Performance Review of Deep Learning on FPGAs and GPUs: Transforming Next-Generation Industrial Applications

Santhoshini P[1],Sadha B[2],Nikshitha V[3],Murugavel chetty M K [4]
[1] *Assistant Professor, Department of ECE, R.M.D Engineering College, Thiruvallur*
[2,3,4] *B.E Student Department of ECE, R.M.D Engineering College, Thiruvallur*

Abstract*—Industries are increasingly focusing on sophisticated components for the development of multi-purpose machinery and devices, driven by the habitual integration of engineering and technology. Significant advancements in the fields of electronics, computer science, and automation, particularly in clustering, deep learning, neural networks, and machine learning techniques, are being leveraged. These advancements are being implemented in modern components such as Field-Programmable Gate Arrays (FPGAs) and Graphics Processing Units (GPUs). This review paper comprehensively examines the development and application of deep learning algorithms in FPGAs and GPUs. We explore the latest trends, compare the performance and efficiency of various approaches, and discuss the implications of these technologies for future industrial applications. The findings highlight the potential for improved computational efficiency and the acceleration of complex tasks, which could significantly impact the design and functionality of next-generation machinery and devices

***Keywords*—VLSI, Deep Learning, FPGA, Neural Networks.**

Definitions:

VLSI: It refers to the process of creating integrated circuits (ICs) by combining thousands to millions of transistors onto a single chip. VLSI technology is a critical aspect of modern electronics, enabling the development of complex devices such as microprocessors, memory chips, and other digital and analog circuits.

MICROPROCESSOR: A microprocessor is an integrated circuit (IC) that serves as the central processing unit (CPU) of a computer or other digital device. It performs the arithmetic, logic, control, and input/output (I/O) operations specified by the instructions in a program.

FPGA: It is an integrated circuit (IC) that can be programmed or reprogrammed by the user after manufacturing. Unlike traditional application-specific integrated circuits (ASICs), which are designed for a specific task during fabrication, FPGAs can be customized and reconfigured multiple times to perform a wide range of functions.

Machine Learning (ML): Is a branch of artificial intelligence (AI) that focuses on developing algorithms and models that enable computers to learn from and make predictions or decisions based on data. Unlike traditional programming, where a computer follows explicit instructions, machine learning allows the system to identify patterns and make decisions with minimal human intervention.

Neural Network: Is a computational model inspired by the way biological neural networks in the human brain process information. It is a key component in many machine learning and deep learning systems, particularly those used for tasks like image recognition, natural language processing, and speech recognition.

CNN: Computer Neural Network typically refers to an artificial neural network (ANN) used in computational tasks. These networks are modeled after the neural structure of the human brain and are designed to recognize patterns, learn from data, and make decisions. Here's a basic overview of how they work

DNN: Deep Learning Neural Networks refer to neural networks with multiple layers, known as deep neural networks (DNNs). These networks are designed to model complex patterns and representations in data through a hierarchical learning process.

I. INTRODUCTION

In this fast-pacing world with technology improving daily and turning complex processing huge amounts of data is a crucial but a tedious task. Data analysis has become a huge demand which solely relies on machine learning algorithms [1]. There is constant need for processing units that can compile intensive computation effectively in less time. To exploit them in real time applications they should consume less power and provide maximum throughput. DNN algorithms are computationally intensive, requiring significant processing power for training and inference. Various hardware platforms, such as CPUs, GPUs, and ASICs, FPGAs are used to implement these algorithms. Each platform has unique

characteristics, leading to specific challenges when deploying deep learning models. The journal explores the complexities involved in implementing deep learning algorithms on CPUs, GPUs, ASICs.

This article explores the role of FPGAs in accelerating deep learning tasks. It highlights the advantages of FPGAs in terms of flexibility, energy efficiency, and low-latency processing over other platforms. The survey reviews various architectures and design strategies for implementing neural networks on FPGAs, emphasizing optimizations like quantization and pruning. It also discusses challenges such as limited on-chip memory and the complexity of mapping deep learning models to FPGA hardware, alongside emerging solutions to enhance performance and scalability [2].

II. LITERATURE SURVEY

Machine learning algorithm like DNN is used to compute intensive calculations in very less time with a high degree of accuracy. This is exploited in real-time: medical and space applications for diagnosis, treatment and investigatory purposes respectively. [13]

Many sets of each kind of layer are commonly found in contemporary DNNs. They are all composed of a convolutional layer, an activation layer (also known as a Re LU layer), layers for batch normalization and pooling (to minimize the size of the calculation to prevent overfitting and for the subsequent layer). FC Earlier networks (like Alex Net and VGG) used layers. Later Networks (ResNet, for example) use either none or very few FC levels. In modern DNN most (about 90%) of computation occurs in convolutional layer.[3]

DNN technology due to its complex computation nature is used to assist amputated individuals. But managing its power requirements is practically very challenging while applying it in wearable devices. Implementation of these algorithms on FPGA architecture ensures power efficiency, low complexity. According to this research the results proves an architecture synthesized using Zynq Ultra Scale +FPGA consumes 4.46x less power[4,5].

DNN framework is optimized for low precision training and real time edge computing. While comparing ASIC, GPU, FPGA, FPGAs becomes the more adaptable and apt for portable neural decoding applications that consumes less power. While ASIC provides high performance but followed by exorbitant prices. GPUs provide high memory bandwidth while the energy requirement is huge[6]

Overruling these techniques FPGA architecture is characterized by a) High flexibility, b) Performance per watt c) High degrees of Parallelism which refers to the capability of building a customized hardware circuits that are deeply pipelined and are inherently multithreaded. d)Reduced memory bottlenecks, e) Algorithm-level optimizations, f) Dynamic Reconfiguration g) High level programming models: FPGA design tools are more compatible with high level software practices like OpenCL, CUDA.[7]

The memory bandwidth of CPUs is typically lower, which can become a bottleneck when handling large datasets and model parameters. Contrastingly, FPGAs doesn't depend on external memory fetching which hinders performance.[][]. An architecture synthesized on Xilinx 7 FPGA shows 43x-fold speedup than an Intel i5 CPU [8].

In regards with space applications ML algorithms such as Q-learning and Deep Q learning implementations is constrained by chip size, processing power and radiation hardening. FPGA implementation of advanced learning algorithms enables greater autonomy and adaptability in robotics technology. [8]

Adopting Dark FPGA- a software/hardware codesigned framework which leverages Batch Level Parallelism to accelerate the entire DNN training on single FPGA platform. The CHWB (Channel Height Width Batch) pattern to train data for developing the FPGA architecture optimizes data transfer and processing.The out-turn is an accelerator that performs 10 times faster than CPU and 2.5x more energy efficient.[9]

The advancements in VLSI design enhances performance and efficiency design which enables the optimization of VLSI architectures for implementing the Data Encryption Standard (DES). The advance techniques aid in improving encryption throughput and resource utilization on FPGA platforms, discussing trade-offs between area, speed, and power consumption. It also addresses challenges in achieving high-speed encryption and explores various architectural approaches and FPGA implementation strategies to improve DES efficiency in practical applications.[2]

In practice DNN algorithms are not fully autonomous and, tunable hyper parameters adjustment is

necessary. In Deep learning when complexity of model increases with the presence of many combinations of hyper parameters such as 1. Training iterations,2. Learning rate, 3. Mini batch Size, 4. Number of hidden layers. The act of setting these values by grid or random search and among the most adaptive methods Bayesian optimization is most popular.[10,11]

FPGA use in deep learning, resolves the initial challenges that conventional approaches to artificial intelligence posed under data processing and enhances resource management. The state of art methods of FPGA implementation to build accelerators concentrates on the transition from high precision, energy consuming to low-precision and energy efficient architecture which can be adopted for embedded and real time control system applications. [12,13]

Recent advancements include high-level synthesis tools improving deployment of complex models. Future trends focus on reconfigurable computing and hybrid architectures, addressing limitations like on-chip memory and precision, enabling broader FPGA adoption in deep learning.

III. PROPOSED METHODOLOGY

The vision of this paper is to propose a forward-looking approach that encourages readers to consider the integration of deep learning algorithms, such as Deep Neural Networks (DNN) and Convolutional Neural Networks (CNN), into Field-Programmable Gate Array (FPGA) kits. FPGAs offer unique advantages over other electronic devices, particularly in terms of strict power consumption and higher efficiency, which are crucial when deploying computationally intensive algorithms like DNNs and CNNs. By implementing these deep learning models on FPGAs, we can significantly enhance the efficiency of the resulting outputs.

This proposal highlights the potential of FPGAs to optimize the performance of deep learning applications, making them a compelling alternative to traditional electronic platforms. FPGAs are well-suited for handling the demands of deep learning due to their ability to be customized for specific tasks, which allows for greater control over power usage and performance. This paper aims to demonstrate that integrating deep learning algorithms into FPGAs is not only feasible but also advantageous, offering a pathway to more efficient and powerful computing solutions. Through this approach, we can harness the full potential of deep learning while maintaining energy efficiency and achieving superior results. After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper; use the scroll down window on the left of the MS Word Formatting toolbar.

IV. WHAT IS FPGA?

FPGA stands for Field-Programmable Gate Array. It is an integrated circuit that can be configured by the user or designer after manufacturing, which is why it is called "field-programmable." Unlike traditional processors, where the function is fixed after manufacturing, FPGAs allow designers to program and reprogram the hardware to perform specific tasks, making them highly versatile [14,15].

Key Components of an FPGA:

A. Configurable Logic Blocks (CLBs):

CLBs are the core building blocks of an FPGA, consisting of small units of logic gates like AND, OR, and XOR. They can be configured to perform a variety of logical operations and are typically arranged in a grid pattern. Each CLB can be programmed to execute specific logic functions.

B. Input/Output Blocks (IOBs):

IOBs manage the communication between the FPGA and external devices. They control the flow of data into and out of the FPGA, providing the interface for connecting to peripherals, sensors, and other components.

C. Switching Matrix (Interconnect):

The switching matrix or interconnect is a network of programmable switches that connect CLBs and IOBs. It allows the various components of the FPGA to communicate with each other and ensures that signals are routed correctly within the chip. Clocking Resources: FPGAs contain dedicated clocking resources, including clock generators and phase-locked loops (PLLs), which synchronize the operations of the various components. These resources

ensure that all parts of the FPGA operate in unison and at the correct timing.

D. Memory Blocks:

FPGAs often include embedded memory blocks, such as RAM, which can be used to store data temporarily during processing. These memory blocks are distributed throughout the FPGA and can be configured for different purposes.

E. Digital Signal Processing (DSP) Blocks:
Many FPGAs include specialized DSP blocks designed for high-speed arithmetic operations, such as multiplication and addition. These blocks are particularly useful for signal processing, image processing, and other compute-intensive tasks.

F. Configuration Memory:

This memory stores the configuration data that defines the logic functions and interconnections within the FPGA. The configuration can be updated, allowing the FPGA to be reprogrammed for different tasks as needed.

G. Embedded Processors (optional):

Some modern FPGAs come with embedded processors (such as ARM cores) that allow for a combination of software programmability and hardware reconfigurability, providing a powerful platform for complex applications.[15]

V. COMPLEXITY IN OTHER DEVICES

Implementing deep learning algorithms on traditional electronic devices like CPUs, Microprocessors and modern devices such as GPUs, and ASICs (Application-Specific Integrated Circuits) presents various complexities when compared to using FPGAs.

1. Fixed Architecture vs. Customizability:

a. CPUs and GPUs: These devices have fixed architectures. CPUs are general-purpose processors designed for a wide range of tasks, but they lack the parallel processing capabilities required for deep learning. GPUs, while better suited for parallel processing, have a fixed number of cores and memory bandwidth, which can limit their efficiency for certain deep learning tasks.
b. ASICs: ASICs are designed for specific tasks and can be highly efficient, but they lack flexibility. Once manufactured, the logic of an ASIC cannot be changed, making it difficult to adapt to new deep learning models or algorithms.
c. FPGAs: In contrast, FPGAs offer reconfigurable hardware, allowing developers to customize the architecture for specific deep learning models. This flexibility reduces the complexity of implementing various algorithms, as the FPGA can be reprogrammed to optimize performance for different models.

2. Power Consumption and Efficiency:

a. CPUs: While powerful, CPUs are not energy-efficient for deep learning tasks due to their sequential processing nature. They consume more power when attempting to perform parallel operations, which is a requirement for most deep learning algorithms.
b. GPUs: GPUs are more energy-efficient than CPUs for parallel processing, but they still consume significant power, especially when handling large-scale deep learning models. Managing power consumption while maintaining high performance can be challenging.
c. ASICs: ASICs are energy-efficient because they are designed for specific tasks, but their efficiency comes at the cost of flexibility. Any change in the algorithm or model requires redesigning the entire chip.
d. FPGAs: FPGAs offer a balance between efficiency and flexibility. They allow for customized architectures that can be optimized for power consumption and performance, making them more efficient than CPUs and GPUs for deep learning tasks.

3. Scalability and Parallelism:

a. CPUs: CPUs are limited in terms of scalability for deep learning tasks due to their lower core counts and lack of inherent parallelism. Handling large datasets or complex models can lead to bottlenecks.
b. GPUs: GPUs excel in parallelism, with thousands of cores designed for tasks like matrix multiplications in deep learning. However, managing memory bandwidth and ensuring efficient data transfer between cores can add complexity.
c. ASICs: ASICs can be designed for specific parallel tasks but scaling them for different deep learning models can be complex and costly due to the fixed nature of their design.
d. FPGAs: FPGAs support massive parallelism and can be scaled to meet the specific needs of different deep learning algorithms. Developers can design

custom data paths and parallel processing units within the FPGA, which simplifies the implementation of large-scale deep learning models.

4. Development and Implementation Complexity:

a. CPUs and GPUs: Developing deep learning algorithms on CPUs and GPUs often involves optimizing software code to fit the hardware's capabilities. This process can be complex and time-consuming, especially when trying to achieve high performance.
b. ASICs: ASIC development is highly complex, requiring extensive design, verification, and testing processes. The lack of reconfigurability means any errors or changes require redesigning the chip, adding to the complexity.
c. FPGAs: While FPGA development requires knowledge of hardware description languages (HDLs), the reconfigurability of FPGAs reduces the risk associated with errors or design changes. The ability to test and iterate designs quickly on FPGAs makes the development process less complex compared to ASICs.

5. Cost and Time-to-Market:

a. CPUs and GPUs: While widely available, high-performance CPUs and GPUs can be costly, especially when scaling for large deep learning models. Additionally, optimizing software for these platforms can increase time-to-market.
b. ASICs: ASICs, though efficient, are extremely expensive to design and manufacture, with long development cycles. This makes them less attractive for rapidly evolving fields like deep learning.
c. FPGAs: FPGAs offer a middle ground in terms of cost and time-to-market. They are more cost-effective than ASICs and offer faster development cycles due to their reconfigurability. This reduces the complexity associated with bringing new deep learning solutions to market.

VI. WORKING PRINCIPLE

The working principle of an FPGA (Field-Programmable Gate Array) revolves around its ability to be configured and reconfigured to perform specific digital logic functions. Unlike traditional processors with fixed architectures, FPGAs can be programmed to implement custom logic circuits, allowing them to perform a wide range of tasks. Here's how FPGAs work:

Configuration and Programming:

- *Design Entry:* The process begins with designing the desired logic circuit using a Hardware Description Language (HDL) like VHDL or Verilog. This design describes the behavior of the digital circuit in terms of logic gates and interconnections.
- *Synthesis:* The HDL code is synthesized into a netlist, which is a representation of the logic gates, and their connections needed to implement the design.
- *Mapping: The* netlist is then mapped onto the FPGA's resources, such as Configurable Logic Blocks (CLBs), Input/Output Blocks (IOBs), and interconnects. The synthesis tool assigns specific CLBs and other resources to implement the desired logic.
- *Place and Route:* The mapped design is then placed into the specific locations on the FPGA, and the interconnections (routing) between these blocks are established. This step ensures that the design meets timing and performance constraints.
- *Configuration File Generation:* After place and route, a configuration file (bitstream) is generated. This file contains the binary data needed to program the FPGA.

VII. SUMMARY

Field-Programmable Gate Arrays (FPGAs) have emerged as a versatile and efficient platform for implementing deep learning algorithms, offering distinct advantages over traditional CPUs, GPUs, and even ASICs. The key strength of FPGAs lies in their reconfigurability, allowing for the hardware to be precisely tailored to the requirements of specific deep learning models. This capability enables significant parallelism, enhancing computational efficiency and reducing energy consumption—benefits that are particularly critical in power-sensitive applications such as edge computing. Additionally, FPGAs provide low-latency processing, making them ideal for real-time applications like autonomous systems. While ASICs may offer superior performance in speed and power efficiency, their fixed architecture post-manufacture limits adaptability. FPGAs, on the other hand, can be reprogrammed to accommodate evolving neural network models, providing a scalable and cost-effective solution for deep learning implementations.

VIII. CONCLUSION

Tabular Comparison (CPU vs GPU vs ASIC vs FPGA):

Parameter	CPU	GPU	ASIC	FPGA	Why FPGA is best
Architecture	General-purpose, sequential	Parallel processing with many cores	Application-specific, highly optimized	Reconfigurable logic blocks	Can be reprogrammed to fit specific deep learning tasks
Performance	Moderate performance for general tasks	High performance in parallel tasks	Very high for specific tasks	High performance with customization	Offers near-ASIC performance with the flexibility to adapt
Flexibility	Highly flexible	Less flexible, specialized for parallelism	No flexibility	Highly flexible, reprogrammable	Allows real-time adjustments to algorithms and hardware
Energy Efficiency	Moderate energy consumption	High energy consumption due to parallelism	Very energy-efficient	Efficient and customizable	Can be optimized for both performance and power consumption
Latency	High latency	Moderate latency	Very low latency	Low latency, can be optimized	Ideal for real-time deep learning applications

Parameter	CPU	GPU	ASIC	FPGA	Why FPGA is best
Development Complexity	Simple	Moderate, requires parallel programming	High, custom design	Moderate, but powerful	Provides a balance between complexity and high-performance gains
Cost	Low to moderate	Moderate to high	High, due to custom design	Cost-effective for medium-scale production	Lower cost for production runs compared to ASICs
Scalability	Limited scalability	High scalability	Not scalable, fixed	Scalable and adaptable	Can be scaled to different levels of deployment and complexity
Use Case	General-purpose tasks	Parallel processing for training	Specific tasks like AI inference	Customizable for a wide range of tasks	Suitable for both prototyping and production
Availability	Widely available	Available, but specialized	Custom made, limited availability	Available and customizable	Wide range of FPGAs available with extensive design tools

VIII . REFERENCES

[1] Y. Tu, S. Sadiq, Y. Tao, M. -L. Shyu and S. -C. Chen, "A Power Efficient Neural Network Implementation on Heterogeneous FPGA and GPU Devices," 2019 IEEE 20th International Conference on Information Reuse and Integration for Data Science (IRI), Los Angeles, CA, USA, 2019, pp. 193-199, doi: 10.1109/IRI.2019.00040

[2] A. Shawahna, S. M. Sait and A. El-Maleh, "FPGA-Based Accelerators of Deep Learning Networks for Learning and Classification: A Review," in IEEE Access, vol. 7, pp. 7823-7859, 2019, doi: 10.1109/ACCESS.2018.2890150.

[3] Nurvitadhi, Eriko & Subhaschandra, Suchit & Boudoukh, Guy & Venkatesh, Ganesh & Sim, Jaewoong & Marr, Debbie & Huang, Randy & Hock, Jason & Liew, Yeong & Srivatsan, Krishnan & Moss, Duncan. (2017). Can FPGAs Beat GPUs in Accelerating Next-Generation Deep Neural Networks? 5-14. 10.1145/3020078.3021740.

[4] Hassan, Omiya. University of Missouri - Columbia ProQuest Dissertations & Theses, 2023. 30488506.

[5] A. Nimbekar, Y. V. S. Dinesh, A. Gautam, V. Hunsigida, A. R. Nali and A. Acharyya, "Reconfigurable VLSI Design Architecture for Deep Learning Established Forelimb and Hindlimb Gesture Recognition for Rehabilitation Application," in IEEE Access, vol. 11, pp. 70061-70070, 2023, doi: 10.1109/ACCESS.2023.3293422.

[6] A. Shawahna, S. M. Sait and A. El-Maleh, "FPGA-Based Accelerators of Deep Learning Networks for Learning and Classification: A Review," in IEEE Access, vol. 7, pp. 7823-7859, 2019, doi: 10.1109/ACCESS.2018.2890150.

[7] Deep Learning on FPGAs: Past, Present, and Future https://doi.org/10.48550/arXiv.1803.05900

[8] P. R. Gankidi and J. Thangavelautham, "FPGA architecture for deep learning and its application to planetary robotics," 2017 IEEE Aerospace Conference, Big Sky, MT, USA, 2017, pp. 1-9, doi: 10.1109/AERO.2017.7943929

[9] C. Luo, M. -K. Sit, H. Fan, S. Liu, W. Luk and C. Guo, "Towards Efficient Deep Neural Network Training by FPGA-Based Batch-Level Parallelism," 2019 IEEE 27th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), San Diego, CA, USA, 2019, pp. 45-52, doi: 10.1109/FCCM.2019.00016.

[10] J. Bergstra and Y. Bengio. Random search for hyper-parameter optimization. The Journal of Machine Learning Research, 13(1):281–305, 2012.

[11] J. Snoek, H. Larochelle, and R. P. Adams. Practical bayesian optimization of machine learning algorithms. In Advances in neural information processing systems, pages 2951–2959, 2012. K. Elissa, “Title of paper if known,” unpublished.

[12] Charles Rajesh Kumar J., Vinod Kumar D., Baskar D., Mary Arunsi B., Jenova R., M.A. Majid,VLSI design and implementation of High-performance Binary-weighted convolutional artificial neural networks for embedded vision-based Internet of Things (IoT), Procedia Computer Science, Volume 163,2019, Pages 639-647, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2019.12.145.

[13] Morteza Babaee Altman, Wenbin Wan, Amineh Sadat Hosseini, Saber Arabi Nowdeh, Masoumeh Alizadeh,Machine learning algorithms for FPGA Implementation in biomedical engineering

applications:Areview,Heliyon, Volume10, Issue4,2024, e26652, ISSN2405-8440, https://doi.org/10.1016/j.heliyon.2024.e26652

[14] A Survey of FPGA Based Deep Learning Accelerators: Challenges and Opportunities Teng Wang, Chao Wang, Xuehai Zhou, Huaping Chen, https://doi.org/10.48550/arXiv.1901.04988

[15] Kaiyuan Guo, Shulin Zeng, Jincheng Yu, Yu Wang, and Huazhong Yang. 2019. [DL] A Survey of FPGA-based Neural Network Inference Accelerators. ACM Trans. Reconfigurable Technol. Syst. 12, 1, Article 2 (March 2019), 26 pages. https://doi.org/10.1145/3289185