International Journal Of Artificial Intelegence Research
Vol. 6, No. 1, June 2022
ISSN: 2579-7298

Analysis of Learning Algorithms for Multilayer Neural Networks

Muhammad Khoiruddin Harahap[a],[1], Eko Pramono[b],[2], Hilda Yulia Novita[c],[3], Maharina[c],[4], Dimas Sasongko[d],[5] , Candra Zonyfar[c],[6,*]

[a]*Informatics Engineering of Ganesha Polytechnic, North Sumatera, Indonesia*
[b] *Universitas Bina Sarana Informatika, Kramat Raya No. 98 Street, Jakarta 10450, Indonesia*
[c] *Universitas Buana Perjuangan Karawang, H.S. Ronggowaluyo Street Telukjambe Timur, Karawang 41361, Indonesia*
[d] *Universitas Muhammadiyah Magelang, Mayjend. Bambang Soegeng Street, Magelang 56172, Indonesia*

[1] *choir.harahap@polgan.ac.id;* [2] *eko.eop@bsi.ac.id;* [3] *hilda.yulia@ubpkarawang.ac.id;* [4] *maharina@ubpkarawang.ac.id;*
[5] *dimassasongko@ummgl.ac.id;* [6*] *candra@ubpkarawang.ac.id*
* *corresponding author*

ARTICLE INFO

Article history:
Received 18 Nov 2021
Revised 29 Dec 2021
Accepted 25 Feb 2022

Keywords:
Artificial Neural Network
FPGA
Radial Basis Function
Multilayer NN
Learning Algorithm

ABSTRACT

The management of such systems requires the development of new management methods, since the modification and improvement of traditional management techniques does not always ensure the fulfillment of stringent requirements for management quality indicators. Classical control methods are mainly based on the theory of linear systems, while most real objects are non-linear. The problem of the synthesis of control systems under conditions of uncertainty is currently one of the central problems in the modern theory of automatic control. The complexity of the control object itself, structural, parametric and information uncertainties in the description of the control object, and the complexity of control problems, the multi criteria of optimization problems, the lack of possible analytical solutions, the need to take into account all the properties of disturbances, etc. The solution to this problem requires a search for alternative approaches to the design of control systems, one of which involves the introduction of neural network systems. Neural network control systems are a high-tech direction of control theory and belong to the class of nonlinear dynamic systems. High performance due to parallelization of input information in combination with the ability to train neural networks makes this technology very attractive for creating control devices in automatic systems. Neural networks can be used to build regulating and switching devices, reference, adaptive, nominal and inverse-dynamic models of objects, on the basis of which objects are studied, analysis of the influence of disturbances acting on an object, determination of the optimal control law, search or calculating the optimal program for changing the impact when changing the values of the parameters of the object and the characteristics of the input data. In addition, neural networks can be used to identify objects, predict the state of objects, recognize, cluster, classify, analyze a large amount of data arriving at high speed from a large number of devices and sensors, and the like. The ability to learn according to a given principle of functioning allows creating automated control systems that are optimal in terms of speed, energy consumption, etc.

I. Introduction

Adaptive systems can be divided into two major classes: self-organizing and self-adjusting. In self-organizing systems in the process of functioning is the formation of the control algorithm, which allows you to optimize the system in terms of the purpose of control. Such tasks arise, for example, in conditions of changing the structure and parameters of the control object depending on the mode of operation, when a priori information is not enough to determine the current mode of operation of the object. With a wide class of possible structures of the object, it is difficult to hope for the choice of a single structure of the control algorithm, capable of providing a closed system to achieve the goal of control in all modes of operation. Therefore, we are talking about synthesis with a free structure of the regulator. The obvious complexity of the problem does not allow to hope for simple algorithms for its solution, and hence for the widespread implementation of such systems in practice. The task is significantly simplified if the structure of the control object is known and unchanged, and the behavior depends on a number of unknown parameters. This problem is solved in the class of self-adjusting systems (SNA), in which the structure of the controller is given and you need to determine only the algorithm for adjusting the coefficients of the controller.

II. Research Method

SNAs are divided into two subclasses: search and non-search. In SNA search engines, the minimum (or maximum) measure of quality (installation performance, fuel consumption, etc.) is determined using specially organized search signals. The simplest search engines are most of the extreme systems, in which the lack of a priori information is filled at the expense of current information obtained in the form of the object's reaction to search influences. In searchless SNAs, explicitly or implicitly, there is a model with the desired dynamic characteristics. The task of the adaptation algorithm is to adjust the coefficients of the controller so as to reduce the inconsistency between the control object and the model to zero. Such control is called direct adaptive control, and systems are called adaptive systems with a reference model. In the case of indirect adaptive control, the object is first identified and then the corresponding controller coefficients are determined. Such regulators are called self-tuning regulators.

With direct adaptive control the contours of adaptation work in a closed cycle. This allows you to fend off changes in the parameters of the object and the controller during operation. However, each self-tuning circuit increases the order of the system by at least one, and at the same time significantly affects the overall dynamics of a closed system.

In the case of indirect adaptive control, the self-tuning circuits operate in an open cycle and do not affect the dynamics of the system. However, all identification errors, deviations of the parameters of the object and the controller significantly affect the accuracy of control. In searchless self-tuning systems, the reference model can be implemented as a real dynamic link (explicit model) or be present in the form of some reference equation that connects the regulated variables and their derivatives (implicit model). In the implicit model, the coefficients of the reference equation are the parameters of the adaptation algorithm.

From the very beginning of the third stage, associated with the adaptive formulation of the main task of management, there was a huge number of scientific articles and reports on developments in the field of ANN in the context of adaptive management theory, namely the implementation of such components of adaptive systems as regulators and object models. The use of ANN for the implementation of SU has many advantages due to the following properties: ANN are ideal approximators of any nonlinear function of many variables, which allows you to model the OK and form control functions of any complexity; internal adaptability due to the possibility of self-learning; high speed and parallel information processing.

Neural networks can be used in facilities such as robotics, unmanned aerial vehicle control, vehicle control, pattern recognition, analysis and decision making in Internet of Things systems, spacecraft control, military equipment and many others. areas of application in modern technologies. In these systems, neural networks can be used to identify objects, predict the state of objects, recognition, clustering, classification, analysis of large amounts of data coming at high speed from a large number of devices and sensors, etc. When describing control systems with neural network elements and their implementation on a modern element base, a discrete representation of OK is more convenient, so in the future we will consider the dynamics of the system in the form of.

The properties of ANN allow to model complex nonlinear dynamic control objects in the form of direct and models for measuring the "input-output" of this object. Both models are used to calculate the vectors of the state of the object and the formation of its control function. They can be the basis for the structural synthesis of functionally more complex control systems. One of the main features of neural networks is the parallel processing of signals. Multilayer neural networks are a homogeneous computing environment. In the terminology of neuroinformatics, these are universal parallel computing structures designed to solve a variety of classes of problems. Consider the basic principles of parallel signal processing on the FPGA and the concept of implementing neural networks on the FPGA. Conveyor processing is a method in which several instructions or data packets can be processed by the processor in parallel. To do this, the whole procedure of processing one command is divided into several stages, at the end of each of which the result is stored in temporary registers, through which the individual stages are switched between them.

Deployment of internal cycles is a well-known method, which in some cases can lead to a significant increase in productivity. The essence of the method is to present cyclic processing algorithms with a finite number of iterations in the form of a long combinational chain that is processed in one cycle. In multiprocessor processing, more than one processor in the system processes incoming instructions, thereby allowing completely parallel processing.

In the software and hardware implementation of artificial neural networks on the FPGA, each layer of the network works in parallel with the other, here the principle of the pipeline is used. Neurons in each layer also work in parallel on the principle of multiprocessor data processing. That is, each artificial neuron in the network is a separate processor and the processing of information in each neuron takes place simultaneously. Each neuron is represented by a separate block, as shown in Fig. 1, which in turn consists of several parallel processes, and the neural network is a multiprocessor system. The programming language allows you to explicitly specify the signals that start the process. The input signal of this neuron is used to start a neuron.

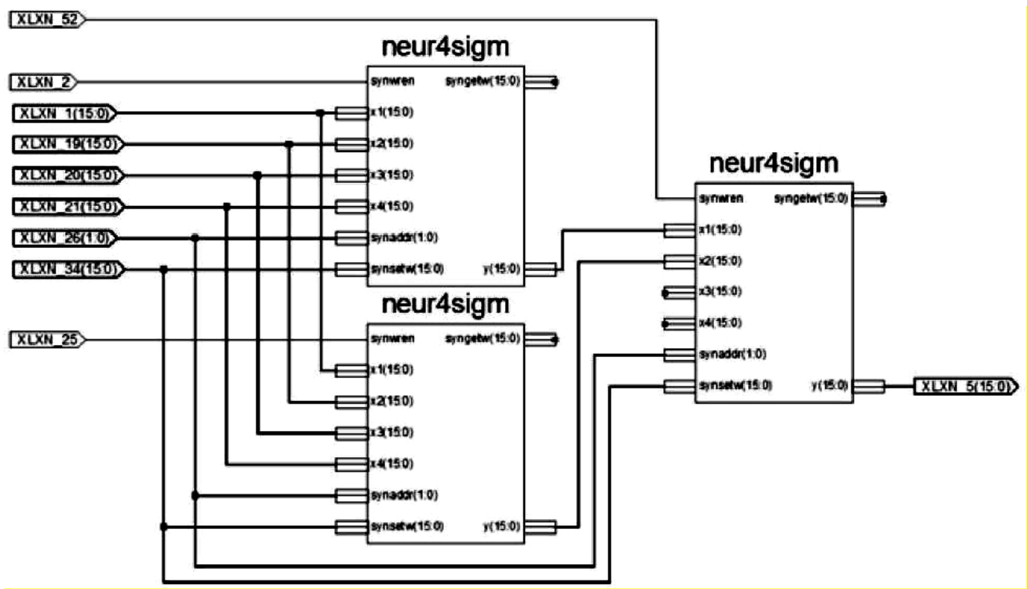

Fig. 1.Neural network built on an FPGA

Series of direct propagation neural networks, the FPGA resource occupied by them in the number of valves of the logical matrix LUTs, the speed and error of the presentation in table 1. were modeled.

Table 1. The results of neural network modeling in FPGA

Neural Network	Number of synapses	Resource FPGA (LUT)	The initial value of the ANN		Error	Speed-action, ns
			MatLab	*FPGA*		

1-1-1	3	484	0.56389366	0.546875	0.017019	120.243
1-2-1	5	585	0.62573413	0.609375	0.016359	126.216
1-5-1	11	523	0.78328235	0.765625	0.017657	128.754
2-2-1	8	488	0.63863534	0.625	0.013635	150.481
2-3-1	11	790	0.70144096	0.6875	0.013941	130.580
2-4-2	18	899	0.75747616	0.75	0.007476	133.601
4-4-4	36	1104	0.79223947	0.765625	0.026614	133.534
3-5-1	23	725	0.82549114	0.796875	0.028616	134.047
2-4-1	14	933	0.75747616	0.75	0.007476	133.537
1-3-1	7	723	0.68372392	0.671875	0.011849	127.707
2-1-1	5	602	0.57070375	0.5625	0.008204	126.216
1-4-1	9	829	0.73651211	0.734375	0.002137	127.632
1-6-1	13	1065	0.82373748	0.796875	0.026862	132.262
3-1-1	7	723	0.577080667	0.5625	0.014581	128.705
3-2-1	11	819	0.65058263	0.625	0.025583	134.138
2-8-2	34	1369	0.907020339	0.890625	0.016395	163.237

In the table, the structure of the studied neural networks is represented by three numbers, where the first is the number of neurons in the input layer, the second is the number of neurons in the hidden layer and the third is the number of neurons in the output layer. The obtained values in LUTs may change slightly with the change of the bit synaptic weights of neurons. In comparison with the analogue where the ANN 3-2-1, which occupies 1125 LUT, is presented, the network implemented in this work occupies 819 LUT.

To obtain the error of the hardware-implemented ANN in the FPGA, a test model was collected in the software package MatLab, Fig. 2, presents the values of the output of the ANN implemented on the FPGA, and in MatLab, with the same inputs of the ANN and the parameters, their difference in the column "Error"

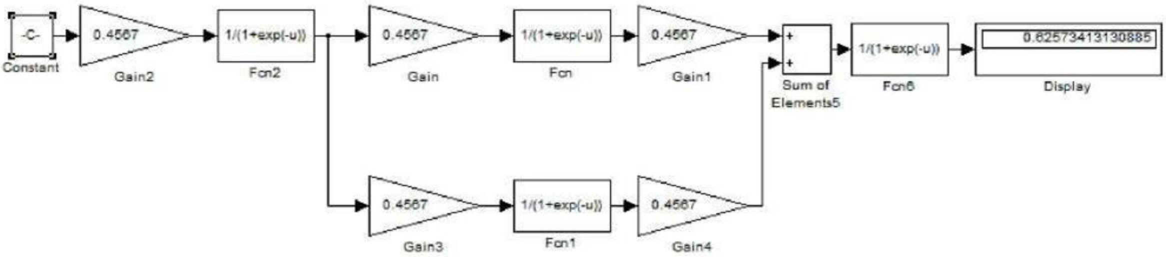

Fig. 2.Model of NN 1-2-1 in MatLab

The hardware implementation of Hopfield neural networks on FPGA differs from the hardware implementation of multilayer neural networks of direct propagation by introducing additional feedback and time delay elements on these connections, which in turn takes more FPGA resources. Modeling of Hopfield neural networks with one layer and three neurons in it and two layers of three neurons in each showed that for their implementation the required FPGA resource is respectively 2521 and 4945 LUTs when implemented on the Xilinx Spartan 3 chip.

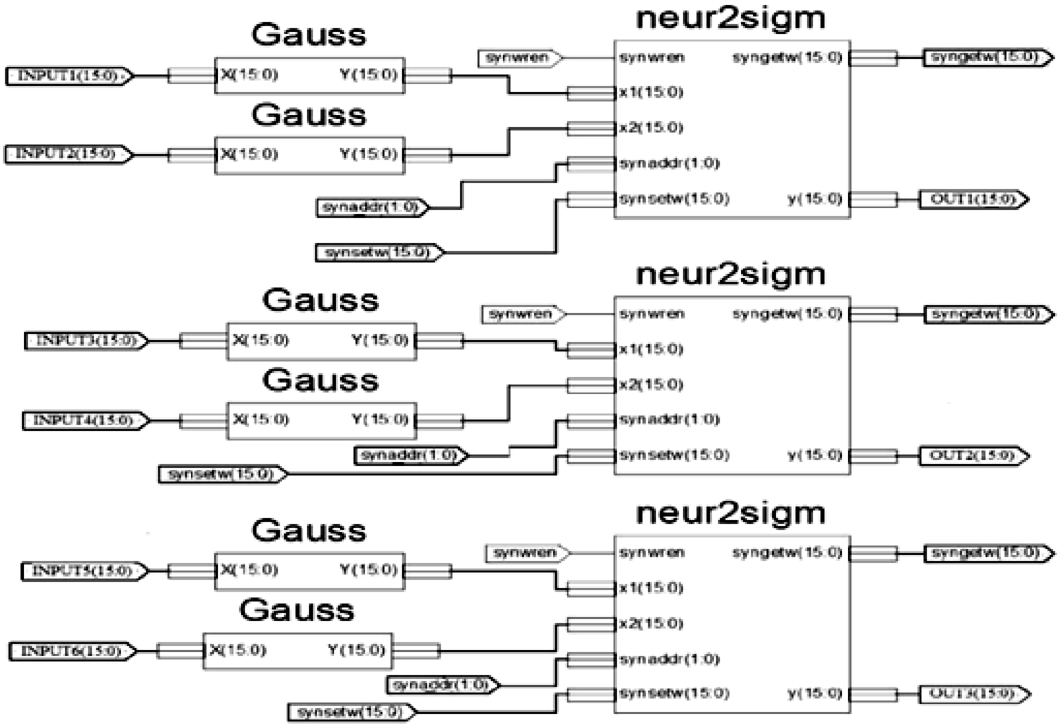

Fig. 3.RBF NN 2-3 built on FPGA

RBF networks were hardware implemented in the work. In Fig. 3 presents an RBF neural network, topologies 2-3, built on FPGA.

III. Result

The results of modeling the RBF-network and, accordingly, the FPGA-LUTs resource occupied by them, the performance are presented in Table 2.6. The topology of neural networks is represented by two numbers, where the first is the number of Gaussian functions in the input layer, the second is the number of neurons with the sigmoidal activation function in the output layer. When configuring radial-based neural networks, a chip of the Spartan 3 family was used. The values obtained in LUTs may change slightly when the constants that set the synaptic weights of neurons change.

Table 2. The Results of NN modeling in FPGA

Neural network	**Gaussian functions**	**Resource FPGA (LUT)**	**Speed, ns**
1-1	1	452	86.671
2-1	2	771	86.736
3-1	3	1090	86.761
4-1	4	1409	86.770
1-2	2	695	90.043
2-2	4	1110	90.107
3-2	6	1525	90.133
4-2	8	1940	90.141
1-3	3	942	93.913
2-3	6	1470	93.978
3-3	9	1998	94.003

4-3	12	2526	94.012
1-4	4	1193	101.579
2-4	8	1838	101.755
3-4	12	2483	101.78
4-4	16	3128	101.789
1-5	5	1445	105.123
2-5	10	2192	105.188
1-6	6	1679	107.54
2-6	12	2511	107.604
1-8	8	2156	112.968
2-8	16	3196	113.033

In Fig. 3 show graph of the dependence of the amount of resource used on the number of neurons and the speed of the RBF neural network on the number of neurons. As can be seen from the graphs with increasing number of neurons in the network, the speed has not changed as each neuron and computational operations in it are separate parallel processes.

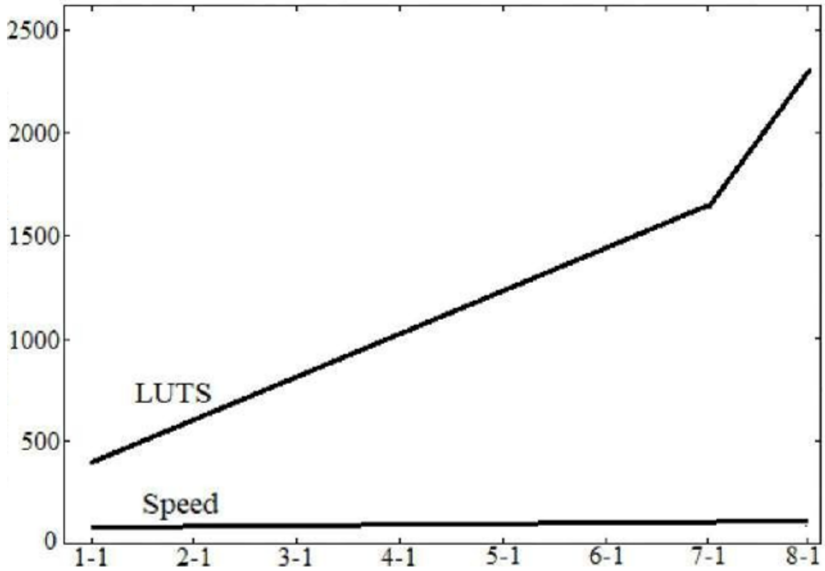

Fig. 4. Dependence amount used resource on the number of neurons and speed RBF NN on number of neurons

Table 3. Compares the results of modeling RBF NN on FPGA implemented by the nearest known analogue

Neural network	**Resource FPGA (LUT)**		**Speed, NS**	
	Analog	***Designed***	***Analog***	***Designed***
2-1	1170	257	41.26	27.524
4-1	2370	420	46.87	27.752

Analogue and the implemented RBF neural network in accordance with the developed method and algorithm in this work. The simulation is performed on the same Xilinx Virtex-6 chip.

As can be seen from Table 3, the implementation of RBF neural networks according to the developed method and algorithm has a higher speed and requires less chip resource. In the realized neural networks in this work nonlinear, sigmoidal functions of an initial layer in contrast to linear in work are used.

IV. Conclusion

Based on the proposed method, algorithms for hardware implementation of an artificial neuron with a sigmoidal activation function and a neuron of the hidden layer of the RBF network with a Gaussian activation function have been developed. A study of implemented artificial neurons and ANN. It is shown that due to the developed method and algorithms significant optimization of the used resource is provided, the speed of calculations of hardware units with ANN and their accuracy in comparison with analogues increases. This method and algorithms can be basic for further design of neural network components of dynamic object control systems based on FPGA. The main results obtained in this section include the following: A method for designing nonlinear activation functions of an artificial neuron on programmable logic integrated circuits is developed, which differs in that the coefficients of the piecewise linear approximation of the activation function are stored in memory only for positive or only for negative values of the argument, which allowed optimize the amount of used computing resources and increase the speed of computing neural networks; An algorithm for hardware implementation of an artificial neuron with a sigmoid activation function was developed, which allowed to increase the speed due to the parallelization of a large number of operations, and to make optimal use of the occupied resource in the FPGA; Developed an algorithm for hardware implementation of the neuron of the hidden layer of the RBF network with the Gaussian activation function according to the developed method of designing the activation function, which allowed to make optimal use of the occupied resource in FPGA and increase the synthesis of such components; Implemented and modeled artificial neurons and neural networks, according to the developed method and algorithm, according to the simulation results showed high accuracy and speed.

References

[1] Mirsadeghi, Maryam, et al. "STiDi-BP: Spike time displacement based error backpropagation in multilayer spiking neural networks." Neurocomputing 427 (2021): 131-140.

[2] Kiranyaz, S., Avci, O., Abdeljaber, O., Ince, T., Gabbouj, M., & Inman, D. J. (2021). 1D convolutional neural networks and applications: A survey. Mechanical systems and signal processing, 151, 107398.

[3] Liu, Xiaorui, et al. "Elastic graph neural networks." International Conference on Machine Learning. PMLR, 2021.

[4] Goh, Gabriel, et al. "Multimodal neurons in artificial neural networks." Distill 6.3 (2021): e30.

[5] Peng, Yijie, et al. "A new likelihood ratio method for training artificial neural networks." INFORMS Journal on Computing (2021).

[6] Katz, Guy, et al. "Reluplex: a calculus for reasoning about deep neural networks." Formal Methods in System Design (2021): 1-30.

[7] Cinar, Ahmet Cevahir. "Training feed-forward multi-layer perceptron artificial neural networks with a tree-seed algorithm." Arabian Journal for Science and Engineering 45.12 (2020): 10915-10938.

[8] Larriva-Novo, Xavier A., et al. "Evaluation of cybersecurity data set characteristics for their applicability to neural networks algorithms detecting cybersecurity anomalies." IEEE Access 8 (2020): 9005-9014.

[9] Adhikari, Shyam Prasad, et al. "Hybrid no-propagation learning for multilayer neural networks." Neurocomputing 321 (2018): 28-35.

[10] Widrow, Bernard, et al. "The no-prop algorithm: A new learning algorithm for multilayer neural networks." Neural Networks 37 (2013): 182-188.