Accelerating Neural Networks on FPGAs: An Overview

Moiz Zaheer Malik
B.Eng. Electronic Engineering
Hochschule Hamm-Lippstadt
Lippstadt, Germany
moiz-zaheer.malik@stud.hshl.de

***Abstract*—Neural networks are used in many smart systems today. But they need a lot of computing power to work fast. CPUs are often slow, and GPUs use a lot of power. This is why FPGAs are becoming popular. FPGAs can be changed and programmed to do a specific task very well. In this report, we explain how FPGAs can help speed up neural networks.**

I. Introduction

As digital technologies continue to advance and the availability of reliable and trustworthy data increases, artificial intelligence (AI) and deep learning techniques are gaining widespread popularity [24]. These technologies have demonstrated remarkable effectiveness in solving complex problems that were once considered beyond computational reach [24].

However, deep learning models particularly convolutional neural networks (CNNs) require extremely high computational power and memory bandwidth [24]. Such demands are difficult for traditional central processing units (CPUs) to meet efficiently, often resulting in limited performance [24]. To overcome these challenges, hardware accelerators such as application specific integrated circuits (ASICs) and field-programmable gate arrays (FPGAs) have emerged as key solutions for improving the processing speed of AI applications, including CNNs [24].

Among these, FPGAs are widely used to accelerate deep learning tasks because they excel at executing multiple operations in parallel and offer significantly better power efficiency compared to general purpose processors (GPPs) [24].

To make results more accurate in real time like in self driving cars and robots CNNs have to get bigger by adding more layers [39]. This makes them more powerful but also much heavier to process, with billions of calculations and millions of parameters that need strong computing power for training and testing [34, 4, 24].

Because of such massive computational requirements it create massive challenges for traditional general purpose processors (GPPs) [24]. As a result, hardware accelerators such as ASICs, GPUs, and FPGAs are increasingly used to improve the performance and outputrate of CNNs [24].

But still, there are some drawbacks between these platforms. GPUs, for instance, are widely used for CNN acceleration in both training and inference due to their high memory bandwidth and efficient parallel computation capabilities [53, 23, 49, 24]. But yet, GPUs high power consumption makes them less suitable for cloud based systems or battery powered CNN applications [15, 24].

In contrast, FPGAs have emerged as a preferred platform because they offer a balance between performance and energy efficiency [24]. Experimental results show that FPGAs deliver higher power efficiency (performance per watt) [24], than other accelerators, despite having relatively limited I/O bandwidth and computing resources compared to GPUs. They can still achieve moderate performance with significantly lower power consumption [30, 24].

ASICs can achieve high throughput by customizing memory hierarchies and dedicating resources to specific tasks [10, 24] . However, they suffer from long development cycles, high costs, and low flexibility in deep learning applications [18, 9, 24]. As an alternative, FPGA based accelerators provide high throughput at a reasonable cost, with low power consumption and reconfigurability [47, 37].

Another major advantage of modern FPGAs is the availability of High Level Synthesis (HLS) tools that allow developers to program hardware using C or C++, significantly simplifying the design process and reducing development time for FPGA based accelerators [15, 24].

II. What is an FPGA?

Field Programmable Gate Arrays (FPGAs) are special programmable chips they are off-the-shelf programmable devices which means FPGAs can be bought ready made and one just has to configure them for their own purpose, They can customized to perform many different hardware functions [24]. Unlike normal processors that follow fixed instructions, an FPGA can be configured to perform like any digtal circuit depending how its programmed. this is what makes it very usefull and flexible for application that require high speed and custom design [50].

The figure 1 Shows the basic structure of an FPGA. Inside an FPGA there are thousands of small building blocks called configuranle Logic Blocks (CLBs) that contain look Up Tables (LUTs) and Flip Flops (FFs) [24]. These are connected through programmable interconnection network, allowing signals to move freely between them. Around the edges of the

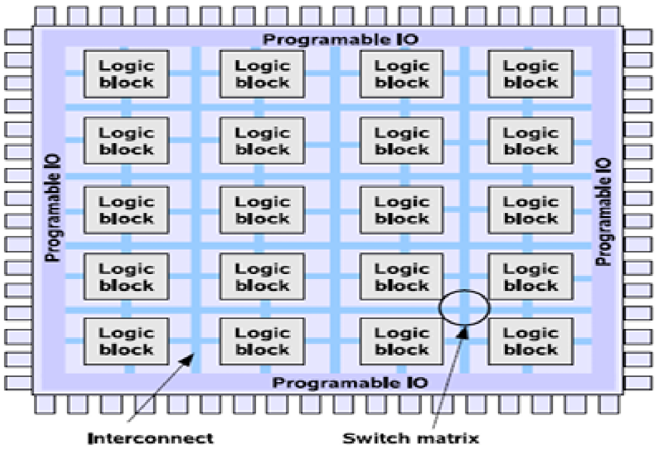

Fig. 1: FPGA Basic Structure [24, 50]

chip, there are Input/Output (I/O) cells that connect the FPGA to external devices [22, 24].

FPGAs also include Digital Signals Processing (DSP) Block for fast Math operations, Block RAM (BRAM) for temporary data storage, and clock managment unites to keep every thing synchronized. Some even include high speed communication interfaces, making them great for data intensive applications [48, 28, 24].

Because of such structure, FPGAs can handle a large amount of parallel processing, meaning they can run many small task at once insted of running them one by one [24]. This makes the ideal for deep learning and neural network accelertion, where thousands of operations happen every second [11]. They are also ver power efficient, giving high performance for much less energy compared to CPUs or GPU [28, 24].

Another very strong feature of FPGAs is partial reconfiguration, which allows part of the chip to be reprogrammed while the rest keeps running and this is somthing very helpfull for deep learing models where one layer can be reloaded while another is still processing [24].

Recently, Programming models like OpenCL and tools such as High Level Synthesis (HLS) have made FPGA development easier [24]. They let developers use familiar languages such as C and C++ to desing hardware, reducing desing time and cost [31, 44, 35, 52].

III. What is a Neural Network?

Deep learning is becoming very popular and one of the important tools for solving complex problem because of the avilabilty of big data and also because of its computational capaibilties [36]. the domains the deep learing is to solve problem include image recognition [27], speech processing [2, 14, 20] , natural language processing [6], language translation [8], and autonomousvehicles [29, 36].

The popular algorithmic approch for deep learning Convolutional neural networks is imerging as the best in most of the domains. CNN can be dedvided into two different parts

1) Training: During training, a neural network learns from a large number of datasets by adjusting its internal parameters, known as weights [36]. A deep learning engineer designs the network architecture deciding how many layers it has, what operations each layer performs, and how the layers are connected [36]. These weights control how strongly one neuron influences the next, determining what kind of features the layers detect, such as edges, colors, or shapes in images.

The goal of training is to find the best possible values for those weights. This is done using a mathematical optimization method called Stochastic Gradient Descent (SGD) [36].

The Training happens in three main steps. Forward propagation where the input goes through the network layer by layer to produce an output. next step is Error calculation where the network checks how much different the output is from the correct soulution and the difference here is called Error or Loss. Then in the Back Propagation the error is sent back through all the layers to update weights slightly to reduce the error next time

2) Inference: After the completion of training the, the networks is put into the real world. Where it takes new input data and makes predictions or classificatons[36]. This is the stage where a trained neural network is used to make predictions. Unlike training, it only performs forward propagation data passes through the layers to produce an output, without updating any weights. However, inference can still require massive computational power, especially for deep networks with hundreds of layers or for large inputs like high definition video [36]. Because many applications such as self driving cars and smart devices run on limited energy, achieving high energy efficiency during inference is crucial.

Neural networks use activation functions to introduce nonlinearity into the model [36]. One of the most common is the Rectified Linear Unit (ReLU), which sets all negative values to zero. The resulting values, called activations, represent the output of a layer and serve as the input to the next layer [36]. In practice, a large portion of these activations often between 50% and 70% become zero, which reduces the amount of meaningful data passed forward and influences later hardware optimization strategies [36].

To illustrate the scale of modern neural networks, Table I summarizes three well known architectures. Each contains millions of parameters and requires billions of multiplication operations for a single inference, highlighting the need for efficient hardware implementations [36].

TABLE I: Characteristics of popular CNN architectures [36].

Network	Conv. Layers	Weights (MB)	Multiplies (B)
AlexNet	5	1.73	0.69
GoogLeNet	54	1.32	1.10
VGGNet	13	4.49	15.3

Convolutional Neural Networks (CNNs) are composed of multiple layers arranged in a sequence, forming a cascade of feature extraction and transformation steps [29]. These layers include convolutional layers that apply small filters (1×1, 3×3, or 5×5) that slide across the image and multiply with small

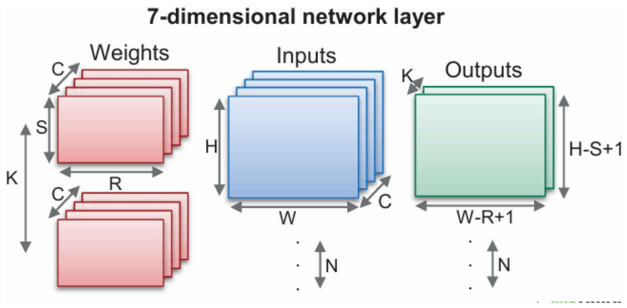

Fig. 2: Basic structure of a convolutional layer showing input activations, learned filter weights, and resulting output activations [36]

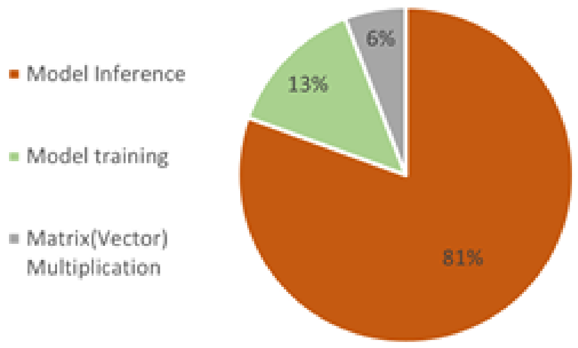

Fig. 3: Distribution of FPGA based ML research showing that inference dominates due to low latency and energy efficient characteristics of FPGAs [1]

parts and each filter detects specific patterns like vertical lines, texture or edges, non linear activation layers such as ReLU that introduce nonlinearity, and pooling layers that reduce the spatial resolution while preserving important information [36]. The weights that are learned during training, enabling the network to detect edges, shapes, and complex visual features [36]. In deeper networks, fully connected layers near the end combine all extracted features to perform classification. The output of one layer, known as an activation map, serves as the input for the next layer, allowing hierarchical feature learning from simple to complex representations [36].

As illustrated in Fig. 2, each convolutional layer in a neural network applies a small filter (set of weights) across the input feature maps to generate output activations. This operation extracts spatial features such as edges or textures and passes them to deeper layers for higher level representation [36].

IV. Why Use FPGA for Neural Networks?

One of the important subsets of artificial intelligence, Machine Learning (ML), focuses on algorithms that learn from large datasets to predict outcomes and perform tasks autonomously without explicit programming [36]. Recent research has achieved remarkable progress in domains such as image segmentation [43], object classification [26, 21] and detection [3], data classification [41], natural language processing (NLP) [12], edge computing [32], large scale scientific computing [16], and even for circuit design and optimization [42]. To achieve higher accuracy, ML models have become increasingly deep and complex, often containing redundant parameters that significantly increase computational and memory requirements [1, 19].

As a result, ML inference and training demand massive processing power and memory bandwidth [1]. Traditional computing platforms such as Central Processing Units (CPUs) and Graphics Processing Units (GPUs) are widely used for ML but have notable limitations. CPUs are optimized for general purpose, mostly sequential tasks, making them inefficient for highly parallel ML workloads [1]. GPUs, while powerful for parallel operations, consume substantial energy and generate heat, posing challenges for portable or energy constrained devices [1].

Therefore, custom hardware architectures specifically designed for ML algorithms are becoming essential [1]. Field Programmable Gate Arrays (FPGAs) are particularly suited for this role because of their reconfigurable architecture. Their internal logic blocks, interconnections, and memory organization can be modified to match the structure of a neural network, even during runtime, allowing them to adapt dynamically to changing workloads [1]. This adaptability, combined with fine-grained parallelism and energy efficiency, makes FPGAs an ideal choice for both small scale edge computing and large scale cloud acceleration of ML tasks. Moreover, FPGAs are well suited for real time neural network inference because their hardware can be customized to the model structure, reducing latency and improving performance [1].

Another key advantage of FPGAs in neural network acceleration is their ability to deliver low latency inference [1]. Many real time applications, such as autonomous driving and video analytics, require response times within milliseconds [1]. FPGAs are good at this because they can run many things at the same time using parallel circuits and pipelined data paths that keep the work flowing smoothly [1, 33, 38, 40]. They also move data inside the chip very efficiently, which saves time by reducing how often they need to access slower external memory [1, 13, 54, 51]. This characteristic explains why a majority of FPGA based ML studies (approximately 81%) focus on inference rather than training, as illustrated in Fig. 3 [1].

Energy efficiency is another important reason to use FPGAs. They use less power because the hardware can be designed exactly for what the algorithm needs, so there is no wasted work or extra data movement [1, 5, 25, 45, 17]. Their streaming dataflow design and built in power control help reduce energy use even more [1]. They can also use smaller data types, like 8-bit numbers instead of 32-bit, and reuse memory inside the chip instead of reading it from outside. This makes them great for devices that run on batteries or for data centers where saving energy is important [1]. Compared to GPUs, FPGAs are more energy efficient, and unlike ASICs, they can still be reprogrammed for different models [46, 55, 1].

Neural networks also fit well with the structure of FPGAs. A lot of neural network work involves math like convolutions and matrix multiplications, and FPGAs can run many of these calculations in parallel using their DSP blocks. This makes them fast, efficient, and flexible for different types of ML models [1, 7].

Overall, FPGAs are powerful because they combine speed, flexibility, and low power use. They can handle real time tasks, save energy, and adapt to new models easily, making them a very good choice for accelerating neural networks.

V. How FPGAs Accelerate Neural Networks

VI. Related Work and Current Results

A. *Surveyed FPGA Accelerators (Quantitative)*

B. *Representative Case Studies*

C. *Discussion: Trends and Takeaways*

D. *Architecture Examples and Case Studies*

E. *Common FPGA Acceleration Techniques*

1) *Parallelism:*

2) *Pipelining:*

3) *Quantization:*

4) *Data Reuse and Local Memory:*

5) *Loop Unrolling and Tiling:*

6) *Custom Precision and Fixed Point Arithmetic:*

7) *Streaming Architectures:*

VII. Advantages of FPGA for AI

VIII. Challenges

IX. Conclusion

References

[1] H. Ali et al. "A Survey on FPGA-Based Accelerator for Machine Learning". In: *2021 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*. IEEE, 2021, pp. 145–150. DOI: 10.1109/ICAIIC51459.2021.9415204.

[2] Dario Amodei et al. "Deep Speech 2: End-To-End Speech Recognition in English and Mandarin". In: *arXiv preprint arXiv:1512.02595* (2015). URL: https://arxiv.org/abs/1512.02595.

[3] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao. "YOLOv4: Optimal Speed and Accuracy of Object Detection". In: *arXiv preprint arXiv:2004.10934* (2020).

[4] T. M. Chilimbi et al. "Project Adam: Building an efficient and scalable deep learning training system". In: *Proceedings of the 11th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*. Vol. 14. taken from FPGA-Based Accelerators of Deep Learning Networks for Learning and Classification: A Review. 2014, pp. 571–582.

[5] P. Colangelo et al. "Exploration of low numeric precision deep learning inference using Intel FPGAs". In: *2018 IEEE 26th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*. 2018, pp. 73–80.

[6] Ronan Collobert et al. "Natural Language Processing (Almost) From Scratch". In: *arXiv preprint arXiv:1103.0398* (2011). URL: https://arxiv.org/abs/1103.0398.

[7] D. Diamantopoulos and C. Hagleitner. "A system-level transprecision FPGA accelerator for BLSTM using on-chip memory reshaping". In: *2018 International Conference on Field-Programmable Technology (FPT)*. IEEE, 2018, pp. 338–341.

[8] Gregory Diamos et al. "Persistent RNNs: Stashing Recurrent Weights On-Chip". In: *Proceedings of the International Conference on Machine Learning (ICML)*. 2016.

[9] L. Du et al. "A Reconfigurable Streaming Deep Convolutional Neural Network Accelerator for Internet of Things". In: *IEEE Transactions on Circuits and Systems I: Regular Papers* 65.1 (2018). taken from FPGA-Based Accelerators of Deep Learning Networks for Learning and Classification: A Review, pp. 198–208. DOI: 10.1109/TCSI.2017.2705055.

[10] H. Esmaeilzadeh et al. "Neural acceleration for general-purpose approximate programs". In: *45th Annual IEEE/ACM International Symposium on Microarchitecture*. taken from FPGA-Based Accelerators of Deep Learning Networks for Learning and Classification: A Review. 2012, pp. 449–460. DOI: 10.1109/MICRO.2012.52.

[11] C. Farabet et al. "Large-Scale FPGA-Based Convolutional Networks". In: *Scaling up Machine Learning: Parallel and Distributed Approaches*. Ed. by R. Bekkerman, M. Bilenko, and J. Langford. Cambridge, U.K.: Cambridge University Press, 2011, pp. 399–419.

[12] S. B. Goldberg et al. "Machine Learning and Natural Language Processing in Psychotherapy Research: Alliance as Example Use Case". In: *Journal of Counseling Psychology* 67.4 (2020), p. 438.

[13] Y. Gong et al. "N3H-Core: Neuron-designed neural network accelerator via FPGA-based heterogeneous computing cores". In: *Proceedings of the 2022 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*. 2022, pp. 112–122.

[14] Alex Graves and Jürgen Schmidhuber. "Framewise Phoneme Classification With Bidirectional LSTM and Other Neural Network Architectures". In: *Neural Networks* (2005).

[15] K. Guo et al. "Angel-Eye: A Complete Design Flow for Mapping CNN onto Embedded FPGA". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 37.1 (2018). taken from FPGA-Based Accelerators of Deep Learning Networks for

Learning and Classification: A Review, pp. 35–47. DOI: 10.1109/TCAD.2017.2760518.

[16] E. Haghighat and R. Juanes. "SciANN: A Keras/TensorFlow Wrapper for Scientific Computations and Physics-Informed Deep Learning Using Artificial Neural Networks". In: *Computer Methods in Applied Mechanics and Engineering* 373 (2021), p. 113552.

[17] M. Hall and V. Betz. "From TensorFlow graphs to LUTs and wires: Automated sparse and physically aware CNN hardware generation". In: *2020 Int.*

[18] S. Han et al. "EIE: Efficient Inference Engine on Compressed Deep Neural Network". In: *43rd Annual International Symposium on Computer Architecture (ISCA)*. taken from FPGA-Based Accelerators of Deep Learning Networks for Learning and Classification: A Review. 2016, pp. 243–254. DOI: 10.1109/ISCA.2016.30.

[19] S. Han et al. "Learning Both Weights and Connections for Efficient Neural Network". In: *Advances in Neural Information Processing Systems*. Vol. 28. 2015.

[20] Awni Hannun et al. "Deep Speech: Scaling Up End-To-End Speech Recognition". In: *arXiv preprint arXiv:1412.5567* (2014). URL: https://arxiv.org/abs/1412.5567.

[21] K. He et al. "Deep Residual Learning for Image Recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 770–778.

[22] M. C. Herbordt et al. "Computing Models for FPGA-Based Accelerators". In: *Computer Science and Engineering* 10.6 (2008), pp. 35–45.

[23] G. Hinton et al. "Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups". In: *IEEE Signal Processing Magazine* 29.6 (2012). taken from FPGA-Based Accelerators of Deep Learning Networks for Learning and Classification: A Review, pp. 82–97. DOI: 10.1109/MSP.2012.2205597.

[24] Asmaa Ibrahim, Ashraf Attia, and Nayera Hussein. "FPGA-Based Accelerators of Deep Learning Networks for Learning and Classification: A Review". In: *IEEE Access* 8 (2020), pp. 134824–134849. DOI: 10.1109/ACCESS.2020.3009390.

[25] G. G. Ko et al. "Accelerating Bayesian inference on structured graphs using parallel Gibbs sampling". In: *2019 29th International Conference on Field Programmable Logic and Applications (FPL)*. 2019, pp. 159–165.

[26] A. Krizhevsky, I. Sutskever, and G. E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*. Vol. 25. 2012.

[27] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Proceedings of the International Conference on Neural Information Processing Systems (NIPS)*. 2012.

[28] G. Lacey, G. W. Taylor, and S. Areibi. *Deep Learning on FPGAs: Past, Present, and Future*. arXiv preprint arXiv:1602.04283. 2016. URL: https://arxiv.org/abs/1602.04283.

[29] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep Learning". In: *Nature* 521 (2015), pp. 436–444.

[30] J. Misra and I. Saha. "Artificial Neural Networks in Hardware: A Survey of Two Decades of Progress". In: *Neurocomputing* 74.1–3 (2010). taken from FPGA-Based Accelerators of Deep Learning Networks for Learning and Classification: A Review, pp. 239–255. DOI: 10.1016/j.neucom.2010.03.021.

[31] A. Munshi. "The OpenCL Specification". In: *Proceedings of the IEEE Hot Chips 21 Symposium (HCS)*. 2009, pp. 13–14.

[32] M. S. Murshed et al. "Resource-Aware On-Device Deep Learning for Supermarket Hazard Detection". In: *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2020, pp. 871–876.

[33] H. Nakahara et al. "A lightweight YOLOv2: A binarized CNN with a parallel support vector regression for an FPGA". In: *Proceedings of the 2018 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*. 2018, pp. 31–40.

[34] O. Nomura and T. Morie. "Projection-field-type VLSI convolutional neural networks using merged/mixed analog-digital approach". In: *Proceedings of the International Conference on Neural Information Processing*. taken from FPGA-Based Accelerators of Deep Learning Networks for Learning and Classification: A Review. Berlin, Germany: Springer, 2007, pp. 1081–1090.

[35] A. R. Omondi and J. C. Rajapakse. *FPGA Implementations of Neural Networks*. Vol. 365. Boston, MA, USA: Springer, 2006.

[36] A. Parashar et al. "SCNN: An Accelerator for Compressed-Sparse Convolutional Neural Networks". In: *Proceedings of the 44th Annual International Symposium on Computer Architecture (ISCA)*. ACM, 2017, pp. 27–40. DOI: 10.1145/3079856.3080254.

[37] A. Putnam et al. "A Reconfigurable Fabric for Accelerating Large-Scale Datacenter Services". In: *ACM SIGARCH Computer Architecture News* 42.3 (2014). taken from FPGA-Based Accelerators of Deep Learning Networks for Learning and Classification: A Review, pp. 13–24. DOI: 10.1145/2678373.2665678.

[38] Z. Que et al. "A reconfigurable multithreaded accelerator for recurrent neural networks". In: *2020 International Conference on Field-Programmable Technology (ICFPT)*. 2020, pp. 20–28.

[39] O. Russakovsky et al. "ImageNet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision* 115.3 (2015). taken from FPGA-Based Accelerators of Deep Learning Networks for Learning and Classification: A Review, pp. 211–252. DOI: 10.1007/s11263-015-0816-y.

[40] V. Rybalkin and N. Wehn. “When massive GPU parallelism ain’t enough: A novel hardware architecture of 2D-LSTM neural network”. In: *FPGA 2020: ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*. 2020, pp. 111–121.

[41] R. Saravanan and P. Sujatha. “A State of Art Techniques on Machine Learning Algorithms: A Perspective of Supervised Learning Approaches in Data Classification”. In: *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*. IEEE, 2018, pp. 945–949.

[42] K. Settaluri et al. “AutoCkt: Deep Reinforcement Learning of Analog Circuit Designs”. In: *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2020, pp. 490–495.

[43] B. A. Skourt, A. El Hassani, and A. Majda. “Lung CT Image Segmentation Using Deep Neural Networks”. In: *Procedia Computer Science*. Vol. 127. 2018, pp. 109–113.

[44] J. E. Stone, D. Gohara, and G. Shi. “OpenCL: A Parallel Programming Standard for Heterogeneous Computing Systems”. In: *Computer Science and Engineering* 12.3 (2010), pp. 66–73.

[45] M. Sun et al. “Hardware-friendly acceleration for deep neural networks with microstructured compression”. In: *2022 IEEE 30th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*. 2022.

[46] Y. Umuroglu et al. “FINN: A Framework for Fast, Scalable Binarized Neural Network Inference”. In: *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*. 2017, pp. 65–74.

[47] W. Vanderbauwhede and K. Benkrid. *High-Performance Computing Using FPGAs*. taken from FPGA-Based Accelerators of Deep Learning Networks for Learning and Classification: A Review. New York, NY, USA: Springer, 2013.

[48] B. S. C. Varma, K. Paul, and M. Balakrishnan. *Architecture Exploration of FPGA-Based Accelerators for Bioinformatics Applications*. Singapore: Springer, 2016.

[49] A. Vasudevan, A. Anderson, and D. Gregg. “Parallel multi-channel convolution using general matrix multiplication”. In: *2017 IEEE 28th International Conference on Application-specific Systems, Architectures and Processors (ASAP)*. taken from FPGA-Based Accelerators of Deep Learning Networks for Learning and Classification: A Review. 2017, pp. 19–24. DOI: 10.1109/ASAP.2017.7995251.

[50] J. Villasenor and W. H. Mangione-Smith. “Configurable Computing”. In: *Scientific American* 276.6 (1997), pp. 66–71.

[51] M. P. Véstias et al. “Hybrid dot-product calculation for convolutional neural networks in FPGA”. In: *2019 29th International Conference on Field Programmable Logic and Applications (FPL)*. 2019, pp. 350–353.

[52] H. M. Waidyasooriya, M. Hariyama, and K. Uchiyama. *Design of FPGA-Based Computing Systems with OpenCL*. Cham, Switzerland: Springer, 2018.

[53] A. Yazdanbakhsh et al. “Neural acceleration for GPU throughput processors”. In: *Proceedings of the 48th International Symposium on Microarchitecture (MICRO)*. taken from FPGA-Based Accelerators of Deep Learning Networks for Learning and Classification: A Review. 2015, pp. 482–493.

[54] M. Yingchang and Q. Liu. “M4BRAM: Mixed-precision matrix-matrix multiplication in FPGA block RAMs”. In: *2023 International Conference on Field Programmable Technology (ICFPT)*. 2023.

[55] H. Zhou et al. “Enabling Low Latency Edge Inference of Deep Neural Networks on FPGAs”. In: *2019 IEEE 27th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*. IEEE, 2019, pp. 1–8.