INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2023, VOL. 69, NO. 3, PP. 599-604
Manuscript received September 10, 2022; revised July, 2023. DOI: 10.24425/ijet.2023.146513

FPGA Implementation of Neural Nets

B A Sujatha Kumari, Sudarshan Patil Kulkarni, and C G Sinchana

***Abstract*—The field programmable gate array (FPGA) is used to build an artificial neural network in hardware. Architecture fora digital system is devised to execute a feed-forward multilayer neural network. ANN and CNN are very commonly used architectures. Verilog is utilized to describe the designed architecture. For the computation of certain tasks, a neural network's distributed architecture structure makes it potentially efficient. The same features make neural nets suitable for application in VLSI technology. For the hardware of a neural network, a single neuron must be effectively implemented (NN). Reprogrammable computer systems based on FPGAs are useful for hardware implementations of neural networks.**

***Keywords*—artificial neural network; Spartan-6; field programmable gate arrays (FPGAs); convolutional neural network**

I. INTRODUCTION

ANArtificial neural network and Convolutional network is a form of intelligent processing system that aims to replicate the composition and functionality of the human brain. In recent days it is a major topic in a broad range of fields and an effective in problem-solving situations. ANNs have been used to resolve issues where conventional methods have failed in the past, such as speech synthesis, recognition, image processing ,coding, pattern recognition and classification, forecasting power load, interpreting and predicting financial trends for the financial markets, generating compositestructures, and processing model construction, monitoring [1].The FPGA's flexibility and adaptability allow it to meet the needs of reiterative learning processes, weight adjustment, and even architectural reconfiguration. FPGAs cannot incorporate infinite amounts of logical resources into a single package, hence optimizations are necessary [5]. The implementation of ANN digital computer systems has been relatively limited, with examples representing recent research available from it. The latest developments in the systematic concept allowfor the use of large ANNs on a single scheduled gateway device (FPGA) device. A small improvement in component production technique, where the data capacity of electronic components increases every 18 months, is the main cause[3]. ANNs are biologically motivated which needs DSPs and microprocessors which are not suitable for comparable systems. ASICs and VLSIs can be utilized to design fully compatible modules, but creating such devices is expensive and time consuming. To offer ANN a design effect that is just suitable for a particular use. FPGAs offer flexible designs, costsavings, and design cycles in addition to compatibility [4].

II. RELATED WORKS

The development and use of Floating Point Multipliers and Adders (FPMA)-based Artificial Neural Networks (ANN) with Field Programmable Gate Array (FPGA). On the Spartan II FPGA, several nine neurons were implemented at a speed of 13 M bits/sec. It is possible to employ multiple FPGAs oneach layer to create a larger ANN on an FPGA. [3].

The implementation of essential ANN in FPGA is described. The FPGA implementation could use parallelism to accelerate processing time in comparison to software. In addition, hardware implementation can use fewer resources than CPU/GPU. Additionally, the system is intended for identifying objects with minimal hardware resources and low power usage as partof the work. [2].

Taxonomy for dividing up ANN for FPGA implementations is discussed. Design challenges and several implementation strategies are discussed. Furthermore, future research trends are presented. A parallel and distributed network of straightforward nonlinear processing units connected in a layered configuration make up an artificial neural network (ANN). Three computational characteristics highly associated with ANNsthey are parallelism, modularity, and dynamic adaptation [6]. A hardware implementation of an ANN using an FPGA is proposed. The FPGA was identified due to its more affordable costs for a single implementation. Creating a hardware solution that enabled the direct use of weight training is typically created in a software environment with a much higher resolution than typically obtained in FPGAs implementation was the main objective [10].

FPGA-based design and implementation of an ANN are discussed. The FPGA was identified due to its more affordable costs for a single implementation. The objective was to create a hardware solution that allowed the direct use of weights, which is typically prepared in a software environment with resolutions that are significantly higher than any of those often achieved in FPGA implementation [11]. The creationof a hardware architecture that uses a Convolutional Neural Network (CNN) framework that is adaptable and enables the configuration and execution of different neural network models. The new HW-CNN module was tested and the CNN framework was developed using the High-Level Synthesis (HLS) language on an Altera Stratix V FPGA integrated witha Teresa DE-5-Net board [13].

Using the FPGA hardware ZYBO Z7, a convolution neural network based on the Lenet-5 model was implemented. The three-layer convolution layer, the two-layer down sampling layer, and the two-layer fully connected layer make up the Lenet-5 network module. The typical FPGA clock frequency is merely a few hundred MHz, and it may often be clocked more rapidly than general-purpose CPUs [20].

The evaluation of various journal articles on numerous subjects indicated that designing hardware accelerators for neural

Authors are with Sri Jayachamarajendra College of Engineering,JSS Science and Technology University, Mysore, India (e-mail: sujathakumari@sjce.ac.in, sudarshan_pk@sjce.ac.in, sinchanacg03@gmail.com).

networks is significant as it greatly improves computational efficiency. The effectiveness and efficiency of machine learning models would rise with the usage of neural networks. Some researchers also suggest utilizing the FPGA to implement ANN or CNN in hardware. The parallelism, flexibility, power consumption, reconfigurability, and lower costs for a single implementation were thought about while selecting the FPGA.

III. MATERIALS AND METHODS

A. Vivado ISE

High-level synthesis and system on a chip development are now possible with the Xilinx software package Vivado Design Suite. It is used to synthesize and analyze designs published in the hardware description language (HDL). Vivado offers a total rebuild and rethinking of the whole design cycle in comparison to ISE. Vivado contains an integrated logic simulator, just like later versions of ISE. Another development of Vivado is high-level synthesis. It converts C code into programmable circuits via a tool chain. The Vivado High-Level Synthesis compiler enables programs written in C, C++, and System C to be directly targeted at Xilinx devices without manually writing RTL. The capability of Vivado HLS to support C++ classes, modules, functions, and operator overloading has been proven.

B. NIST standard Data set

In the Modified National Institute of Standards (MNIST), 10,000 test instances compensate for the dataset of handwritten digits, which has 60,000 training sets. It is a modest portion ofa larger dataset that NIST has made available. The characters have been size-normalized and placed in a fixed-size image. The source NIST (bi-level) dark images were scaled downto fit inside a 20x20 pixel box while preserving their aspect ratio. The resulting photos have grey levels because of the anti-aliasing strategy used by the normalization algorithm. Theimages were centered in a 28x28 image by finding the centroidof the pixels and translating the image to place this point inthe center of the 28x28 field.

IV. PROPOSED METHOD

Figure 1 depicts the block diagram of ANN implementation. This paper involves a handwritten dataset as input and which is converted into vectors of 784 lines and saved into a txt file. Once the neuron gets trained the architecture will be builtwith the desired number of layers.The used inputs are 784and two hidden layers and one output layer with 10 neurons. With the help of the Xilinx Vivado, the simulation tool will be synthesized to the desired number of LUTs, flip-flops, and Power consumption.

The block diagram CNN implementation is shown in figure 2 CNN architecture uses the same data which is used in the ANN architecture. As we are using the convolution, first convolute the image of size 200*72 and with the help of bias values, The model is trained. Synthesis is done using Xilinx Vivado simulation tool to get the desired number of LUTs, flip-flops, and Power consumption.

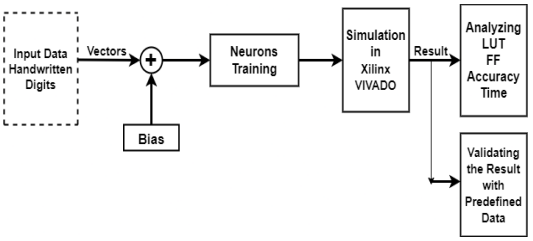

Fig. 1. Block Diagram for ANN

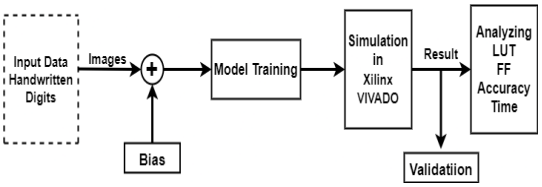

Fig. 2. Block Diagram for CNN

V. IMPLEMENTATION

A. ANN Implementation

ANN is based on the structure and function of the biological neural network. The input layer with 784 neurons, because there have been used 28*28 pixel value images and each pixel is converted into 16 bits.

A. Input Layer: The layer has been implemented with 784 neuron inputs, because the data set image had 28*28 pixels and converted into binary for each pixel and made into 784 lines, and given input to all the 784 neurons on the first layer of the architecture.

B. Three hidden Layers: Each hidden layer for 30 neurons and sent to each neuron of the input layer output, andin a similar fashion sent to another layer also.

C. At the output layer, there are 10 neurons and with helpof maximum function can determine the output from the neuron.

D. The Expected output and output of the neuron obtained from the architecture and used as comparision factor. The serial, parallel, and Distributed Architectures used for ANN implementation is shown in the figure 3.

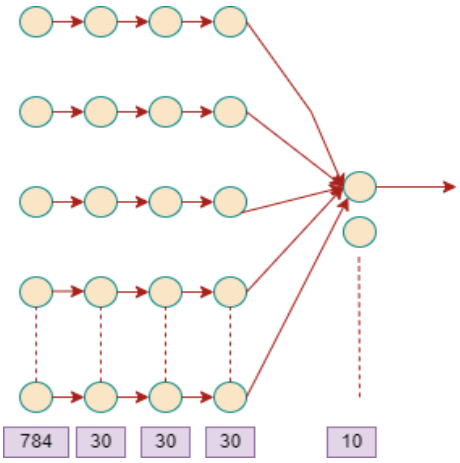

Fig. 3. Serial Architecture

B. *CNN Implementation:*

The architecture of CNN is shown in figure 4. CNN's architecture can be summarized in the following manner. The network's input is an RGB patch of 32 × 32 pixels. There are three pooling layers in addition to three convolutional layers. The convolutional layer has a 5 x 5 kernel with 2-pixel padding. The input patch is 32 × 32 in size and features 3 RBG channels. 32×32 feature mappings are provided by the first convolutional layer.

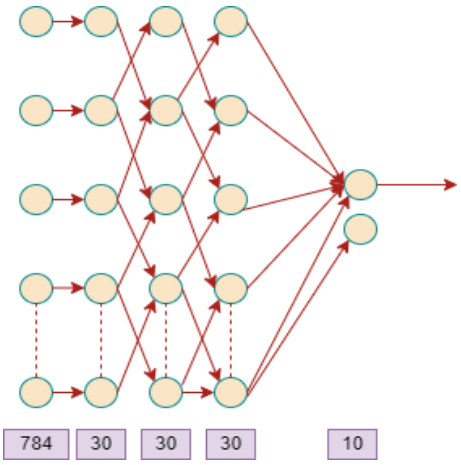

Fig. 4. Parallel Architecture

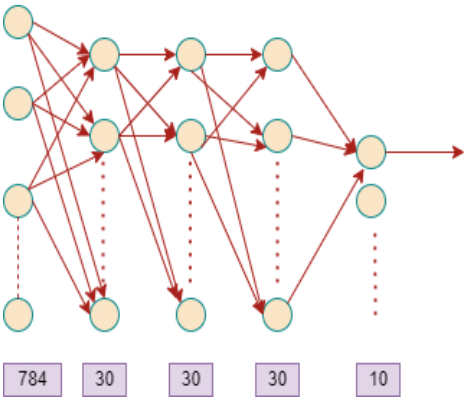

Fig. 5. Distributed architecture

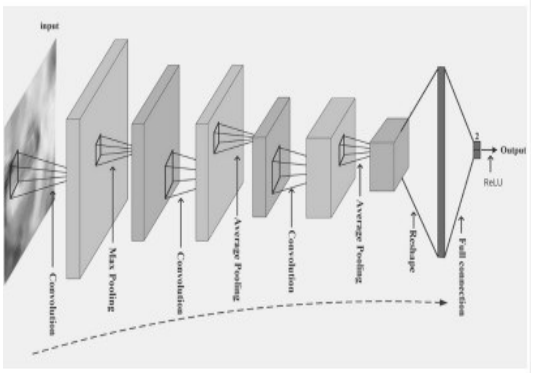

Fig. 6. Block diagram of CNN Architecture Implementation

After max pooling, the 32 feature maps were down-sampled into 16 × 16. Finally, 125 feature maps with a 5 x 5 size are formed employing two more convolutional layers and average pooling layers. The features are combined into a feature vector, which is subsequently fed to the ReLU and full connection layers for classification. Local response normalizing layers and rectified linear unit layers are also included in CNN but are not shown for simplicity.

VI. RESULT

The Serial, Parallel and distributed architecture are implemented for ANN and the obtained results are discussed in the following section.

A. *Distributed Architecture:*

Distributed Architecture which permits the completion of an intricate image recognition assignment on a number of resources.The method significantly reduces the amount of memory and processing power needed by each devices tostore and process ANN model. Designers create several ANN models and use the models for identifying human posture as the case study. In this architecture. It permits the completionof an intricate image recognition assignment on a number of resources used one input layer consists of 784 neurons and 2 hidden layers of 30 neurons each, and an output layer of 10 neurons. The architecture was designed for the handwritten recognition dataset. In this neuron architecture, each neuron will take input from all the neurons of the previous layer.For example, in the 2nd hidden layer, the 1st neuron will receive the output of 30 neurons of the 1st hidden layer. Resource utilization is shown in the below figure. LUT is 12.95% used, Flipflop is 6.23%,BRAM is 10.71% DSAP is 72.72%,IO is 52.50%,BUFG is 3.13% used.

TABLE I
RESOURCE UTILIZATION OF DISTRIBUTED ARCHITECTURE OBTAINED IN SYNTHESIS

Resource	Utilization	Available	Utilization [%]
LUT	6887	53200	12.95
FF	6653	106400	6.25
BRAM	15	140	10.71
DSAP	160	220	72.73
IO	105	200	52.50
BUFG	1	32	3.13

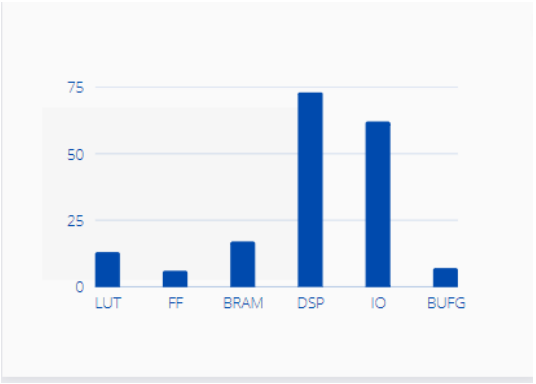

Fig. 7. Resource utilization graph of Distributed architecture

B. *Serial Architecture*

In this architecture, the input layer consists of 784 neurons and 2 hidden layers of 30 neurons each, and an output layer of 10 neurons. The architecture was also designed for the hand-written recognition dataset. In this neuron architecture, each neuron will take input from all the neurons of the previous layer but with some time delay. For example, in the 2nd hiddenlayer, the 1st neuron will receive the output of 30 neuronsof the 1st hidden layer. Instead of going to the next neuronof the 3rd hidden layer, it will forward the output to all the10 neurons of the output layer. Once it's all neuron receives the data then it goes to the 2nd neuron of the second hidden layer. As a result, the amount of information transmitted to each neuron will be more.

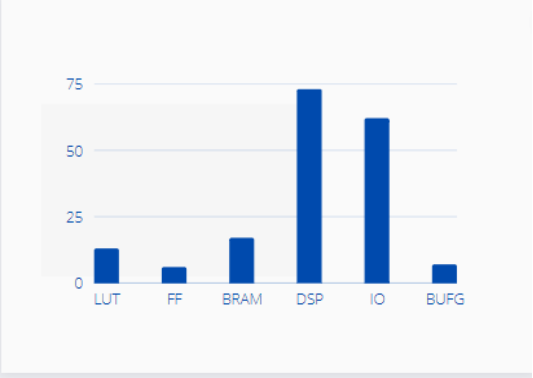

Fig. 8. Resource utilization graph of Serial architecture

TABLE II
RESOURCE UTILIZATION OF SERIAL ARCHITECTURE OBTAINED IN SYNTHESIS

Resource	Utilization	Available	Utilization [%]
LUT	6907	53200	12.98
FF	6457	106400	6.07
BRAM	15	140	10.71
DSAP	160	220	72.73
IO	105	200	52.50
BUFG	1	32	3.13

C. *Parallel Architecture*

In this architecture, the input layer consists of 784 neurons and 2 hidden layers of 30 neurons each, and an output layer of 10 neurons. The architecture was designed for the handwritten recognition dataset. In this neuron architecture, each neuron will take input from all the neurons of the previous layer. For example, in the 2nd hidden layer, the 1st neuron will receive the output of the 1st neurons of the 1st hidden layer. Similarly,the 2nd neuron of the second hidden layer will receive the2nd neuron output. As a result, the number of information transmitted to each neuron will be more and takes more lookup table. Obtained accuracy is around 82percent.

TABLE III
RESOURCE UTILIZATION OF PARALLEL ARCHITECTURE OBTAINED IN SYNTHESIS

Resource	Utilization	Available	Utilization [%]
LUT	1296	53200	2.44
FF	1505	106400	1.41
BRAM	0.50	140	0.36
DSAP	26	220	11.82
IO	105	200	52.50
BUFG	1	32	3.13

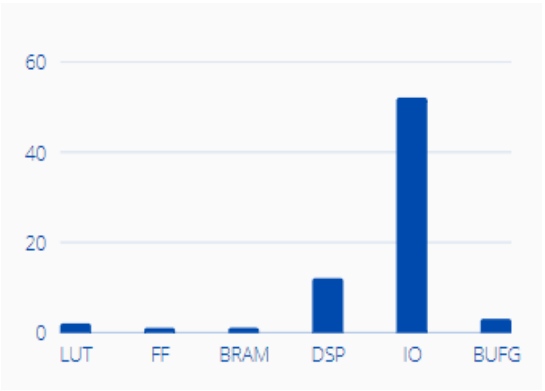

Fig. 9. Resource utilization graph of Parallel architecture

TABLE IV
RESOURCE UTILIZATION OF DISTRIBUTED ARCHITECTURE OBTAINED IN IMPLEMENTATION

Resource	Utilization	Available	Utilization [%]
LUT	6887	53200	12.95
FF	6653	106400	6.25
BRAM	15	140	10.71
DSAP	160	220	72.73
IO	105	200	52.50
BUFG	1	32	3.13

TABLE V
RESOURCE UTILIZATION OF SERIAL ARCHITECTURE OBTAINED IN IMPLEMENTATION

Resource	Utilization	Available	Utilization [%]
LUT	6907	53200	12.98
FF	6457	106400	6.07
BRAM	15	140	10.71
DSAP	160	220	72.73
IO	105	200	52.50
BUFG	1	32	3.13

TABLE VI
RESOURCE UTILIZATION OF PARALLEL ARCHITECTURE OBTAINED IN IMPLEMENTATION

Resource	Utilization	Available	Utilization [%]
LUT	1296	53200	2.44
FF	1505	106400	1.41
BRAM	0.50	140	0.36
DSAP	26	220	11.82
IO	105	200	52.50
BUFG	1	32	3.13

D. *Synthesis Results of CNN*

The synthesis results for the CNN are found to be 1697 LUTs and 910 Flip- flops which is occupying 1% of total available LUTs and Flip-flops.

TABLE VII
RESOURCE UTILIZATION OF CNN

Resource	Estimation	Available	Utilization [%]
LUT	1697	134600	1.26
FF	910	267600	0.34
IO	357	500	71.40
BUFG	1	32	3.13

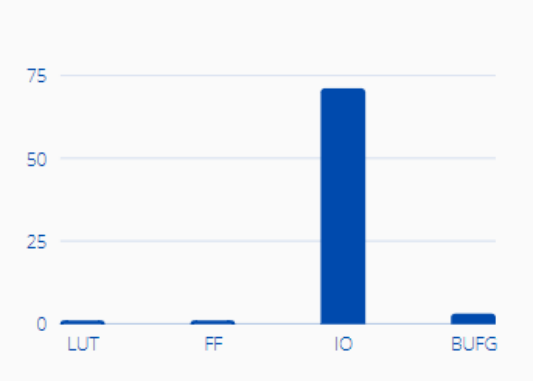

Fig. 10. Resource utilization graph of CNN architecture

E. *Implementation of CNN*

The model was trained for a handwritten digit recognition dataset having an image size of 200*72.

TABLE VIII
SYNTHESIS RESULT OF CNN

Resource	Estimation	Available	Utilzation [%]
LUT	1697	134600	1.26
FF	910	267600	0.34
BRAM	9	365	2.47
IO	357	500	71.40
BUFG	1	32	3.13

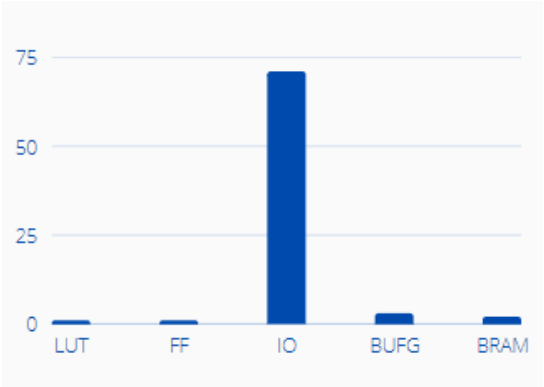

Fig. 11. Resource utilization graph of Distributed architecture

F. *Comparison of ANN and CNN Architectures*

The comparison of the architectures are shown in the table IX Considering the power requirements the distributed architecture takes around 248.273 W, where series architecturetakes 243.273 and Parallel was found out to be less which is about 28W. The architecture performs a better fitting to the image data set due to the reduction in the number of param- eters involved and reusability of weights. So this drastically reduces the actual usage of neurons in the real scenarios and actual implementation.

TABLE IX
COMPARSION OF ARCHITECTURE

	Distributed	Serial	Parallel
LUT	6805	6917	1296
FF	6625	6456	1505
Power [watt]	248.273	243.273.	27.43
Total time [ns]	218	2.20	480
Accuracy [%]	92	87	82

CONCLUSION

The ANN and CNN implementation on FPGA is done with the help of neuron architecture. Four layers are imple- mented and analyzed on FPGA and the results are compared. Using Distributed architecture in ANN, an accuracy of 92percent is obtained and CNN has given accuracy of 89.42 percent. The various other key considerations is the size, parameter validity, and the number of interlayer connectionsof multipliers. The first primarily outlines the area resources needed, while the second outlines the routing. The arithmetic operations are not the bottleneck, and it is possible to explore massive parallelism in the convolutional trees. FPGAs offer enough memory resources to implement the feature maps. The multilayer approach enables the implementation of neural netswith a larger number of hidden layers.

REFERENCES

[1] Jihong Liu, Deqin Liang, “A Survey of FPGA-Based Hardware Implementation of ANNs”, School of Information Science and Engineering Northeastern University, Shenyang-110004, China-2005. Communications, vol. 25, no. 5, pp. 10-15, October 2018, https://doi.org/10.1109/MWC.2018.1800049

[2] Shuai Li, Ken Choi, “Artificial Neural Network Implementation in FPGA: A Case Study”, Department of Electrical and Computer En- gineering, Illi- nois Institute of Technology, USA/Yunsik Lee, School of ECE, UNIST, Ulsan, Korea- ISOCC 2016. https://doi.org/10.1109/ISOCC.2016.7799795

[3] PDr. Reza Raeisi , Armin Kabir, “Implementation of Artificial Neural Network on FPGA”, Indiana State University, Indiana., American Soci- ety for Engineering Education, Illinois-Indiana and North Central Joint Section Conference IPFW (Mar-2006).

[4] Esraa Zeki Mohammed and Haitham Kareem Ali, “Hardware Implemen- tation of Artificial Neural Network Using Field Programmable Gate Array”, International Journal of Computer Theory and Engineering, Vol. 5, No. 5, 2013. https://doi.org/10.7763/IJCTE.2013.V5.795

[5] Philippe Dondon, Julien Carvalho, R´emi Gardere, Paul Lahalle, Georgi Tsenov and Valeri Mladenov, “Implementation of a Feed-forward Arti- ficial Neural Network in VHDL on FPGA”, 12th Symposium on Neural Network Applications in Electrical Engineering (NEUREL 2014), Mare, Romania (2019).

[6] Jihan Zhu and Peter Sutton, “FPGA Implementations of Neural Net- works A Survey of a Decade of Progress”, School of Information Technology and Electrical Engineering, The University of Queensland, Brisbane, Queensland 4072, Australia

[7] Marcin Pietras, “Hardware conversion of neural networks simulation models for Neural Processing Accelerator implemented as FPGA-based SoC”, Com- puter Science and Information Technology West Pomera- nian University of Technology, ZUT Szczecin, Poland (2018).

[8] Saima Kanwal , Arslan Yousaf , Maria Imtiaz , Jalil Abbas , Arslan Ali, “Survey paper on Advanced Equipment Execution of ANN for FPGA”, Computer Engineering and Intelligent Systems Vol.9, No.7, (2019).

[9] S. Oniga, A. Tisan, D. Mic, A. Buchman and A. Vida-Ratiu, "Optimizing FPGA Implementation of FeedForward Neural Networks", Electronic and Computer Engineering Department, North University, Baia

[10] Pedro Ferreira, Pedro Ribeiro, Ana Antunes, and Fernando Morgado Dias, "Artificial Neural Networks Processor – A Hardware Implementation Using a FPGA" – (2018).

[11] Hardik H. Makwana , Dharmesh J. Shah , Priyesh P. Gandhi, "FPGA Implementation of Artificial Neural Network", International Journal of Emerg- ing Technology and Advanced Engineering (2004).

[12] Suhap Sahin, Yasar Becerikli, and Suleyman Yazici , "Neural Network Implementation in Hardware Using FPGAs", Department of Computer Eng., Kocaeli University, Izmit, Turkey, Vol 12, 2018.

[13] Marco Bettoni, Gianvito Urgese, Yuki Kobayashi, Enrico Macii, and Andrea Acquaviva, "A Convolutional Neural Network Fully Implemented on FPGA for Embedded Platforms", first New Generation of CAS (2017).

[14] Yongmei Zhou, Jingfei Jiang, "An FPGA-based Accelerator Implementation for Deep Convolutional Neural Network", 4th International Conference on Computer Science and Network Technology (ICCSNT 2015).

[15] Yasmeen Farouk, Sherine Rady, "Optimizing MRI Registration using Software/Hardware Co-Design Model on FPGA", International Journal of Innovative Technology and Exploring Engineering, Vol 10, Issue 12 (IJITEE) 2020.

[16] Yuchen Yao, Qinghua Duan, Zhiqian Zhang, Jiabao Gao, Jian Wang, Meng Yang, "A FPGA-based Hardware Accelerator for Multiple Convolutional Neural Networks", State Key Laboratory of ASIC System, Fudan University, Shanghai 2018.

[17] Yufeng Hao, "A General Neural Network Hardware Architecture on FPGA", University of Birmingham, 2018.

[18] Fasih Ud Din , Farrukh Tuo XieChun, ZhangZhihua Wang, Fellow, "Optimization for Efficient Hardware Implementation of CNN on FPGA", IEEE international conference on integrated circuits and technologies and applications (2018).

[19] Mohammad Samragh, Mohammad Ghasemzadeh, and Farinaz Koushanfar, "Customizing Neural Networks for Efficient FPGA Implementation", IEEE 25th Annual International Symposium on Field-Programmable Custom Computing Machines 2018. https://doi.org/10.1109/FCCM.2017.43

[20] Dai Rongshi, Tang Yongming , "Accelerator Implementation of Lenet-Convolution Neural Network Based on FPGA with HLS", 2019 3rd International Conference on Circuits, System and Simulation.

[21] Leandro D. Medus, Taras Iakymchuk, Jose V. Frances-Villora, Manuel Bataller-Mompeán, Alfredo Rosado-Muñoz,"A Novel Systolic Parallel Hardware Architecture for the FPGA Acceleration of Feedforward Neural Networks", IEEE Access,vol 2920885, 2019. https://doi.org/10.1109/ACCESS.2019.2920885

[22] M. Zhu, Q. Kuang, J. Lin, Q. Luo, C. Yang and M. Liu, "A Z Structure Convolutional Neural Network Implemented by FPGA in Deep Learning,"44th Annual Conference of the IEEE Industrial Electronics Society, pp. 2677-2682,2018.

[23] H. O. Ahmed, M. Ghoneima and M. Dessouky, "Concurrent MACunit design using VHDL for deep learning networks on FPGA," IEEE Symposium on Computer Applications Industrial Electronics ISCAIE, pp. 31-36, 2018. https://doi.org/10.1109/ISCAIE.2018.8405440

[24] M. Hailesellasie, S. R. Hasan, F. Khalid, F. A. Wad and M. Shafique, "FPGA-Based Convolutional Neural Network Architecture with Reduced Parameter Requirements," IEEE International Symposium on Circuits and Systems , pp. 1-5, 2018.

[25] L. Bai, Y. Lyu and X. Huang, "A Unified Hardware Architecture for Convolutions and Deconvolutions in CNN," IEEE International Symposium on Circuits and Systems (ISCAS), pp. 1-5,2020.

[26] S. -P. Pan, Z. Li, Y. -J. Huang and W. -C. Lin, "FPGA realization of activation function for neural network," 7th International Symposium on Next Generation Electronics , pp. 1-2, 2018.

[27] Y. -C. Ling, H. -H. Chin, H. -I. Wu and R. -S. Tsay, "DesigningA Compact Convolutional Neural Network Processor on Embedded FPGAs," IEEE Global Conference on Artificial Intelligence and Internet of Things (GCAIoT), pp. 1-7,2020.

[28] C. Crema et al., "Embedded platform-based system for early detection of Alzheimer disease through transcranial magnetic stimulation," IEEE Sensors Applications Symposium (SAS), pp. 1-6, 2018. https://doi.org/10.1109/SAS.2018.8336774

[29] S. Ivanov, S. Stankov and T. Nenov, "FPGA Based Neural Networks for Characters Recognition," 20th International Symposium on Electrical Apparatus and Technologies (SIELA), pp. 1-3,2018.

[30] Yufei Ma, Yu Cao, Fellow, Sarma Vrudhula, and Jae-sun Seo,"Optimizing the Convolution Operation to Accelerate Deep Neural Networks on FPGA", IEEE Transactions On Very Large Scale Integration (Vlsi) Systems,2018. https://doi.org/10.1109/TVLSI.2018.2815603

[31] Sachin Nayak, Shweta Vincent, Sumathi K, Om Prakash Kumar, and Sameena Pathan," An Ensemble of Statistical Metadata and CNN Classification of Class Imbalanced Skin Lesion Data", Intl Journal of Electronics and Telecommunications, 2022, vol. 68, no. 2, pp. 251-257 https://doi.org/10.24425/ijet.2022.139875