IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 11, No. 4, December 2022, pp. 1362~1372
ISSN: 2252-8938, DOI: 10.11591/ijai.v11.i4.pp1362-1372

Reduced hardware requirements of deep neural network for breast cancer diagnosis

Yasmine M. Tabra[1], Furat N. Tawfeeq[2]
[1]Information and Communication Department, college of Information Engineering, Al Nahrain University, Baghdad, Iraq
[2]Website Division, University of Baghdad, Baghdad, Iraq

Article Info

Article history:

Received Sep 29, 2021
Revised Jun 27, 2022
Accepted Jul 26, 2022

Keywords:

Breast cancer
Deep neural networks
Field programmable gate array
Very hardware description language
Xilinx system generator

ABSTRACT

Identifying breast cancer utilizing artificial intelligence technologies is valuable and has a great influence on the early detection of diseases. It also can save humanity by giving them a better chance to be treated in the earlier stages of cancer. During the last decade, deep neural networks (DNN) and machine learning (ML) systems have been widely used by almost every segment in medical centers due to their accurate identification and recognition of diseases, especially when trained using many datasets/samples. in this paper, a proposed two hidden layers DNN with a reduction in the number of additions and multiplications in each neuron. The number of bits and binary points of inputs and weights can be changed using the mask configuration on each subsystem to futher reduce the hardware requirements. The DNN was designed using a system generator and implemented using very hardware description language (VHDL). The system achievments outcomes the superior's accuracy rate of approximately 99.6 percent in distinguishing bengin from malignant tissue. Also, the hardware resources were reduced by 30 percent from works of literature with an error rate of 7e-4 when using the Kintex-7 xc7k325t-3fbg676 board.

Corresponding Author:

Yasmine M. Tabra
Information and Communication Department, College of Information Engineering, Al Nahrain University
Baghdad, Iraq
Email: yasminetabra@gmail.com

1. INTRODUCTION

Mammography is the traditional way for early detection of breast cancer, which increases the patient's chance to beat cancer [1]. Another way is to use computerized methods under two conditions: correctly and timely. The hardware implementation of deep neural network (DNN) must ensure the efficient diagnosis of breast cancer with minimum possible hardware requirement on the field programmable gate array (FPGA) [2]. Xilinx company provides tools to work with programmable hardware using Xilinx system generator (XSG) to implement the DNN [3]–[6]. XSG will convert a block diagram implemented in Simulink into a very hardware description language (VHDL) that can be used to program an FPGA board [3], [7]–[9]. The performance of the DNN classification can be evaluated using accuracy tests. The familiar statistics used are true positive (TP), false positive (FP), true negative (TN), false negative (FN) terms. TP is the number of 'positive' classes that are correctly classified as positive. FP is the number of 'positive' classes that are incorrectly classified as positive and should be in the negative class. TN is the number of 'negative' classes that are correctly classified as negative. Finally, FN refers to the number of 'negative' classes that are classified as positive [9], [10]. These statistics are used to calculate some common performance metrics such as accuracy and precision [11]:

$$Accuracy = \frac{TP+TN}{P+N} \tag{1}$$

$$Precision = \frac{TP}{TP+FP} \tag{2}$$

To determine the correctness rate of benign and malignant detection, the receiver operating characteristic (ROC) curve method was used [12], [13]. ROC graphs are useful to organize and visualize classifiers' performance. ROC is used in decision-making, machine learning (ML), and data mining (DM) in medical research. The tradeoff between TP and FP is declared using the ROC curve [11], [14]. DNN uses multiple hidden layers, where the first layer represents the input layer, the final layer represents the output layer, and all the intermediate layers represent the hidden layers. Each layer contains nodes called neuron that passes the information through layers, starting from a neuron in the input layer passing through hidden layer nodes, and finally the output layer nodes [15]–[19]. The product of inputs and weights are added to bais at each node. Then the output is passed through the activation function. This function works based on thresholding. When the node value or set of nodes passes the threshold, the nodes' values are passed to the next layer. The equation that describes the proposed DNN model in this paper:

$$y_{1,2} = \mathrm{F}\underbrace{\left(\sum_{k=1}^{2} \mathrm{w_k}\, F \underbrace{\left(\sum_{j=1}^{8} \mathrm{w_j}\, \mathrm{F} \underbrace{\left(\sum_{i=1}^{10} \overbrace{\mathrm{I_i}}^{input} . \mathrm{w_i} + bias1 \right)}_{hidden\ layer\ 1} + bias2 \right)}_{hidden\ layer\ 2} + bias3 \right)}_{output\ layer} \tag{3}$$

One of the activation functions used with NN is the sigmoid function [4], [20]:

$$f(x) = c + bx + ax^2 \tag{4}$$

Where $c = \frac{1}{1+e^{-y_0}}, \quad b = \frac{1}{1+e^{-y_1}}, \quad a = \frac{1}{1+e^{-y_2}}$

With $\vec{y} = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \end{bmatrix}, where\ \ \vec{y} = \vec{V}/\vec{F}$

And $\vec{y} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 2 & 2 \\ 1 & 4 & 4 \end{bmatrix}$. Finally, $\vec{F} = \begin{bmatrix} 0 \\ 2 \\ 4 \end{bmatrix}$

To benefit work similar to human nature in making decisions based on expert opinion, DNN should be trained on a large number of training data [18]. Xilinx provided a Simulink library that works the same as the other elements of Simulink by providing many blocks doing different functions. VHDL or Verilog code is generated from these blocks by implementing the hardware architecture using resources integrated with Simulink and also .m files MATLAB. Once completed, XSG automatically generates the VHDL code [15]–[17]. The integrated synthesis environment (ISE) tool is used with XSG to generate the bitstream code. Figure 1 represents these steps.

Several researchers used neural network (NN) as a tool to detect breast cancer instead of traditional biological ways for efficient and timely detection. Aguiar *et.al.* [3] built a single neuron artificial neural netwok (ANN) to test the newly designed hyperbolic tangent activation function. The results showed that using 16 bits is more convenient than other tested solutions. Khodja *et.al.* [4] implemented a single neuron ANN on FPGA using XSG. Sigmoid function designed using the polynomial form. The results showed that ANN occupied 63% of the used Spartan3 board. Sepandi *et. al.* [9] created an ANN model to accurately predict breast cancer risk in patients using datasets that include monographic results and other risk factors. Due to the limitation of registered data, their ANN can support only decision-making. Ayer *et.al.* [12] evaluated the effect of using large datasets in training ANN to discriminate between benign and malignant diseases. The authors used feedforward NN with three layers having 1,000 neurons in hidden layers. The results showed success in detecting benign accurately and predicting risks of breast cancer abnormality. Janghel *et.al.* [21] developed an ANN system to classify breast cancer. Four models of NN have been implemented: back propagation algorithm

(BPA), radial basis function networks (RBF), learning vector quantization (LVQ), and competitive learning network. The Back propagation model with a single hidden layer having 25 neurons showed the best results.

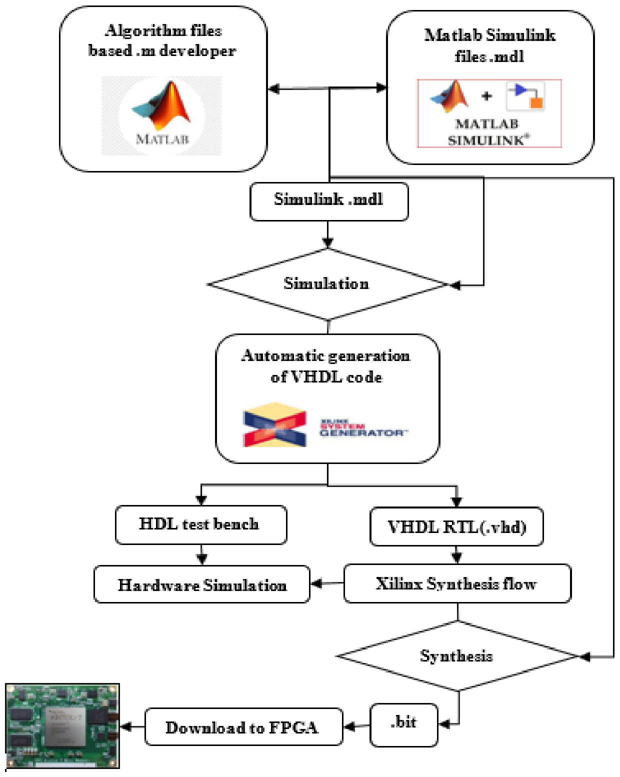


Figure 1. Xilinx system generator design steps

Singhal and Pareek [22] created a backpropagation ANN with 6 neurons in the hidden layer. It can early predict breast cancer using the Wisconsin Breast Cancer (Diagnostic) Dataset. This database contains 372 records. The ANN was implemented using C-language. Shukla *et.al.* [23] trained three types of ANN namely, BPA, RBF, and LVQ network. BPA showed the best accuracy of other networks with 40 hidden neurons, while RBFN showed the least training time. Bharati *et.al.* [24] reviewed the advantages and limitations of ANN literature models for breast cancer diagnosis via mammography. Then described the preprocessing methods used with ANN. Desai and Shah [25] reviewed the employment of several NN types for breast cancer diagnosis. A Comparison was made to identify the most accurate detection and showed that CNN provides higher accuracy than ANN. Tisan *et.al.* [26] designed five different activation functions then a single neuron ANN was implemented using XSG for each of these activation functions. The resource consumption of the implementations was calculated and compared to show that lookup table activation provides minimum used gates.

However, these previously mentioned works tried to employ NNs to detect breast cancer with reduced hardware components of FPGA. In this paper, the designed DNN detection accuracy was higher, and the hardware resources were less than those in the literature. This reduction was due to reducing the multiplications and additions processes per single neuron from NxN to (N-1) x (N-1) addition and multiplication. A different number of bits and floating points were tested to allow us to increase the reduction of the used hardware component.

The dataset used to design a NN to classify cancers as either benign or malignant depending on the characteristics of sample biopsies. Dataset has 699 cases 241 for malignant cases and 458 for benign cases. Thus containing 34.5% benign and 65.5% malignant. The dataset contains two variables:

i) cancerInputs: with a 9x699 matrix defining nine attributes of 699 biopsies. These are: (clump thickness, uniformity of cell size, uniformity of cell shape, marginal adhesion, single epithelial, cell size, bare nuclei, bland chomatin, normal nucleoli, and mitoses)
ii) cancerTargets: with a 2x966 matrix where each column indicates a correct category with a one in either element 1 or element 2. These are:(benign and malignant)

Only 70 percent of the input samples are used in training, while the remaining 15 percent in testing, and 15 percent for validation.

2. RESEARCH METHOD

In this paper, several steps need to be done to design a DNN with high detection accuracy and reduced hardware requirements by following several steps. Figure 2 shows these design steps. Each block in the flowchart represents a single step in the design and implementation of the proposed DNN. These steps are described in detail in the following steps.

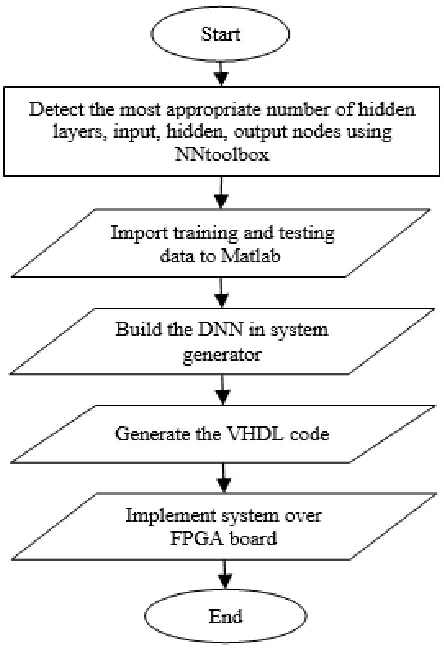
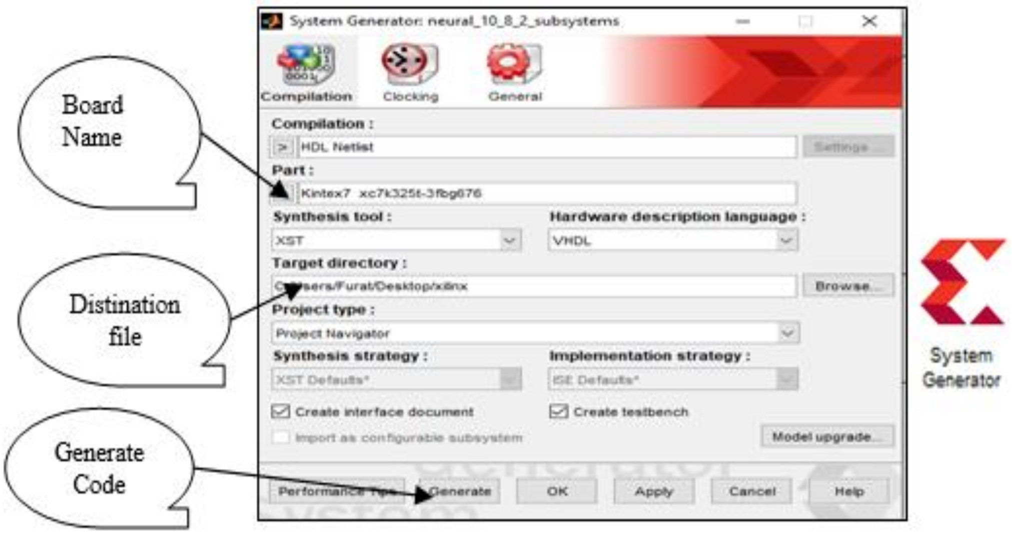

Start
Detect the most appropriate number of hidden layers, input, hidden, output nodes using NNtoolbox
Import training and testing data to Matlab
Build the DNN in system generator
Generate the VHDL code
Implement system over FPGA board
End

Figure 2. Flowchart of proposed work

i) Step one
- Use MATLAB NN toolbox to detect the most suitable number of nodes in each layer, the number of hidden layers, and the optimum weight used in each neuron. NN simulator results showed the best results for a DNN with two fully connected hidden layers. The DNN input layer contains nine nodes according to the number of features in the dataset used for training. The first hidden layer contains ten neurons, and the second hidden layer contains eight neurons. Finally, the output layer with two neurons produces two outputs; one represents benign and the other for malignant detection.
- Import the network weights to Matlab workspace to be used later in system generator.

ii) Step two
- Import the training datasets file ‘cancerInputs’ and ‘cancerTargets’ to MATLAB workspace, to test the DNN design accuracy in system generator.

iii) Step three:
- In MATLAB Simulink window, add system generator block and fill-in the required information such as FPGA board name and model, target working directory as in Figure 3. Also include the generate button which is used to convert the model to VHDL code.
- Use the ‘from workspace’ block to read data from the workspace. All inputs are grouped in the ‘Inputs’ subsystems, as shown in Figure 4. All the inputs and weights are defined as ‘signed’ numbers with fixed-point precision. The reduction in the number of bits and binary points will affect the implementation of DNN design by reducing the required resources. Also, the use of fixed points precision will reduce the expenses of implementing floating points in hardware. All the subsystems use a mask configuration, the parameters of these masks are the number of bits and the binary points. Different values of these parameters were used to, as shown in Figure 5.



System Generator: neural_10_8_2_subsystems

Compilation : HDL Netlist

Part : Kintex7 xc7k325t-3fbg676

Synthesis tool : XST

Hardware description language : VHDL

Target directory : C:/Users/Furat/Desktop/xilinx

Project type : Project Navigator

Synthesis strategy : XST Defaults*

Implementation strategy : ISE Defaults*

[✓] Create interface document

[✓] Create testbench

[ ] Import as configurable subsystem

Figure 3. System generator compiler and VHDL code generator

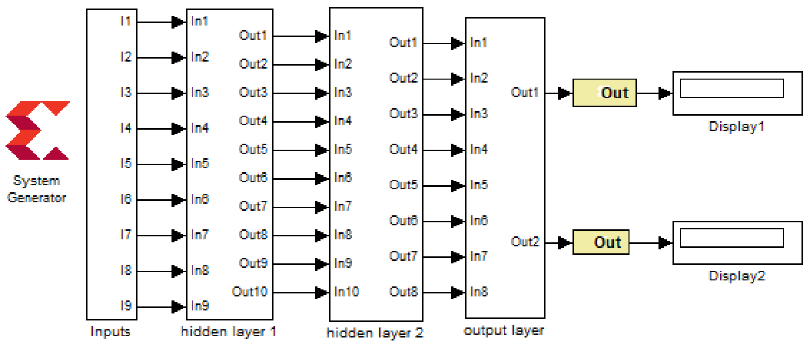
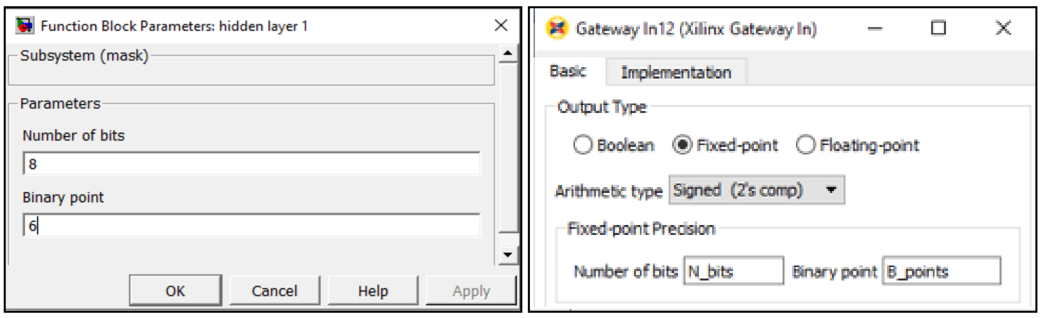


Figure 4. DNN subsystems



Function Block Parameters: hidden layer 1

Subsystem (mask)

Parameters

Number of bits: 8

Binary point: 6

Gateway In12 (Xilinx Gateway In)

Basic | Implementation

Output Type: ○ Boolean [✓ ● Fixed-point] ○ Floating-point

Arithmetic type: Signed (2's comp)

Fixed-point Precision

Number of bits: N_bits Binary point: B_points

Figure 5. Mask for Subsystem parameters selection

- Create a single neuron using ‘Multi’ and ‘addsub’ blocks. Each input is multiplied by weight using ‘Multi’ block then summed using ‘addsub’ block. The output of the final ‘addsub’ block is passed to the ‘FuncActiv’ subsystem, which represents the loglog activation function described in (4). ‘FunActiv’ subsystem final output has a value of either 0 or 1. Figure 6 shows the modeling of the loglog activation function.
- Finally, join all these blocks together in a subsystem and call it ‘Nero’ as shown in Figure 7. In our design, the number of ‘Multi’ blocks will be less than the total number of inputs by 1. The same is true for the number of ‘addsub’ blocks. Use ‘gatewayIn’ for each input to the ‘Nero’ block.

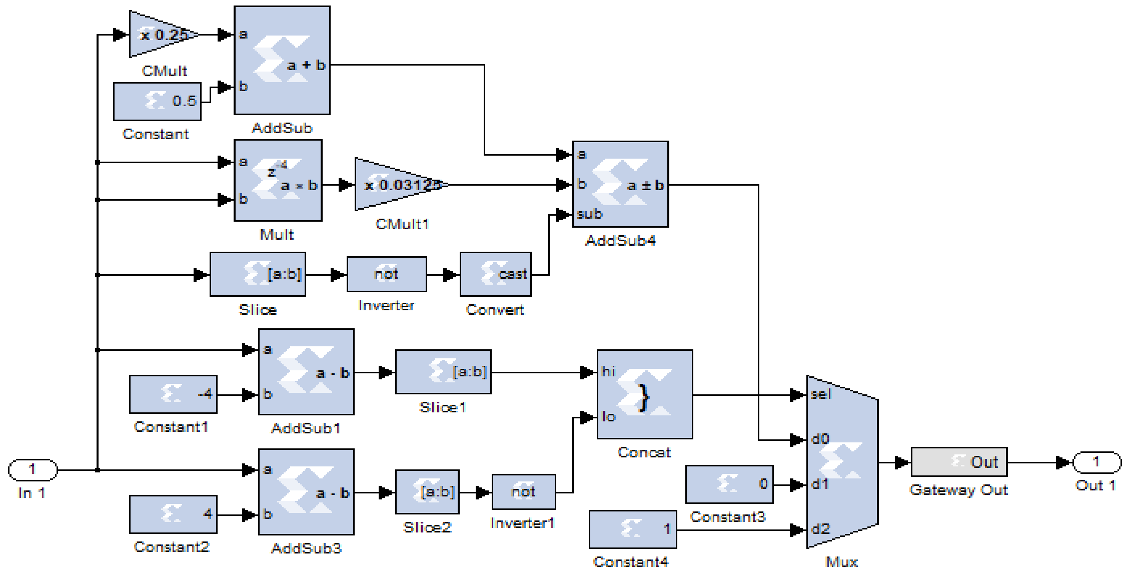


Figure 6. Modelling logsig activation function subsystem

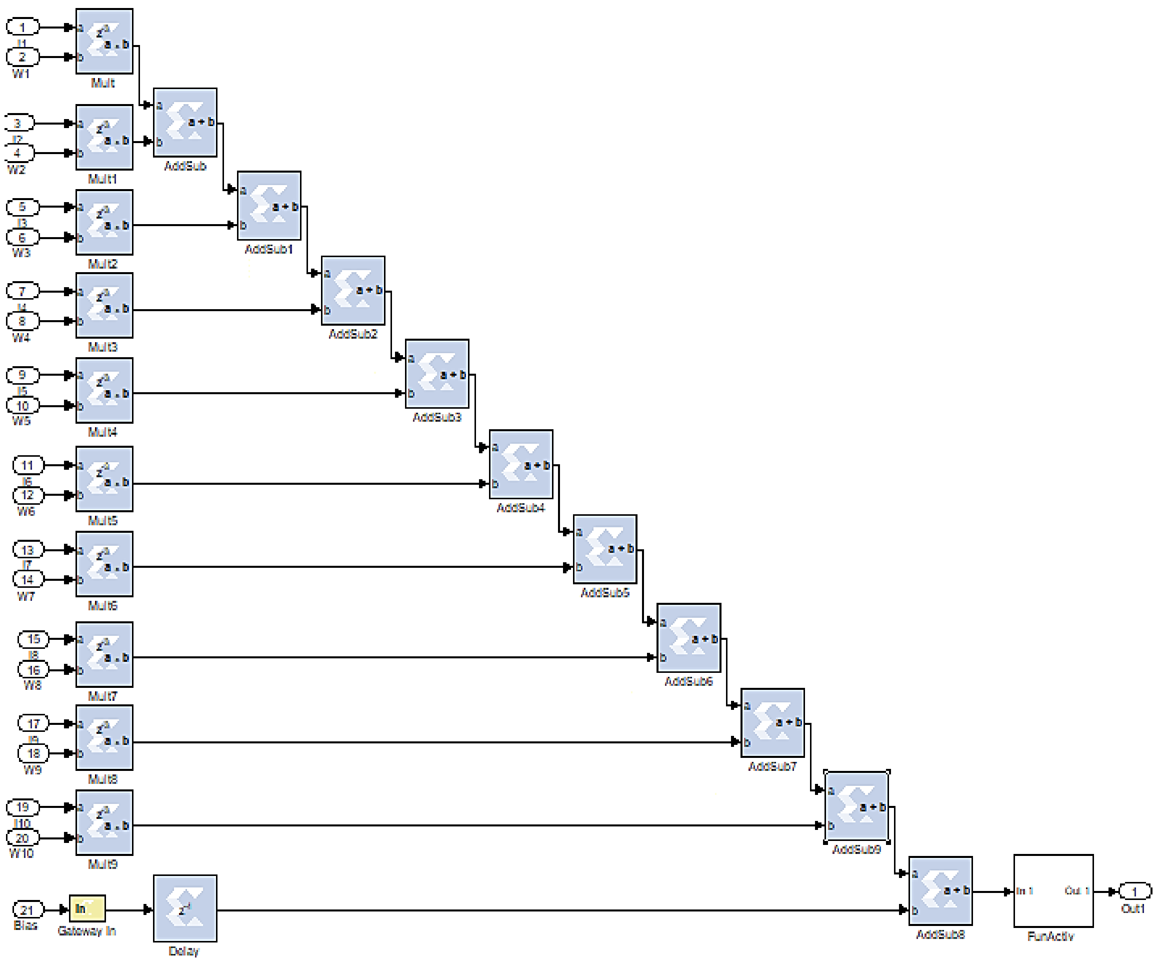


Figure 7. Neuron subsystem

- Avoid using ‘gatewayOut’ blocks between each ‘Nero’ block or between subsystems to reduce the overall used hardware components.

- Duplicate 'Nero' subsystem to the number of neurons required in each hidden layer, complete the connections, and group all in subsystem call it 'hidden layer1', 'hidden layer 2' and 'output layer 1'in Figure 4. The first hidden layers include ten fully connected neurons (Nero 1, Nero 2, …, Nero 10). Each Nero subsystem will produce one output, and the 'hidden Layer 1' subsystem will produce ten outputs. As in Figure 8. These ten outputs represent the inputs to the 'hidden layer 2' subsystem. The second subsystem includes eight neurons (Nero1, Nero2, ..., Nero8) subsystems that represent the eight neurons as in Figure 9. 'Hidden layer 2' subsystem will produce eight outputs which will be the inputs to the output layer subsystem. The output layer has two neurons (Nero1, Nero2) subsystems. It receives eight inputs from the previously hidden layer and will produce two outputs, as in Figure 10.
- Use the testing data to check the accuracy of the implementation. Two 'display' blocks were used to show classification results. The First for 'benign' and the second for 'malignant'. If display1 value '1' and display2 value '0' then classifier indicates as 'benign' case. And if display1 value '0' and display2 shows '1' then the case is classified as 'malignant'.

iv) Step four

- Press the generate button to create the VHDL code. This code assembles the implemented model, as shown in Figure 3. Open saved VHDL code using ISE design suite 14.7. Synthesize the code and convert it to a .bit file to be uploaded to the Kintex-7 xc7k325t-3fbg676 board.

3. RESULTS AND DISCUSSION

After training the DNN using MATLAB toolbox for 72 epochs, a single epoch represents the entire pass of the training algorithm over all the training sets. Each epoch consists of 10.9219 iterations, where a single iteration is one step taken in the gradient descent algorithm towards minimizing the loss function using a mini-batch. The best performance in this training was 7e-4 with a total training time of 2.262 sec. The training is completed with a total of 786.375 iterations. Figure 11 shows the DNN performance of training. The DNN accuracy of training, testing, and validation were concluded using the confusion matrix: TP, TN, FP, and FN rates as shown in Table 1.

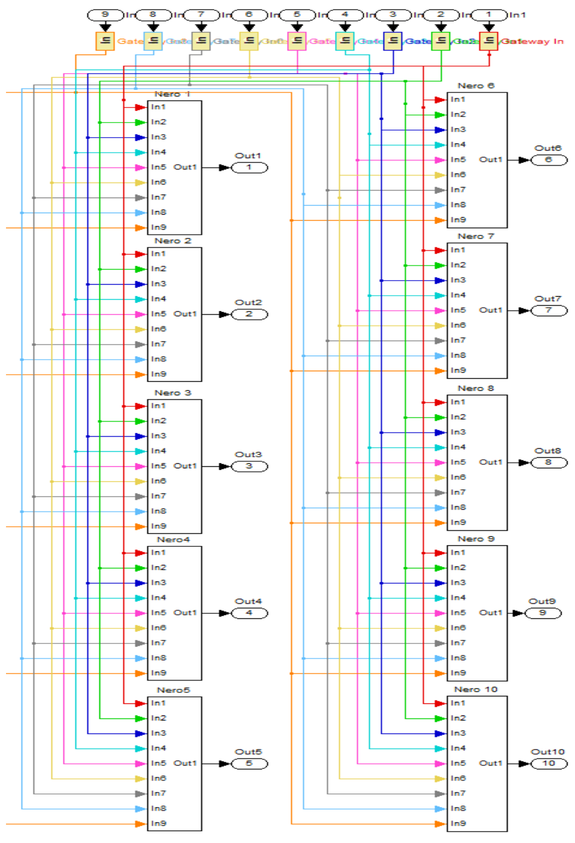


Figure 8. First hidden layer with ten neurons

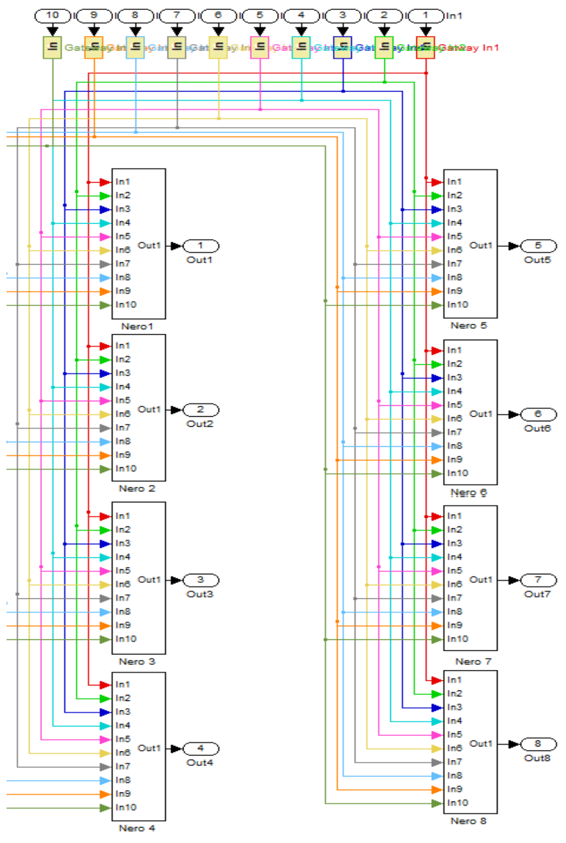


Figure 9. Second hidden layer subsystem

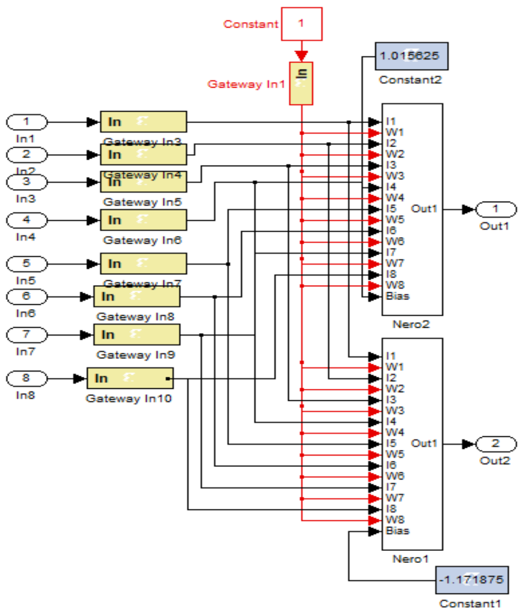


Figure 10. Output layer subsystem

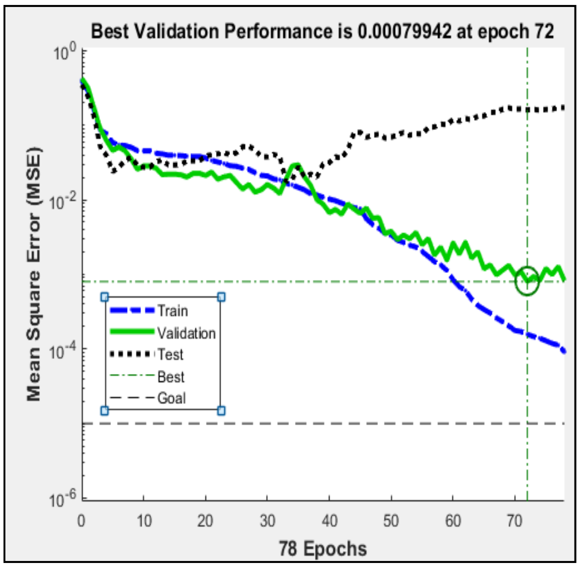


Figure 11. Performance of training and testing

Table 1. Confusion matrix

Output class		Target Class	
		Positive	Negative
	Positive	65.5%(TP)	0.4% (FN)
	Negative	0% (FP)	34.0% (TN)

The Accuracy according to the Table 1 and (5) and (6) is as:

$$\text{Accuracy} = \frac{34+65.5}{34.4+65.5} = 99.6\% \tag{5}$$

$$\text{Precision} = \frac{65.5}{65.5+0} = 100\% \tag{6}$$

Another used measurement to test the performance of DNN is ROC. The closer the ROC curve to the northwest side of the window, the higher the accuracy of DNN gives higher accuracy since TP is larger than FP. Figure 12 views the NN ROC curve for training, testing, and validation. After generating the VHDL code from the system generator, Xilinx ISE 14.7 synthesizer was carried out to find space resources occupied by the VHDL. The result is displayed in the synthesizer report as given in Figure 13. The system was uploaded to the Kintex-7 xc7k325t-3fbg676 board.

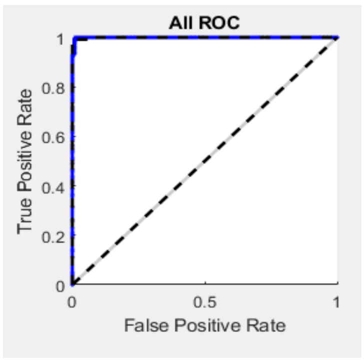
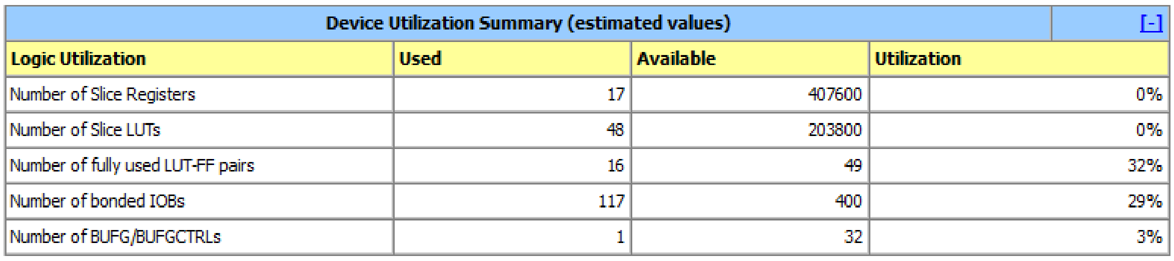


Figure 12. ROC



Device Utilization Summary (estimated values)			[-]
Logic Utilization	Used	Available	Utilization
Number of Slice Registers	17	407600	0%
Number of Slice LUTs	48	203800	0%
Number of fully used LUT-FF pairs	16	49	32%
Number of bonded IOBs	117	400	29%
Number of BUFG/BUFGCTRLs	1	32	3%

Figure 13. Summary of used space on the FPGA

To evidence the total equivalent gates used as a function of the number of bits allocated for hardware implementation of the DNN, an ISE report was generated for the hardware implementation. A resource utilization report was generated for the different number of bits and binary points for the inputs and weights. A comparison among them is shown in Table 2, which shows that 8 bits and 6 binary points give the minimum used space.

Table 2. Device utilization of FPGA logic circuits for hardware implementation

Logic Utilization	**Used**		
	(32,16)	**(16,12)**	**(8,6)**
Number of Slice Registers	65	33	17
Number of Slice LUTs	96	64	48
Number of fully used LUT-FF pairs	64	32	16
Number of bonded IOBs	385	206	112
Number of BUFG/BUFGCTRLs	1	1	1
Sum of required resources	611	336	194

The authers concentrated on using the DNN design that provides the highest accuracy and reduced hardware components. Firstly, using NN toolbox to test different designs then test the performance of each of them. The design that provides the best accuracy was chosen to be implemented in the system generator to be converted to VHDL code and tested over an FPGA device. The weights of DNN were imported from the NN toolbox to be used in the XSG. The results provided show the design can accurately classify benign from malignant cases with mean square error (MSE) equals 7e-4 after 72 training epochs and 786.375 iterations. A comparison with literary works in the same field is in Table 3.

Table 3. Comparison with ANN for breast cancer detection

Ref No.	**Size of network (I-H-O)**	**Accuracy (%)**
Singhal and Pareek [22]	9-6-2	97.8495
Janghel *et.al.* [21]	9-15-1	95.82
Shukla *et.al.* [23]	40(35-5)	92
Dabeer *et.al.* [27]	CNN	93.45
Our work	9-10-8-2	99.6

The results of FPGA implementation showed the lowest resources requirement with 194 logical resource utilization in the case of 8 bits and 6 floating points. Another comparison with researchers implementing NNs with 8 bits and 6 binary points is shown in Table 4. The resource usage and MSE of the implementation are compared with literature to show that this work has less resource usage than literature implementations with less or comparable MSE.

Table 4. Comparison with ANN resource usage and MSE

Ref No	**Resource usage (8,6)**	**MSE**
Aguiar *et.al.* [3]	500	0.2857
Khodja *et.al.* [4]	8820	1.666 e-7
Tisan *et.al.* [26]	877	1.11
Our work	194	7e-4

4. CONCLUSION

DNN was designed to find the difference between benign and malignant tumors and successfully diagnose breast cancer using two hidden layers. The DNN design required fewer adders and multipliers than used in previous papers. The total reduction was 20 adders and 20 multipliers. The Proposed DNN achieved an accuracy rate of 99.6 percent, with a 1.75 percent increase over other similar projects with hardware implementation of the DNN using the XSG. The error rate implementation using (8,6) scenario was 7e-4 with a difference of 28 percent less than similar projects and using 306 fewer hardware components, which represents 30 percent of the FPGA device space. In conclusion, the proposed DNN design successfully reduced the hardware space utilization on FPGA devices while achieving a higher accuracy rate.

REFERENCES

[1] Z. N. Isfahani, I. Jannat-Dastjerdi, F. Eskandari, S. J. Ghoushchi, and Y. Pourasad, "Presentation of novel hybrid algorithm for detection and classification of breast cancer using growth region method and probabilistic neural network," *Computational Intelligence and Neuroscience*, vol. 2021, p. 5863496, 2021, doi: 10.1155/2021/5863496.

[2] R. Sharma, "Implementation of neural network model for cancer detection based on back propagation with the help of Python based hardware description language," *International Journal of Converging Technologies and Management (IJCTM)*, vol. 2, no. 1, 2016.

[3] L. Aguiar, L. Reis, and F. M. Dias, "Neuron implementation using system generator," *Neuron*, vol. 1, no. January 2010, p. 1, 2010.

[4] D. E. Khodja, A. Kheldoun, and L. Refoufi, "Sigmoid function approximation for ANN implementation in FPGA devices," in *International conference on Circuits, Systems, Electronics, Control and Signal Processing - Proceedings*, 2010, vol. January, pp. 112–116.

[5] S. M. Hussain, K. M. Yusof, R. Asuncion, and S. A. Hussain, "Artificial intelligence based handover decision and network selection in heterogeneous internet of vehicles," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 22, no. 2, p. 1124, May 2021, doi: 10.11591/ijeecs.v22.i2.pp1124-1134.

[6] S. I. Prottasha and S. M. S. Reza, "A classification model based on depthwise separable convolutional neural network to identify rice plant diseases," *International Journal of Electrical and Computer Engineering*, vol. 12, no. 4, pp. 3642–3654, 2022, doi: 10.11591/ijece.v12i4.pp3642-3654.

[7] Z. Li *et al.*, "Detection of pancreatic cancer by convolutional-neural-network-assisted spontaneous Raman spectroscopy with critical feature visualization," *Neural Networks*, vol. 144, pp. 455–464, Sep. 2021, doi: 10.1016/j.neunet.2021.09.006.

[8] A. Ghaffari and Y. Savaria, "CNN2GATE: an implementation of convolutional neural networks inference on FPGAs with automated design space exploration," *Electronics (Switzerland)*, vol. 9, no. 12, pp. 1–23, 2020, doi: 10.3390/electronics9122200.

[9] M. Sepandi, M. Taghdir, A. Rezaianzadeh, and S. Rahimikazerooni, "Assessing breast cancer risk with an artificial neural network," *Asian Pacific Journal of Cancer Prevention*, vol. 19, no. 4, pp. 1017–1019, 2018, doi: 10.22034/APJCP.2018.19.4.1017.

[10] N. T. Le, T. T. Phung, A. H. Quyen, B. L. Phung Nguyen, and A. N. Nguyen, "A hybrid approach of artificial neural network-particle swarm optimization algorithm for optimal load shedding strategy," *International Journal of Electrical and Computer Engineering*, vol. 12, no. 4, pp. 4253–4263, 2022, doi: 10.11591/ijece.v12i4.pp4253-4263.

[11] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, Jun. 2006, doi: 10.1016/j.patrec.2005.10.010.

[12] T. Ayer, O. Alagoz, J. Chhatwal, J. W. Shavlik, C. E. Kahn, and E. S. Burnside, "Breast cancer risk estimation with artificial neural networks revisited: Discrimination and calibration," *Cancer*, vol. 116, no. 14, pp. 3310–3321, 2010, doi: 10.1002/cncr.25081.

[13] M. Z. Al-Faiz, A. A. Ibrahim, and S. M. Hadi, "The effect of Z-Score standardization (normalization) on binary input due the speed of learning in back-propagation neural network," *Iraqi Journal of Information & Communications Technology*, vol. 1, no. 3, pp. 42–48, 2019, doi: 10.31987/ijict.1.3.41.

[14] L. Niharmine, B. Outtaj, and A. Azouaoui, "Tifinagh handwritten character recognition using optimized convolutional neural network," *International Journal of Electrical and Computer Engineering*, vol. 12, no. 4, pp. 4164–4171, 2022, doi: 10.11591/ijece.v12i4.pp4164-4171.

[15] M. A. Arshad, S. Shahriar, and A. Sagahyroon, "On the use of FPGAs to implement CNNs: a brief review," in *Proceedings - 2020 International Conference on Computing, Electronics and Communications Engineering, iCCECE 2020*, 2020, pp. 230–236, doi: 10.1109/iCCECE49321.2020.9231243.

[16] D. C. Anghel, A. Ene, and N. Belu, "A matlab neural network application for the study of working conditions," *Advanced Materials Research*, vol. 837, no. November, pp. 310–315, 2014, doi: 10.4028/www.scientific.net/AMR.837.310.

[17] A. D. M. Africa, D. A. P. Abaluna, A. J. A. Abello, and J. M. B. Lalusin, "Implementation of neural network Ccontrol in a nonlinear plant using MATLAB," in *2020 IEEE 12th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management, HNICEM 2020*, 2020, pp. 2–7, doi: 10.1109/HNICEM51456.2020.9400007.

[18] N. N. Hamadneh, W. A. Khan, W. Ashraf, S. H. Atawneh, I. Khan, and B. N. Hamadneh, "Artificial neural networks for prediction of covid-19 in Saudi Arabia," *Computers, Materials and Continua*, vol. 66, no. 3, pp. 2787–2796, 2021, doi: 10.32604/cmc.2021.013228.

[19] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, "Continual lifelong learning with neural networks: a review," *Neural Networks*, vol. 113, pp. 54–71, 2019, doi: 10.1016/j.neunet.2019.01.012.

[20] S. Elfwing, E. Uchibe, and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," *Neural Networks*, vol. 107, pp. 3–11, 2018, doi: 10.1016/j.neunet.2017.12.012.

[21] R. R. Janghel, A. Shukla, R. Tiwari, and R. Kala, "Breast cancer diagnosis using artificial neural network models," in *Proceedings - 3rd International Conference on Information Sciences and Interaction Sciences, ICIS 2010*, 2010, no. CI, pp. 89–94, doi: 10.1109/ICICIS.2010.5534716.

[22] P. Singhal and S. Pareek, "Artificial neural network for prediction of breast cancer," in *Proceedings of the International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud), I-SMAC 2018*, 2019, pp. 464–468, doi: 10.1109/I-SMAC.2018.8653700.

[23] A. Shukla, R. Tiwari, P. Kaur, and R. R. Janghel, "Diagnosis of thyroid disorders using artificial Neural Networks," in *2009 IEEE International Advance Computing Conference*, Mar. 2009, vol. 56, pp. 1016–1020, doi: 10.1109/IADCC.2009.4809154.

[24] S. Bharati, P. Podder, and M. R. H. Mondal, "Artificial neural network based breast cancer screening: A comprehensive review," *International Journal of Computer Information Systems and Industrial Management Applications*, vol. 12, pp. 125–137, 2020.

[25] M. Desai and M. Shah, “An anatomization on breast cancer detection and diagnosis employing multi-layer perceptron neural network (MLP) and convolutional neural network (CNN),” *Clinical eHealth*, vol. 4, pp. 1–11, 2021, doi: 10.1016/j.ceh.2020.11.002.
[26] A. Tisan, S. Oniga, D. MIC, and A. Buchman, “Digital Implementation of The Sigmoid Function for FPGA Circuits,” *Acta Technica Napocensis*, vol. 50, no. 2, pp. 15–20, 2009, [Online]. Available: http://users.utcluj.ro/~atn/papers/ATN_2_2009_4.pdf.
[27] S. Dabeer, M. M. Khan, and S. Islam, “Cancer diagnosis in histopathological image: CNN based approach,” *Informatics in Medicine Unlocked*, vol. 16, no. August, p. 100231, 2019, doi: 10.1016/j.imu.2019.100231.

BIOGRAPHIES OF AUTHORS

Yasmine M. Tabra is an Information and Communication Engineerer and an expert who works in multiple domains, including DSP, Multimedia, Beamforming, DOA, ML, and 5G applications. Her educational background is Ph.D in ICE. She is working as a lecturer in Al Nahrain University/College of Information Engineering for over 13 years. Having over than 10 reserch papers in different fields. She can be contacted at email: yasminetabra@gmail.com.

Furat N. Tawfeeq is an information Engineerer and an expert who works in multiple domains, including ML, Computer programming, image processing, and AI. His educational background is M.Sc. in Information Engineering. He is working as a responsible for website unit\ website division\ University of Baghdad. Having over than 20 reserch papers in different fields. He can be contacted at email: furat@bccru.uobaghdad.edu.iq.