DESIGN AND SIMULATION FUZZY NEURO GENERALIZED LEARNING VECTOR QUANTIZATION-PI (FNGLVQ-PI) ON FIELD PROGRAMMABLE GATE ARRAY (FPGA)

Ricky A. Daniel, M. Anwar Ma'sum, Grafika Jati, and Wisnu Jatmiko
Faculty of Computer Science Universitas Indonesia
Kampus UI Depok, Jawa Barat, 16424, Indonesia

Abstract

Classification is a machine learning technique that used widely for various applications. One of the classification algorithm is FNGLVQ that can be used to solve several types of classification problem in previous research. On the other side, Field Programmable Gate Array (FPGA) is an instrument currently used in many application especially for portable smart device. In this paper we will discuss thoroughly about design and simulation of FNGLVQ algortihm on FPGA. FNGLVQ is an artificial neural network based algorithm that can be used for several applications in previous researches. The design consists of two major phases, training and testing phase. The design was implemented in Xilinx ISE Project Navigator which is an integrated development environtment (IDE) to build the design into the actual Xilinx FPGA. The IDE also provides simulation feature for the design. In this research, we use Iris dataset taken from UCI Machine Learning database. Simulation result shows that this design reached 90.00%, 93.33%, 93.33%, 83.33%, 80.00%, 83.00%, and 86.67% accuracy respectively for epoch value 1, 2, 4, 8, 16, 32 and 64. As comparison, FNGLVQ implemented in MATLAB constantly reached 93.33% accuracy for those variation of epoch. . However, running time on the FPGA side is approximately twenty time faster than on MATLAB side

Keywords: Design, simulation, FPGA, FNGLVQ

1. Introduction

Classification is a machine learning technique that currently used to solve several types of problem in human life. There are real world problems that have irregular pattern and cannot be formulated into compact mathematical equation. Classification is used to categorize data based on their features. There are several algorithm used for classification problem, such as decision tree, support vector machine (SVM), and artificial neural network (ANN) [1].

There are several types artificial neural network used widely for many kind applications such as Single Layer Perceptron, Multilayer Perceptron, Backpropagation, and Learning Vector Quantization (LVQ) [2]. LVQ is a competitive based artificial neural network that uses distance to choose winner vector (winner class). LVQ is proposed by T. Kohonen as supervised version of clustering algorithm Self Organizing Map (SOM) [3]. LVQ was modified by researchers with many types of enhancements. Sato and Yamada applied gradient descent optimization method to LVQ resulting into new algorithm called Generalize Learning Vector Quantization (GLVQ) [4]. In previous research, Setiawan et al modified GLVQ with fuzzy membership function resulting into Fuzzy Neuro Generalize Learning Vector Quantization (FNGLVQ) [5]. They used FNGLVQ for automatic arrhythmias classification. Furthermore, FNGLVQ was modified by Ali Akbar et al with PI membership function resulting FNGLVQ-PI [6].

On the other hand, Field Programmable Gate Array (FPGA) became popular instrument used for various needs. In previous research, it was used for high frequency signal processing, bit error rate measurement, and fuzzy logic controller [7][8][9]. In this paper we propose a design and simulation of FNGLVQ-PI classifier on FPGA. It is an extension of previous research conducted by Suryana et al [10]. The design can be used for many classification cases.

The rest of the paper is organized as follows. Section 2 discusses dataset and features used in this research, Section 3 discussed FNGLVQ-PI classifiers used in this research. Section 4 discusses about the design and simulation of the classifiers in FPGA while section 5 shows experiment results and discussion. In section 6 we will explain the conclusion of the researh.

2. Dataset and Features

In this research we use IRIS dataset from the UCI machine learning dataset. The dataset can be accessed in the online repository [3]. The dataset has three classes which are the Iris setosa, Iris versicolor and Iris vurginica. The dataset has 150 instances with 50 instances for each class. In this research we use sepal length, sepal width, petal length, and petal width as features for classification. Detailed information about dataset is explained in the table 1. In table 1, $\bar{x}$ is mean of the feature value and σ represents the standard deviation.

Table 1. Iris dataset

No.	Feature 1 : Sepal length				Class
	min	max	$\bar{x}$	σ	
1-50	4.3	5.8	5.00	0.35	Iris setosa
51-100	4.9	7.0	5.94	0.52	Iris versicolor
101-150	4.9	7.9	6.59	0.64	Iris virginica
1-150	4.3	7.9	5.84	0.83	
No.	**Feature 2 : Sepal width**				**Class**
	min	**max**	$\bar{x}$	**σ**	
1-50	2.3	4.4	3.42	0.38	Iris setosa
51-100	2.0	3.4	2.77	0.31	Iris versicolor
101-150	2.2	3.8	2.79	0.32	Iris virginica
1-150	2.0	4.4	3.05	0.43	
No.	**Feature 3 : Petal length**				**Class**
	min	**max**	$\bar{x}$	**σ**	
1-50	1.0	1.9	1.46	0.17	Iris setosa
51-100	3.0	5.1	4.26	0.47	Iris versicolor
101-150	4.5	6.9	5.55	0.55	Iris virginica
1-150	1.0	6.9	3.76	1.76	
No.	**Feature 4 : Petal width**				**Class**
	min	**max**	$\bar{x}$	**σ**	
1-50	0.1	0.6	0.24	0.11	Iris setosa
51-100	1.0	1.8	1.32	0.20	Iris versicolor
101-150	1.4	2.5	2.03	0.27	Iris virginica
1-150	0.1	2.5	1.20	0.76	

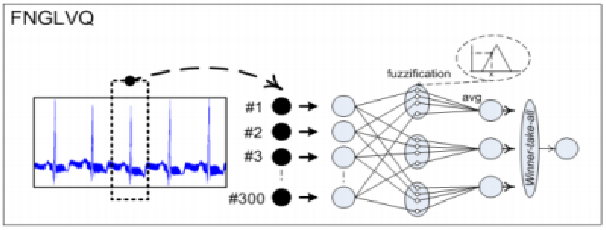

Figure 1. FNGLVQ classifier architecture [5]

3. Fuzzy Neuro Generalized Learning Vector Quantization –PI (FNGLVQ-PI)

Fuzzy Neuro Generalized Learning Vector Quantization was developed by Setiawan et al for automatic arrhythmia classification [5]. They apply fuzzy theorem to GLVQ that was proposed by Sato and Yamada [4]. FNGLVQ architecture is shown in figure 1. Then, FNGLVQ was modified by Akbar et al using PI membership function resulting FNGLVQ-PI. Mathematically, similarity value of FNGLVQ can be formulated by equations below

$$u_j = \frac{1}{k}\sum_{i=1}^{k}\mu_{ij} \qquad (1)$$

$$u_{ij} = h_{ij}(x_i) \qquad (2)$$

where u_{ij} is the similarity value of class j for feature I and u_j is similarity value of class j. Winner vector can be defined using the following equation

$$w_p = \max_{j}\ (u_{ij}) \qquad (3)$$

Miss classification error equation are adopted from GLVQ and can be defined by equation below

$$\varphi(x) = \frac{d_1 - d_2}{d_1 + d_2} \qquad (4)$$

where $\boldsymbol{w_1}$is defined as the reference vector of input vector class $\boldsymbol{C_x} = \boldsymbol{C_{w_1}}$, and $\boldsymbol{w_2}$ is the closest referrence class from different class $\boldsymbol{C_x} \neq \boldsymbol{C_{w_2}}$, $\boldsymbol{d_1}$ is distance between $\boldsymbol{w_1}$ and input vector *x*, $\boldsymbol{d_2}$ is distance between $\boldsymbol{w_2}$ and input vector *x*. In FNGLVQ, distance value is converted into d= 1 - μ, and substituted into equation 4, resulting into the equation below

$$\varphi(x) = \frac{\mu_2 - \mu_1}{2 - \mu_2 - \mu_1} \tag{5}$$

Similar to GLVQ, we define cost function for update reference vector according to the following equation

$$S = \sum_{i=1}^{N} f(\varphi(x)) \tag{6}$$

where f(**x**) is a rising monotonic function. Then, update of reference vector is based on derivation of S to weight that can be expressed as follows

$$\frac{\delta s}{\delta w_i} = \frac{\delta s}{\delta \varphi} . \frac{\delta \varphi}{\delta \mu} . \frac{\delta \mu}{\delta w_i} \tag{7}$$

In the FNGLVQ algorithm proposed by Setiawa et al, fuzzy membership function is defined by the following equation

$$\mu = h(x, w_{min}, w_{mean}, w_{max}) \tag{8}$$

$$= \begin{cases} 0 & , \quad x \le w_{min} \\ \dfrac{x - w_{min}}{w_{mean} - w_{min}}, & w_{min} \le x \le w_{mean} \\ \dfrac{w_{max} - x}{w_{max} - w_{mean}}, & w_{mean} \le x \le w_{max} \\ 0 & , \quad x \ge w_{max} \end{cases}$$

Derivation of membership function to (w_{mean}) produce update formula as follows

- If $w_{mean} < x \le w_{mean}$

$$w_1(t+1) \leftarrow w_1(t) - \alpha \, x \frac{\delta f}{\delta \varphi} \mathrm{x} \frac{2.(1-\mu_2)}{(2-\mu_1-\mu_2)^2} . \; x \left(\frac{x - w_{min}}{(w_{mean} - w_{min})^2}\right) \tag{9}$$

$$w_2(t+1) \leftarrow w_2(t) \alpha \, x \frac{\delta f}{\delta \varphi} \mathrm{x} \frac{2.(1-\mu_1)}{(2-\mu_1-\mu_2)^2} \; \mathrm{x} \left(\frac{x - w_{min}}{(w_{mean} - w_{min})^2}\right) \tag{10}$$

- If $w_{mean} < x < w_{max}$

$$w_1(t+1) \leftarrow w_1 + \alpha \, x \frac{\delta f}{\delta \varphi} \boldsymbol{x} \frac{2.(1-\mu_2)}{(2-\mu_1-\mu_2)^2} \; x \left(\frac{w_{max} - x}{(w_{max} - w_{mean})^2}\right) \tag{11}$$

$$w_2(t+1) \leftarrow w_2(t) \alpha x \frac{\delta f}{\delta \varphi} \mathrm{x} \frac{2.(1-\mu_1)}{(2-\mu_1-\mu_2)^2} \; \mathrm{x} \left(\frac{w_{max} - x}{(w_{max} - w_{mean})^2}\right) \tag{12}$$

- If $x \le w_{min}$ or $x \ge w_{max}$

$$w_i(t+1) \leftarrow w_i(t) \qquad , i = 1,2 \tag{13}$$

where w_1 is reference vector from input class $C_x = C_{w_1}$, and w_2 is the closest reference vector from different class $C_x \ne C_{w_2}$. Update rules for w_{min} and w_{max} follows the equations below

$$w_{min} \leftarrow w_{mean}(t+1) - \left(w_{mean}(t) - w_{min}(t)\right) \tag{14}$$

$$w_{max} \leftarrow w_{mean}(t+1) - \left(w_{mean}(t) - w_{min}(t)\right) \tag{15}$$

The value of α is between 0 and 1. Its value will decrease along with the value of number iteration (t) as defined in the following equation

$$\alpha(t+1) = \alpha_0 \times \left(1 - \frac{t}{t_{max}}\right) \tag{16}$$

To gain better performance, we use additional rule to adjust w_{min} and w_{max} as defined in conditions below

- If μ1 > 0 or μ2 > 0, then at least one of the two reference vectors recognize the input. If φ < 0 then increase fuzziness using equations below

$$w_{min} \leftarrow w_{mean} - (w_{mean} - w_{min}) \; x\left(1 + (\beta.\alpha)\right) \tag{17}$$

$$w_{max} \leftarrow w_{mean} + (w_{max} - w_{mean}) \; x\left(1 + (\beta.\alpha)\right) \tag{18}$$

- If input class is recognized into wrong class (φ ≥ 0), then decrease fuzziness using equations below

$$w_{min} \leftarrow w_{mean} - (w_{mean} - w_{min}) \; x\left(1 - (\beta.\alpha)\right) \tag{19}$$

$$w_{max} \leftarrow w_{mean} + (w_{max} - w_{mean}) \; x\left(1 - (\beta.\alpha)\right) \tag{20}$$

If μ1 = 0 and μ2 = 0, then it means that both reference vector cannot recognize the input vector. Therefore, fuzziness is increased using equations below. We use 0.1 for γ in our research.

$$w_{min} \leftarrow w_{mean} - (w_{mean} - w_{min}) \; x\left(1 - (\gamma.\alpha)\right) \tag{21}$$

$$w_{max} \leftarrow w_{mean} + (w_{max} - w_{mean}) \; x\left(1 + (\gamma.\alpha)\right) \tag{22}$$

In previous research Akbar et al modified FNGLVQ with PI membership function to gain better performance. Illustration of PI membership function can be seen in figure 2.

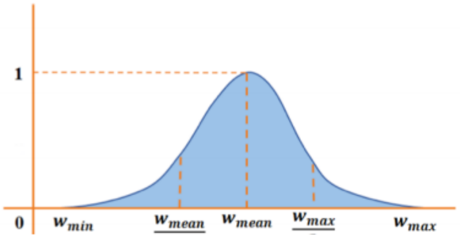

Figure. 2.Illustration of PI membership function

PI membership function can be formulated as equation below.

$$\mu = h(x, w_{min}, w_{mean}, w_{max}) \tag{23}$$
$$= \begin{cases} 2\left(\frac{x - w_{min}}{w_{mean} - w_{min}}\right)^2, w_{min} \le x \le \frac{w_{mean}}{2} \\ 1 - 2\left(\frac{w_{mean} - x}{w_{mean} - w_{min}}\right)^2, \frac{w_{mean}}{2} \le x \le w_{mean} \\ 1 - 2\left(\frac{x - w_{mean}}{w_{max} - w_{mean}}\right)^2, w_{mean} \le x \le \frac{w_{max}}{2} \\ 2\left(\frac{w_{max} - x}{w_{max} - w_{mean}}\right)^2, \frac{w_{max}}{2} \le x \le w_{max} \\ 0, \quad x \le w_{min} \; or \; w_{maz} \le x \end{cases}$$

Then equation 23 is substituted to equation (5) and equation (7) resulting new update rule for reference vector as defined in equations below.

- If $\frac{w_{mean}}{2} < x \le w_{mean}$

$$w_1(t+1) \leftarrow w_1(t) + \alpha \frac{\delta f}{\delta \varphi} \frac{2.(1-\mu_2)}{(2-\mu_1-\mu_2)^2}. \tag{24}$$
$$x\left(\frac{(x-w_{mean})(x-w_{min})}{(w_{mean}-w_{min})^3}\right)$$

$$w_2(t+1) \leftarrow w_2(t) - \alpha \frac{\delta f}{\delta \varphi} \frac{2.(1-\mu_1)}{(2-\mu_1-\mu_2)^2} \tag{25}$$
$$x\left(\frac{(x-w_{mean})(x-w_{min})}{(w_{mean}-w_{min})^3}\right)$$

- If $w_{mean} < x \le \frac{w_{max}}{2}$

$$w_1(t+1) \leftarrow w_1 + \alpha \frac{\delta f}{\delta \varphi} \frac{2.(1-\mu_2)}{(2-\mu_1-\mu_2)^2} \tag{26}$$
$$x\left(\frac{(w_{mean}-x)(x-w_{max})}{(w_{max}-w_{mean})^3}\right)$$

$$w_2(t+1) \leftarrow w_2(t) - \alpha \frac{\delta f}{\delta \varphi} \frac{2.(1-\mu_1)}{(2-\mu_1-\mu_2)^2} \tag{27}$$
$$x\left(\frac{(w_{mean}-x)(x-w_{max})}{(w_{max}-w_{mean})^3}\right)$$

- If $\frac{w_{max}}{2} < x \le w_{max}$

$$w_1(t+1) \leftarrow w_1 + \alpha \frac{\delta f}{\delta \varphi} \frac{2.(1-\mu_2)}{(2-\mu_1-\mu_2)^2} \tag{28}$$
$$x\left(\frac{(w_{max}-x)^2}{(w_{max}-w_{mean})^3}\right)$$

$$w_2(t+1) \leftarrow w_2(t) - \alpha \frac{\delta f}{\delta \varphi} \frac{2.(1-\mu_1)}{(2-\mu_1-\mu_2)^2} \tag{29}$$
$$x\left(\frac{(w_{max}-x)^2}{(w_{max}-w_{mean})^3}\right)$$

- If $x \le w_{min}$ or $x \ge w_{max}$

$$w_i(t+1) \leftarrow w_i(t) \qquad , i = 1,2 \tag{28}$$

Equations 9-13 were used for FNGLVQ with triangle membership function, whereas equations 24-28 were used for PI membership function. In FNGLVQ, the monotonic function used is sigmoid function as written in equation below.

$$f(\varphi, t) = \frac{1}{1 + e^{-\varphi t}} \tag{29}$$

$$\frac{\delta f}{\delta \varphi} = f(\varphi, t)(1 - f(\varphi, t)) \tag{40}$$

4. Design and Implementation

The proposed design includes training and testing environment. The implementation follows the structure presented in Figure 3. There are two kinds of memory in FPGA, block RAM and distributed RAM. Distributed RAM contains look up tables (LUT) in it, and is faster than block RAM. However, block RAM is better if the size of the data is relatively large. The dataset for training and testing session is stored in block RAM. On the other hand, the reference vector is stored in distributed RAM. Each of training and testing data stored in RAM contains feature set and class set, and the reference vector stored in RAM contains the minimum vector, maximum vector, and mean vector. In the update process, the reference vector is written back to distributed RAM. In this design, the data used is offline, which means that it is stored in memory, not directly stream from the source.

An FPGA can perform basic arithmetic operations such as addition, subtraction, and multiplication. Addition and subtraction operations are simple and don't need many resources of the FPGA. Meanwhile, multiplication operation needs large resources and is limited to the number of multiplier units in the FPGA itself. For example, a 32-bit multiplication operation needs 4 multiplier units while the FPGA board used in this research, Xilinx Spartan-3AN XC3S700AN, contains 18 multiplier units. Since the FNGLVQ implementation has division and sigmoid derivative operations, therefore we still need to create the divisor and sigmoid derivative computer components. The divisor core takes the idea of non-restoring division [12] while the sigmoid derivative computer core takes the idea of piecewise polynomial approximation [13].

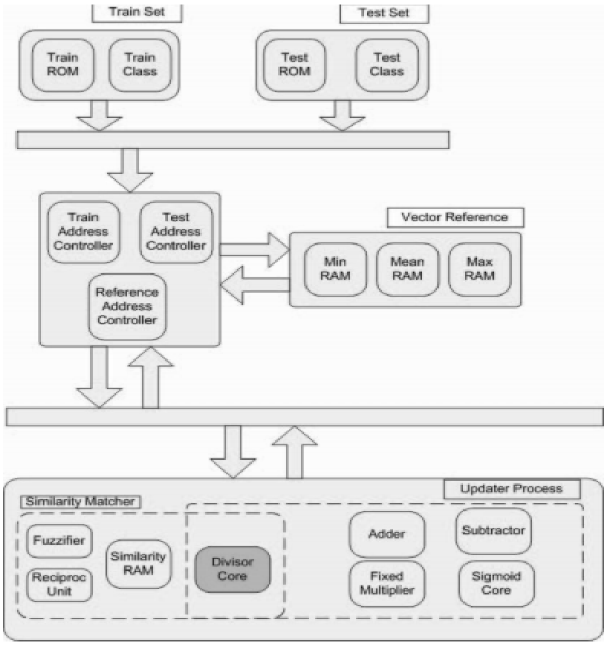

Figure. 3. FNGLVQ core in FPGA, adapted from previous research [10].

The structure of the implementation as shown in Figure 2 is considered as an expansion of our previous works in Wavelet-FLVQ for arrhythmia detection design [14] and FLVQ for Trichloroethylene estimation in white mouse liver image design [15]. The fundamental modified parts are the addition of several units such as the sigmoid derivative computer and change in number format used from 12 bit fixed point to 32 bit fixed point. After designing the whole system with the required components, then it was implemented using register transfer level state machine. The state machine is shown in Figure 4. Those phases consist of several state machines named *init_reference_vector*, *update_reference_vector, next_epoch*, *next_vector, pool*, *compute_distance*, *compute_membership*, *find_miss_classification_error*. Furthermore, we use additional components such as *comparator*, *shifter*, *multiplier*, *divider*, *derivative*, *adder*, and *subtractor.* After the implementation, we can get the FPGA resources utilization of the design. The utilization table is shown in Table (TABEL UTILISASI).

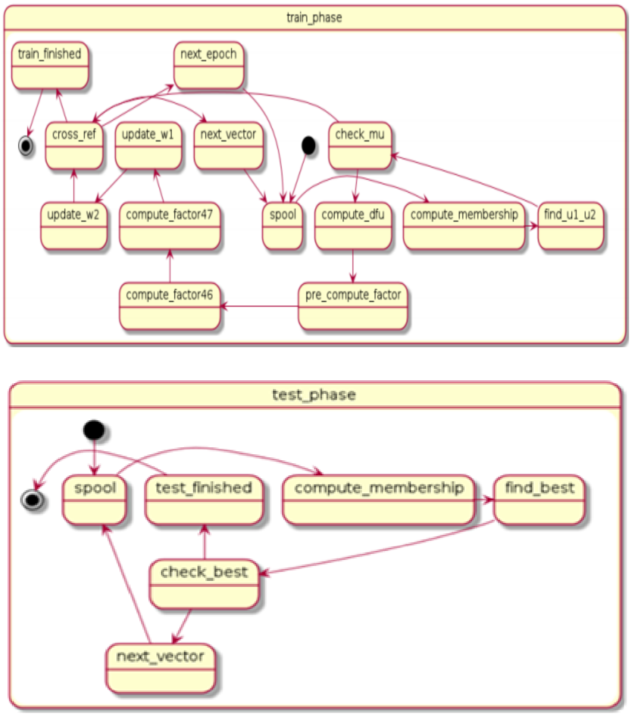

Figure. 4. State Machine

Table 2. Resource utilization

Logic component	Available	Used	Utilization (%)
Slice flip flop	11.776	2684	22%
4 input LUTs	11.776	3063	26%
Occupied slices	5.888	2336	39%
Bonded IOBs	372	26	6%
BUFGMUXs	24	4	16%
DCMs	8	2	25%
MULT18X18SIOs	20	4	20%
RAMB16BWEs	20	3	15%

5. Experiment Result and Discussion

To test the performance of our design, we measure the accuracy and running time. Accuracy of the classifier is calculated on the simulation program. Furthermore, we also compare the result to the MATLAB implementation. We use various epochs starting from 1, 2, 4 until 64. Then, we computed the training process running time of the simulation on both MATLAB and FPGA side. We used personal computer with core i7 processor, and 4 GB RAM to run the simulation. View of the simulation process is shown in figure 5. The result of the experiment is shown in table 3

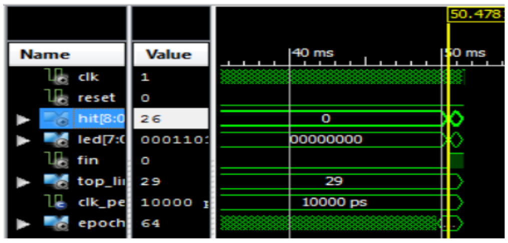

Figure. 5. Simulation process

Table 3. Experiment Result

Epoch	M	T_M	S	T_S	$T_{S_{sim}}$
1	93,33%	0,028 s	90,00%	965 μs	~00:16.650
2	93,33%	0,044 s	93,33%	1796 μs	~00:28.710
4	93,33%	0,077 s	93,33%	3430 μs	~00:52.950
8	93,33%	0,143 s	83,33%	6720 μs	~01:41.520
16	93,33%	0,275 s	80,00%	13428 μs	~03:18.110
32	93,33%	0,536 s	83,33%	26395 μs	~06:24.490
64	93,33%	1,065 s	86,67%	50478 μs	~12:10.660
Mean	93,33%		87,14%		

In table 3, M column represents the accuracy of the MATLAB side and T_M column represents the running time on MATLAB side. S and T_{SIM} columns represent accuracy and running time on simulation respectively while T_S is the running time on FPGA side. Table 3 shows that the mean accuracy on simulation is 87.14% while accuracy on MATLAB is 93.33%. Therefore, accuracy on simulation side is less than accuracy on MATLAB side. However, running time on the FPGA side is approximately twenty time faster than on MATLAB side.

6. Conclusion

This paper proposed design and simulation of FNGLVQ-PI classifiers in FPGA. The design consists of train and test phase that consists of several states. The design has 87.14% accuracy for Iris dataset. The accuracy is worse than the accuracy of the MATLAB implementation of 93.33%. However, the running time on FPGA is approximately twenty times faster than the MATLAB implementation.

Acknowledgement

This work is supported by Strengthening International Research-Based Collaboration Grant Universitas Indonesia No. 1403/H2.R12/HKP.05.00/Perjanjian/2014

References

1. Alpaydin, Ethem. *Introduction to machine learning*. MIT press, 2004.
2. Mitchell, Tom M. "Artificial neural networks." *Machine learning* (1997): 81-127.
3. T. Kohonen, “Learning Vector Quantization for Pattern Recognition,” Report TKK-F-A601, Helsinki University of Technology, Espoo, Finland, 1986.
4. A. Sato and K. Yamada, “A formulation of learning vector quantization using a new misclassification measure,” in Proceedings of the 14th International Conference on Pattern Recognition-Volume 1 - Volume 1, ICPR ’98, (Washington, DC, USA), pp. 322–, IEEE Computer Society, 1998
5. Setiawan, I.M.A.; Imah, E.M.; Jatmiko, W., "Arrhytmia classification using Fuzzy-Neuro Generalized Learning Vector Quantization," *Advanced Computer Science and Information System (ICACSIS), 2011 International Conference on* , vol., no., pp.385,390, 17-18 Dec. 2011.
6. Akbar, M.A; Suryana, M.E.; Imah, E.M.; Agus, IM.; Jatmiko, W., "Modified Fuzzy-Neuro Generalized Learning Vector Quantization for early detection of Arrhytmias," *Advanced Computer Science and Information Systems (ICACSIS), 2012 International Conference on* , vol., no., pp.293,299, 1-2 Dec. 2012.
7. Von Herzen, B., "Signal processing at 250 MHz using high-performance FPGA's," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on* , vol.6, no.2, pp.238,246, June 1998
8. Alimohammad, A; Fard, S.F., "FPGA-Based Bit Error Rate Performance Measurement of Wireless Systems," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on* , vol.22, no.7, pp.1583,1592, July 2014
9. Daijin Kim, "An implementation of fuzzy logic controller on the reconfigurable FPGA system," *Industrial Electronics, IEEE Transactions on* , vol.47, no.3, pp.703,715, Jun 2000
10. Eka S, M.; Fajar, M.; Iqbal T, M.; Jatmiko, W.; Agus, IM., "FNGLVQ FPGA design for sleep stages classification based on electrocardiogram signal," *Systems, Man, and Cybernetics (SMC), 2012 IEEE International Conference on* , vol., no., pp.2711,2716, 14-17 Oct. 2012
11. K. Bache and M. Lichman. UCI machine learning repository,2014.https://archive.ics.uci.edu/ml/datasets/Iris
12. Xilinx. Spartan-3AN FPGA Family Data Sheet, ds557 edition, April 2011
13. Xilinx. Spartan-3A/3AN FPGA Starter Kit Board User Guide, ug334 edition,June 2008.
14. Xilinx. Spartan-3E FPGA Family Data Sheet, ds312 edition, Juli 2013.
15. L. Zadeh. Fuzzy sets. Information and Control, 8(3):338 – 353, 1965.