–19.
- [46] B. Lyu, X. Yue, L. Meng, Japanese literature organization and spatiotemporal database system creation for natural disaster analysis, *Heritage Science* 12 (2024) 14.
- [47] Z. Zhang, Z. Wang, H. Tomiyama, L. Meng, Deep learning and lexical analysis combined rubbing character recognition, in: *2019 International Conference on Advanced Mechatronic Systems (ICAMechS)*, IEEE, 2019, pp. 57–62.
- [48] R. Rizk, D. Rizk, F. Rizk, A. Kumar, M. Bayoumi, An efficient capsule network reconfigurable hardware accelerator for deciphering ancient scripts with scarce annotations, in: *2021 IEEE 34th International System-on-Chip Conference (SOCC)*, IEEE, 2021, pp. 75–78.