

Optimized design of hardware accelerator based on machine learning

Jixuan Liu*

Department of Electrical and Information Engineering, Beijing University of Civil Engineering and Architecture, Beijing, 102616, China

* Corresponding Author Email: 202204010107@stu.bucea.edu.cn

Abstract. Hardware accelerators are playing an increasingly important role in the field of computer science and engineering, especially in accelerating deep learning and large-scale data processing. However, the traditional hardware accelerator design often relies on manual adjustment and experience accumulation, which is inefficient. Therefore, the use of machine learning technology to optimize hardware accelerator design has become a new trend. In this paper, a hardware accelerator optimization design method based on machine learning is proposed, including hardware design, algorithm optimization, system-level optimization, and deep learning model compression. Through the research and verification of the actual case, this research proves the effectiveness and feasibility of the hardware accelerator optimization design method based on machine learning. Compared with the traditional design method, this method can find better design scheme in shorter time and has higher performance prediction accuracy. Therefore, this method can significantly improve the design efficiency and performance of hardware accelerators. The results of this study are of great significance for promoting the development of hardware accelerator technology. It can promote scientific research and engineering applications in related fields to make greater progress, and promote the development of artificial intelligence, big data processing and other fields.

Keywords: Machine learning, Hardware accelerator, Optimization design method.

1. Introduction

In today's information age, the rapid development of machine learning technology has caused revolutionary changes in countless fields, from natural language processing to computer vision, to automatic driving and medical diagnosis and other fields, machine learning has shown people a strong application potential [1]. The advent of machine learning has revolutionized the approach to data analysis, pattern recognition, and predictive modeling. However, with the complexity of machine learning algorithms and the growth of data scale, traditional processors have been difficult to meet people's needs for high performance and low energy consumption. Therefore, many researchers have begun to look at hardware accelerators based on machine learning, in order to achieve efficient acceleration of machine learning tasks. In recent years, the optimization design of hardware accelerators based on machine learning involves many aspects. Continuous breakthroughs and upgrades in hardware and algorithms.

The purpose of this paper is to introduce the theoretical background of machine learning and the principle and structure of hardware accelerator, and explore the innovative methods of machine learning in the optimization design of hardware accelerator. The optimization methods of hardware accelerator include hardware design, algorithm optimization, system-level optimization, and deep learning model compression. This article presents examples of various optimization schemes, focusing on how machine learning algorithms can optimize data flow to improve efficiency, parallelism, flexibility, and security.

2. Theoretical background

2.1. Machine Learning

Machine learning is a leading technology. It enables computers to continuously learn from data and improve and enhance their performance without the need for explicit programming algorithms. Our goal in this cutting-edge field is to develop advanced algorithms and statistically efficient models that can accurately identify patterns and patterns in the data in their calculations, and make predictions and decisions based on this identified content. Machine learning is divided into three main learning modes: supervised learning, unsupervised learning, and reinforcement learning [2]. In supervised learning, the algorithm learns by labeling a trained set of data with a correct output or label for each training sample. Supervised learning the essence of this algorithm is to learn how to map the input to the correct output in order to be able to make predictions about new, never-before-seen data. In this model of unsupervised learning, the training data does not contain labels, which is the biggest difference from supervised learning. Unsupervised learning is an algorithm that tries to find patterns and rules in data, and then cluster or reduce to reveal the most essential intrinsic characteristics and structure of data. In reinforcement learning, an algorithm learns how to perform a specific task by interacting with the external environment. Reinforcement learning is constantly trying different strategies in order to maximize some kind of cumulative reward. Reinforcement learning is a type of learning that is widely used in areas such as game play, robot control, and resource management.

In statistics, machine learning is the theoretical foundation of statistics, which provides applied tools such as probability theory, statistical inference, and regression analysis. Machine learning typically builds models based on statistical principles, using statistical analysis of data to make predictions, classifications, or clusters. At the same time, machine learning also includes optimization theory, which studies how to find the optimal solution or approximate optimal solution of a function and how to optimize the performance of a model by adjusting parameters. Many algorithms in machine learning involve the problem of parameter optimization, such as gradient descent method, Newton method, etc [3, 4]. In addition, linear algebra is one of the most widely used mathematical tools in machine learning. Matrix and vector operations play a very important role in the model representation, calculation, and optimization of machine learning. For example, the matrix multiplication between the weight matrix and the input vector in a neural network model is based on linear algebra. Finally, there is the probability graph model, which is a graph structure used to describe the relationship between variables. Bayesian networks and Markov networks are common probabilistic graph models that are used in machine learning to model dependencies and probability distributions between variables [5, 6]. These models can aid in tasks such as inference, classification, and generation.

As computing power continues to increase and an explosion of data grows, machine learning has made significant progress, especially in the field of deep learning. Deep learning, which can use and process very complex data structures, has shown surprising strength in areas such as image recognition, natural language processing, and autonomous driving technology. In the future, machine learning will continue to improve the efficiency of algorithms, solve training problems on small data, and increase the interpretability of models.

2.2. Structure and principle of hardware accelerator

2.2.1. Hardware accelerator structure

A hardware accelerator is a specialized hardware device that we use to speed up the execution of specific tasks, such as image processing, machine learning, cryptography, etc. The structure of hardware accelerators varies according to different task types. The following sections are generally included. The first control unit is responsible for directing the operating process of the hardware accelerator. The control unit receives instructions from the autonomous processor or other controller and decodes these instructions into operational instructions for various parts of the hardware accelerator. Then the data memory is used to store the input data, the intermediate results, and the

output data for subsequent calculation use. The type of data storage is organized according to the characteristics of different tasks and may include register files, caches, on-chip memory, or external memory. The second unit is the unit that performs specific operational tasks, such as matrix multiplication, convolution operations, encryption, and decryption. This can be a custom processor, field programmable logic array (FPGA) or application-specific integrated circuit (ASIC) to enable efficient parallel computing. There is also a data path that transfers data from the data memory to the computing unit, and the calculation results are transferred back to the data memory. A data path usually consists of multiple data transmission channels in order to support parallel data transmission and computation. The final interface mode is an interface module that communicates with external contacts or devices. Interface modules may include bus interfaces, network interfaces, memory controllers, or other hardware devices for data exchange. The Hardware accelerator structure is shown in figure 1.

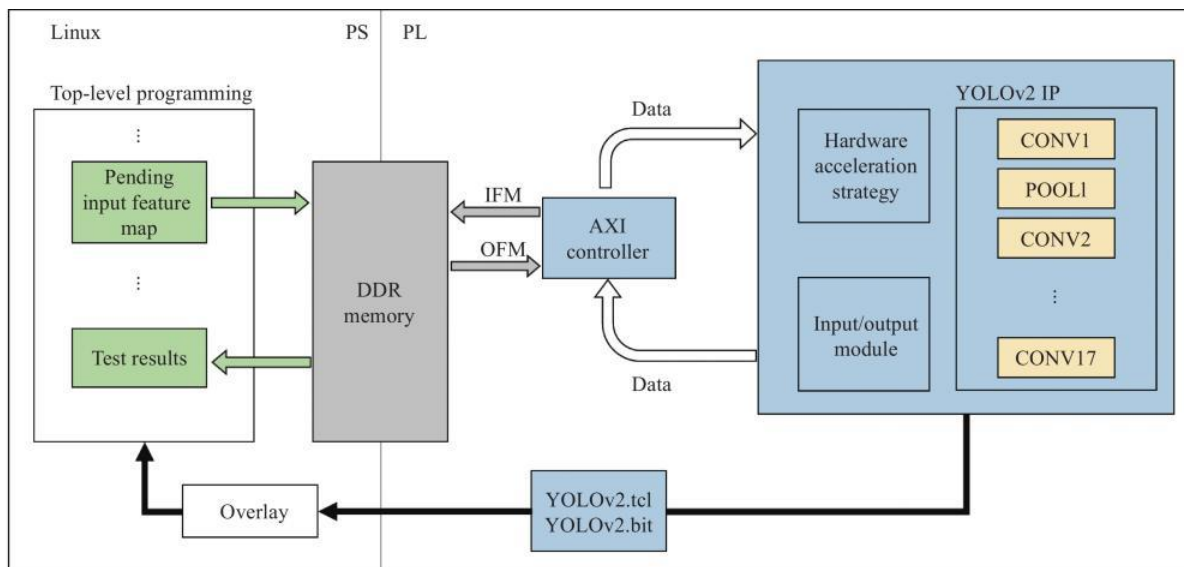


Fig. 1 Hardware accelerator structure [7]

2.2.2. Working principle of hardware accelerator

The working principle of the hardware accelerator is based on the idea of parallel computing and custom hardware design. By translating the computational requirements of a specific task into hardware circuitry, hardware accelerators are able to give us higher performance and energy efficiency than ordinary processors. At the same time, because hardware accelerators focus on the execution of specific tasks, they can better adapt and understand the characteristics of tasks, and provide lower latency and higher throughput.

In short, designers need to fully understand the application scenario and task requirements, select the appropriate hardware accelerator structure, and optimize the design, in order to achieve efficient acceleration of specific tasks, and can also provide important computational support for various fields.

3. Optimized design of hardware accelerator based on machine learning

3.1. Overview of hardware accelerator optimization design methods based on machine learning

Optimized design methods for machine learning-based hardware accelerators are a complex and diverse field in which efficient acceleration of machine learning tasks is achieved by optimizing hardware architecture and algorithm design. The method involves multiple levels and techniques, including hardware design, algorithm optimization, system-level optimization, and deep learning model compression. The following will be a general overview of these methods, and introduce the main ideas and technical means in each aspect.

3.1.1. Hardware Design

Hardware design is one of the key components of hardware accelerator optimization design method based on machine learning. In terms of hardware design, many researchers are often focused on designing efficient hardware architectures to meet the computational requirements of machine learning tasks. This includes designing dedicated accelerators for computing suitable for machine learning, such as graphics processors (GPUs), tensor processors (TPUs), and neural processors (NPU). These accelerators usually have the characteristics of strong parallel computing power, low energy consumption, and high performance, which can effectively accelerate the execution of machine learning algorithms.

3.1.2. Algorithm optimization

Algorithm optimization is also an important part of hardware accelerator optimization design method based on machine learning. In algorithm optimization, many researchers work on designing and optimizing accelerators suitable for hardware in order to satisfy efficient machine learning algorithms. These include optimizing deep learning algorithms such as convolutional layer neural networks (CNNs), recurrent neural networks (RNNs), and Transformers. The goal of algorithm optimization is to reduce computational complexity, reduce storage requirements, and utilize the parallel computing capabilities of hardware accelerators to achieve efficient acceleration of machine learning tasks.

3.1.3. System-Level Optimization

System-level optimization is also an important part of hardware accelerator optimization design method based on machine learning. System-level optimization includes the co-design and optimization between the hardware accelerator and the host system, in order to achieve the efficient transmission of data and the collaborative execution of tasks. This includes designing efficient data paths and storage structures, optimizing network on chip design, and designing efficient data transmission interfaces.

3.1.4. Deep learning model compression

Deep learning model compression is also one of the important technical means of hardware accelerator optimization design method based on machine learning. Since deep learning models have a large number of parameters and computations, they need to be compressed and optimized to accommodate the computing power of hardware accelerators. Deep learning model compression, including parameter pruning, weight quantization, model quantization and model distillation, is designed to reduce the storage requirements and computational complexity of models, thereby improving the execution efficiency of models on hardware accelerators.

3.2. Specific case analysis

3.2.1. Hardware optimization design

Stream method is a design method widely used in neural network algorithms, which optimizes the performance of each network layer by combining different hardware modules. These modules are carefully designed to maximize parallelism between the layers and are tightly connected by pipelining technology. As data flows through the architecture, it sequentially passes through the various parts of the neural network, enabling efficient data processing.

The core advantage of the Stream approach is that it can take full advantage of the parallelism between layers through pipeline technology, so that different processing units can be executed concurrently. This parallel design can not only improve the speed of data processing, but also effectively reduce the waste of resources. However, despite the advantages of the Stream approach, it also faces some challenges. Since each different neural network model needs to generate a new bit stream, this results in a long compilation time. Still, researchers are constantly exploring how to optimize the process to reduce compilation time and improve design efficiency.

In the representational work of Stream mode, DAG, As-Stream and Oracle have all achieved remarkable results [8-10]. Among them, the implementation of DAG utilizes CUDA's multi-stream pipeline parallelism. By analyzing the node connections and dependencies in DAG, the parallel execution sequence of multiple tasks is reasonably planned. In order to maximize the parallel capability of multiple streams, MS-DAG task mapping strategy considers the hardware resources and application requirements of asynchronous data transmission. The study found that Pinned Memory in the CPU-side Memory structure is more suitable for asynchronous data transfer than Pageable Memory, as the latter can lead to data transfer errors. When executing on the GPU, the data required for the application needs to be copied to the GPU's video memory. Asynchronous transmission is suitable for applications where data can be processed in batches, providing increased efficiency compared to synchronous transmission. For applications that can be processed in batches, MS-DAG makes use of the parallelism of multiple streams and combines asynchronous transmission and asynchronous call to realize the folding of data transmission and processing and improve the overall efficiency. The Diagram of multiple-stream pipeline is shown in Figure 2.

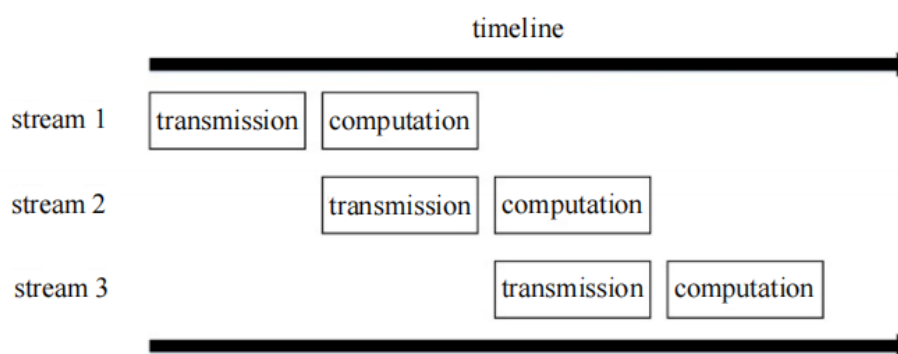


Fig. 2 Diagram of multiple-stream pipeline [8]

3.2.2. Optimization design for specific algorithms

In the process of seeking to improve the inference speed of support vector machine (SVM) algorithm, optimizing the multiplication and accumulation process of the vector to be classified and the support vector has become a key step. Especially in the face of large-scale data processing, effective hardware accelerator design is particularly important to improve the overall performance. Previous studies have proved that reasonable design of accelerator architecture can significantly speed up the point multiplication between vectors, thus improving the inference speed of SVM.

Taking the Hussain study in as an example, this paper proposes an accelerator design scheme combining FPGA and multiple Vector Processor clusters [11]. Each Vector Processor Cluster consists of multiple VPE arrays, and each VPE Array contains multiple Vpes (vector processing units) dedicated to processing vector dot products. This hierarchical structure design enables the accelerator to efficiently handle large-scale vector operations.

During execution, the large-scale matrix is fed into the accelerator as a data stream, while the small-scale matrix is stored in on-chip memory. Each VPE Array stores a specific column of a small-scale matrix, which makes efficient use of on-chip memory resources and improves data access efficiency. In addition, the accelerator also implements fine dot multiplication operations, which decompose each vector dot multiplication into multiple chunks dot multiplication. This approach not only reduces the amount of data per dot multiplication operation, but also helps balance the data transfer load between the FPGA and the CPU.

However, it should be noted that the accelerator structure is not optimized in terms of kernel function calculation. Kernel function is a key component of SVM algorithm, and its computational complexity is often high. Therefore, if the kernel function calculation can also be included in the accelerator optimization category, then the overall acceleration ratio of the system is expected to be further improved.

To give full play to the performance of the accelerator, it is necessary to adjust the chunk size properly to avoid the bottleneck of data transmission between the FPGA and the CPU. In practice, the best chunk size can be found through experiment and simulation, so that the accelerator can achieve the best performance when dealing with vector dot product and kernel function calculation.

3.2.3. System-level optimization design for specific problems

To solve the problem of Online Traffic Classification, a special accelerator is developed [12]. The problem of Online Traffic Classification involves identifying the application behind a TCP connection /UDP transport stream by analyzing the packets in it. To address this challenge, they used the common C4.5 decision tree algorithm. To improve the efficiency of the algorithm, they designed an innovative accelerator structure based on the characteristics of the problem.

The accelerator architecture is mainly composed of two parts: the discretization module and the classification module. The discretization module is responsible for preprocessing the input data, converting the continuous attribute values into discrete representations for subsequent classification processing. The classification module uses decision tree algorithm to classify and decide the discretized data. In the process of processing, the attribute vector of the input data is first fed into the discretization module, and after a series of discretization processing unit operations, the specific attribute value of the data is converted into a discrete representation. This data is then passed to the classification module for classification at each level of the decision tree. Each classifier maintains all intermediate/leaf node information at the current level of the decision tree. When the data is passed to the next level, the classification unit finds the corresponding intermediate node according to the received parameters (including the data attribute set and the intermediate node address), and continues the classification processing until it reaches the leaf node and obtains the final classification result.

With this dedicated accelerator, they hope to significantly improve the efficiency of Online Traffic Classification problems, providing a more efficient and accurate solution for network traffic analysis and management.

3.2.4. Designed for neural network algorithms and deep learning applications

Neural network algorithm and deep learning are hot topics in the field of machine learning, leading the rapid development of artificial intelligence technology. With the continuous innovation of algorithms, corresponding hardware accelerators have emerged, providing powerful computational support for the training and reasoning of deep learning models. Among them, the ASIC hardware acceleration chip DianNao developed by the Institute of Computing Science of the Chinese Academy of Sciences has attracted much attention, and its unique design concept and efficient performance make it stand out in the field of neural network hardware acceleration.

The design idea of DianNao chip is to divide the computing part of the neural network into multiplication unit, addition tree and activation unit, and map these units to the hardware logic, so as to realize the hardware acceleration of the entire computing part. This design enables DianNao chip to give full play to the advantages of hardware parallel processing and significantly improve the computing speed when processing neural network. Compared with traditional CPU and GPU, DianNao chip has lower power consumption and higher energy efficiency ratio when processing neural network computing, which provides strong support for the promotion of deep learning applications.

In addition to the development of hardware accelerators, software optimization is also the key to improve the computational performance of neural networks. In the neural network computing process, the cyclic process occupies a lot of computing resources. Therefore, the optimization of cyclic process is one of the important means to improve the computational performance of neural networks. In this regard, Li Yihuang and Liu Hua propose a cyclic tiling or cyclic partitioning method to optimize the cyclic process [13, 14]. By rationally arranging the computation sequence and computation amount of the cycle, the computation efficiency of the cycle process can be significantly improved, thus reducing the total computation time of the neural network.

In addition, cache management is also one of the key factors affecting the computational performance of neural networks. In the process of calculation, there is an independent relationship between each matrix, which makes the use of cache becomes particularly important. Therefore, a multi-cache mechanism is also proposed to optimize cache management [15]. By allocating cache space reasonably and managing cache access, the delay and conflict of data access can be effectively reduced, and the efficiency of neural network computing can be further improved.

4. Conclusion

Stream method is an effective neural network algorithm design method, which realizes high performance and efficient data processing through parallel design and pipeline technology. Although there are some challenges, such as long compilation time and high hardware cost, with the continuous progress and optimization of the technology, it is believed that these problems will gradually be solved. In the future, we look forward to seeing more innovative applications and practice cases based on the Stream approach to promote neural network algorithms to achieve greater breakthroughs and progress in various fields.

By optimizing the multiplication and accumulation process of the vector to be classified and the support vector, combined with reasonable hardware accelerator design, the inference speed of SVM algorithm can be effectively improved. Future studies can further explore how to incorporate kernel function calculation into accelerator optimization to achieve a higher overall system acceleration ratio. At the same time, the design of the accelerator structure also needs continuous innovation and improvement to adapt to the changing algorithm needs and data scale.

The rapid development of neural network algorithms and deep learning cannot be separated from the support of hardware accelerators and software optimization. ASIC hardware acceleration chip DianNao developed by Institute of Computing Science of Chinese Academy of Sciences is one of the representative works, and its unique design and high efficiency can provide powerful hardware support for neural network computing. At the same time, the application of software optimization method can further improve the performance of neural network hardware accelerators, so as to better support deep learning applications. With the continuous progress of technology, it is believed that neural network algorithms and deep learning will be widely used in more fields in the future, bringing more innovation and change to the development of human society.

References

- [1] Jordan M I, Mitchell T M. Machine learning: Trends, perspectives, and prospects. *Science*, 2015, 349(6245): 255-260.
- [2] Xu Hongxue, Sun Wanyou, Du Yingkui, et al. Classical machine learning algorithm and the application research review. *Computer knowledge and technology*, 2020, (33): 17-19.
- [3] Huang Jun. Research on high order PageRank problem and stochastic gradient descent method of predetermined restart in deep learning. China mining university, 2023.
- [4] Zhang Xinxin, Hou Ming, Fan Liya. Application of Newton's Method to solving support Vector Machine. *Journal of Liaocheng University (Natural Science Edition)*, 2014, 27(04): 10-15.
- [5] Feng Zhuoran, Liu Li, Yuan Yuling, et al. Flooding situation forecast model based on bayesian networks. *Journal of the science and technology*, 2024 (05): 1-3.
- [6] Zhao Ying, Han Chunhao. Markov models in the application of network flow classification and research. *Computer engineering*, 2018, 44 (5): 291-295.
- [7] Xia Qidi, Yan Bingyong, Zhou Jiale, Wang Huifeng. Architecture Design of Target Detection Hardware Accelerator Based on Heterogeneous FPGA. *Journal of East China University of Science and Technology*, 2021, 47(6): 706-715.
- [8] Wang Xuecheng, Ma Jinquan, Li Jianjun. Multi-stream Parallel DAG task mapping Strategy. *Journal of Terahertz Science and Electronics Information*, 2019, 17(02): 299-304+314.

- [9] Li Wei, Li Chenglong, Yang Jiahai. As-Stream: An operator intelligent parallelization strategy for fluctuating data streams. Journal of Tsinghua University (natural science edition), 2022 on conversion (12) : 1851-1863.
- [10] Wang Ernuan. Implementation of Oracle Read-write Separation using Stream replication. Fujian Computer, 2010, 26(05): 174+170.
- [11] Hussain T, Palomar O, Cristal A, et al. Mvpa: An fpga based multi-vector processor architecture. 2016 13th International Bhurban Conference on Applied Sciences and Technology (IBCAST). IEEE, 2016: 213-218.
- [12] Gandhi V R, Qu Y R, Prasanna V K. High-throughput hash-based online traffic classification engines on FPGA. 2014 International Conference on ReConFigurable Computing and FPGAs (ReConFig14). IEEE, 2014: 1-6.
- [13] Li Yihuang, Ma Sheng, Guo Yang, et al. Configurable neural network Accelerators based on tiled data streams. Computer Engineering and Science, 2019, 41(06): 963-972.
- [14] Liu Hua, Tao Guannan, Yang Wenqing. Convolutional Neural network Parallelization design and FPGA implementation. Manufacturing Automation, 2022, 44(11): 147-154.
- [15] Hu Xiaolong, Zhu Yanliang. Research on image input caching mechanism based on FPGA. Microcomputer Information, 2010, 26(02): 5-6+25.