Makara Journal of Technology

Volume 20 | Number 2 Article 5

8-2-2016

Automatic Arrhythmia Beat Detection: Algorithm, System, and Implementation

Wisnu Jatmiko
Faculty of Computer Science, Universitas Indonesia, Depok 16424, Indonesia, wisnuj@cs.ui.ac.id

I Md. Agus Setiawan
Faculty of Computer Science, Universitas Indonesia, Depok 16424, Indonesia

Muhammad Ali Akbar
Faculty of Computer Science, Universitas Indonesia, Depok 16424, Indonesia

Muhammad Eka Suryana
Faculty of Computer Science, Universitas Indonesia, Depok 16424, Indonesia

Yulistiyan Wardhana
Faculty of Computer Science, Universitas Indonesia, Depok 16424, Indonesia

See next page for additional authors

Recommended Citation

Jatmiko, Wisnu; Setiawan, I Md. Agus; Akbar, Muhammad Ali; Suryana, Muhammad Eka; Wardhana, Yulistiyan; and Rachmadi, Muhammad Febrian (2016) "Automatic Arrhythmia Beat Detection: Algorithm, System, and Implementation," *Makara Journal of Technology*: Vol. 20 : No. 2 , Article 5.
DOI: 10.7454/mst.v20i2.3060
Available at: https://scholarhub.ui.ac.id/mjt/vol20/iss2/5

Makara J. Technol. 20/2 (2016), 82-92
doi: 10.7454/mst.v20i2.3060

Automatic Arrhythmia Beat Detection: Algorithm, System, and Implementation

Wisnu Jatmiko[1*], I Md. Agus Setiawan[1], Muhammad Ali Akbar[1], Muhammad Eka Suryana[1], Yulistiyan Wardhana[1], and Muhammad Febrian Rachmadi[2]

1. Faculty of Computer Science, Universitas Indonesia, Depok 16424, Indonesia
2. School of Informatics, University of Edinburgh, Edinburgh EH8 9LE, United Kingdom

[*]*e-mail: wisnuj@cs.ui.ac.id*

Abstract

Cardiac disease is one of the major causes of death in the world. Early diagnose of the symptoms depends on abnormality on heart beat pattern, known as Arrhythmia. A novel fuzzy neuro generalized learning vector quantization for automatic Arrhythmia heart beat classification is proposed. The algorithm is an extension from the GLVQ algorithm that employs a fuzzy logic concept as the discriminant function in order to develop a robust algorithm and improve the classification performance. The algorithm is tested against MIT-BIH arrhythmia database to measure the performance. Based on the experiment result, FN-GLVQ is able to increase the accuracy of GLVQ by a soft margin. As we intend to build a device with automated Arrhythmia detection, FN-GLVQ is then implemented into Field Gate Programmable Array to prototype the system into a real device.

Abstrak

Deteksi Otomatis Detak Arrhythmia: Algoritma, Sistem dan Implementasi. Penyakit jantung adalah salah satu penyebab utama kematian di dunia. Diagnosa awal dari penyakit tersebut didasarkan pada ketidaknormalan pola detak jantung, atau disebut juga sebagai Arrhythmia. Sebuah metode baru bernama fuzzy neuro generalized learning vector quantization (FN-GLCQ) kami kembangkan untuk melakukan klasifikasi terhadap pola tersebut. Algoritma tersebut dimodifikasi dari algoritma GLVQ dengan mengimplementasikan konsep fuzzy logic sebagai fungsi diskriminan untuk meningkatkan performa klasifikasi. Algoritma tersebut dites dengan menggunakan MIT-BIH arrythmia database untuk dianalisis performanya. Berdasarkan uji coba yang telah dilakukan, FN-GLVQ dapat meningkatkan akurasi dari performa GLVQ. Sesuai dengan tujuan kami untuk membangun alat pendeteksian otomatis, FN-GLVQ juga diimplementasikan pada Field Gate Programmable Array sebagai prototipe dari perangkat yang akan dibangun.

Keywords: arrhythmia, learning vector quantization, FN-GLVQ*, FPGA*

Introduction

Automatic heart beat classifications based on ECG has attracted much attention from researchers because automatic classification can save cardiologist time from looking for arrhythmia beats in sheer amount of ECG heart beats data [1-4]. Similar research was held by Ince *et al*. [5]. In that research, most normal heartbeat would lie on the main bell of a Gaussian curve while abnormal beat would lie on the tail part of the curve. Another Arrhythmia classification research is held by Setiawan *et al*. which is fuzzy-neuro generalized learning vector quantization [6].

Learning vector quantization (LVQ), which was introduced by Kohonen, has a simple learning mechanism, yet robust to be applied in a variety of areas such as Arrhythmia [7] and handwritten digit recognition [8]. Another extension of LVQ that has been exhaustively researched [9] is called fuzzy neuro learning vector quantization (FNLVQ). This method is specifically researched in the field of odor type recognition. Its performance was also compared with another variant named fuzzy algorithm LVQ with the same data where both algorithm provides high recognition. However, FNLVQ has another advantage in term of unknown class identification. Jatmiko *et al*. proposed a further extension of FNLVQ with matrix similarity analysis as a further enhancement for odor type recognition [10]. The latest modification was done by Setiawan *et al.*, who introduced fuzzy concept into GLVQ algorithm, namely fuzzy-neuro generalized learning vector quantization (FN-GLVQ) [11], specifically for arrhythmia beat recognition.

Researches related to ECG beat detection typically showed high accuracy performance regardless of its dataset and its classification method. There are various mechanism to extract ECG features, such as morphological characteristic like amplitude [12] and Hermite based function [13]. There is also study by Sani *et al.* in sleep apnea detection from ECG signal on its optimal features, principal components, and nonlinearity [14]. In recent years, there are also popular methods which are used to transform the features into fewer features such as Wavelet Transform. Wavelet Transform has been used in various fields such as denoising [1], feature extraction [15], or feature reduction [7]. Related to feature reduction, T. Mar *et al.* research showed that sequential forward floating search algorithm could only select less 10 features or less to achieve a good result [16]. Usually, feature extraction is a separate process from the classification method. However, an attempt was made to integrate feature extraction inside the classifier as demonstrated by Elly *et al.* [17]. In order to handle huge amount of signal data from 24-hour treatment of patient with critical condition, Sani *et al.* implemented signal compression [18].

In neural network's FPGA implementation, Armato *et al.* presented a system on FPGA using Self Organizing Map (SOM) [19]. We have a similar experience in implementing Arrhythmia beat detection on into a low cost FPGA using FN-GLVQ algorithm [20]. Aside from FPGA, Oresko *et al.* proposed a Robust smart phone based ECG detection system in which, it has achieved a good result [3].

In this paper, we presented an automated framework for Arrhythmia beat detection. The framework consists of three steps. They are data preprocessing, feature extraction, and classification. In the classification step, a novel fuzzy neuro generalized learning vector quantization is used to classify each beat. The developed classifier is then embedded into real hardware, and Field Programmable Gate Array (FPGA) is chosen to prototype the model. Conceptual illustration of the built prototype can be seen at Figure 1.

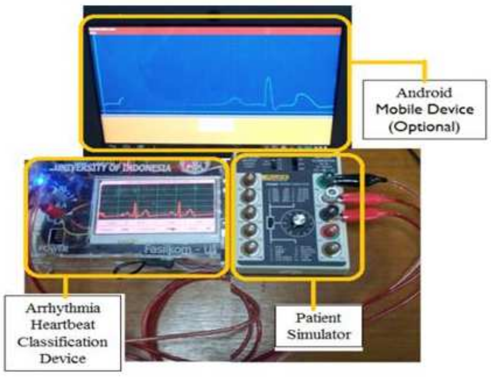
Android Mobile Device (Optional)
Arrhythmia Heartbeat Classification Device
Patient Simulator

Figure 1. System Overview

The rest of the paper is organized as follows: In section II, we show our overall system overview. Section III describes the dataset and feature extraction step employed in this research. Section IV describes the classification technique, the FN-GLVQ algorithm and explaining how the algorithm was derived from GLVQ. Section V describes the design and implementation of FPGA. Section VI and VII shows the experiment results, conclusion, and discuss the future plans of our study.

Dataset and feature extraction step. MIT-BIH arrhythmia database (MITDB) from physionet [21,22] is freely available and had been used by many scholars as a benchmark data to test the performance of the classifier. The MIT-BIH arryrhmia database has a recording of 48 different recordings from subjects in various age group and gender. The 48 records has a 30 minutes from two channels, which originates from the upper MLII lead and lower lead V1/V2/V4/V5. The ECG data have a frequency of 360Hz. MLII lead was chosen as our source data based on [21] that reported normal heartbeats were hard to distinguish in the from the lower head. From a total of 15 available classes, three of them were removed because it has a small number of occurrences. These remover classes are ventricular flutter wave, nodal (Junctional) escape, and paced beat.

Baseline wander removal. The initial step of the preprocessing is removing the baseline noise, or baseline wander. Baseline noise occurs when the ECG signal do not lie or sit on the isoelectric line. This happened because there is a low frequency noise that alters the signal up or down for several degrees during the ECG recording process. There have to be several steps conducted to remove this occurence and prevent miss-interpretation to the ECG signal, as mentioned in [11].

Beat extraction. The next step in the preprocessing process is to extract the individual ECG beats. The original continuous sequential ECG beats will be segmented to seperate ECG beats. The width of these individual beats will be approximated into 300-sample data points and centered on the peak of the R.

To conduct this transformation, we used the annotated data available in the database. The pivot point of each beat is the R peak annotation. Each beat will be a cut off from the continous signal of each R-Peak. The start of the beat will be at the R-150 position and the end peak will be the R+149 position. The total width of a single beat will be 300 sample data.

Outlier removal. The final step in this preprocessing stage is the outlier removal. This process is done because not all of the beats inside the arrythmia beat category is categorized as arrythmia beats. To remove outliers from the heartbeat collection, we ultilized a simple technique called Inter Quartile Range (IQR). The IQR used the percentile information to determine the outlier. In this research, we used the 25th percentil as the lower (Q_1) and 75th percentile as the upper quartile (Q_3). The range between thos quartiles is calculated using the following formula:

$$IQR = Q_3 - Q_1 \tag{1}$$

The outlier boundary's extremity levels are defined as:

$$\begin{aligned} LowLevel &= Q_3 - 1.5 \times IQR \\ UpLevel &= Q_3 + 1.5 \times IQR \end{aligned} \tag{2}$$

The boundary will be applied to every feature on the data. The beats which are outside the extremity level is considered as an outlier and it is removed from the dataset. This process assumes that the correlation for each features is not taken into account.

Feature extraction. In order to extract the feature, we utilized discrete wavelet transformation so that every feature is extracted in the pre-processed individual beat data. The definition of a signal *f*(x) Wavelet Transform is:

$$W_s f(x) = f(x) * \psi_s(x) = \frac{1}{s}\int_{-\infty}^{+\infty} f(t)\psi(\frac{x-t}{s})\mathrm{d}t \tag{3}$$

Where s denotes the scaling factor, basic wavelet dilation $\psi(x)$ is $\psi_s(x) = \frac{1}{s}\psi(\frac{x}{s})$ using scaling factor s. Let $s = 2j$ ($j \in Z$, Z is the integral set), then the WT is called dyadic WT [23]. The dyadic WT of a digital signal *f*(*n*) is calculated with Mallat algorithm:

$$S_{2j} f(n) = \sum_{k \in Z} h_k S_{2j-1} f(n - 2^{j-i} - k) \tag{4}$$

$$W_{2j} f(n) = \sum_{k \in Z} g_k S_{2j-i} f(n - 2^{j-i} - k) \tag{5}$$

Where the smoothing factor is denoted by the $S_2 j$, $S_2 jf(n)$ = a_j a_j denotes the low the low frequency coefficients. The low frequency coefficients are the approximation of the original signal, while the high frequency coefficients that is the detail of the original signal is denoted by $W_2 jf(n) = d_j$; d_j [23]. An important part of wavelet theory is to select the correct number of mother wavelet to still obtain the important information of the wavelet and the coefficients of the wavelet. The decomposition level of the wavelet is also an important part of wavelet theory. Senhadji *et al*. [24] recommended Daubechies as the mother wavelet since it gave the best performance for Arrhythmia recognition. Based on our study, Daubhecies order 8 (db8) showed better results than the others. Hence our research will focus on db8 as mother wavelet and level 1 to 5 will be decomposed as our individual.

Fuzzy-neuro generalized learning vector quantization (FN-GLVQ). Fuzzy-Neuro GLVQ is a classification algorithm proposed by Setiawan *et al*. [11]. This is a new classification method that adopted the concept of fuzzy similarity from Fuzzy Learning Vector Quantization algorithm (FLVQ) [9], [10], combined with the concept of cost minimization function from GLVQ [25,26]. The detailed concept of FN-GLVQ can be seen at [27].

2. Experimental

Preprocessing tasks. Although the core algorithm is implemented in the FPGA, the system still needs an external processing since the data is not streamed online from a real patient. The illustration of the process can be seen on Figure 2. The left side illustrates that all preprocessing works on software, while the right side illustrates the processes implemented on the FPGA. After all the features have been extracted, the outlier is removed by applying the Generalized ESD Many-Outlier Procedure [28], then data are reordered using round-robin selection as required by FN-GLVQ. Then, the data are divided up equally and put them into a training set and a testing set. Initial vector reference was built by choosing random features from a training set of each class. The data were standardized first before being transformed into fixed point 32-bit as number format representation for board development.

Design of arithmetic circuit. In 2004, Basterretxea *et al.* proposed to use the centered linear approximation (CRI) to obtain the sigmoid derivative computer core [29]. We can use the parameter q to approximate the level of approximation. In this research, we chose q = 4 to gain the optimum precision. Deschamps *et al.* explain that the non-restoring division algorithm could be used to obtain the divisor compute core [30]. As Non-Restoring algorithm is intended to process integer numbers, a further preprocessing adjustment is needed. After the preprocessing, it could be then operated in fixed-point numbers. To operate in fixed-point numbers, the numerator

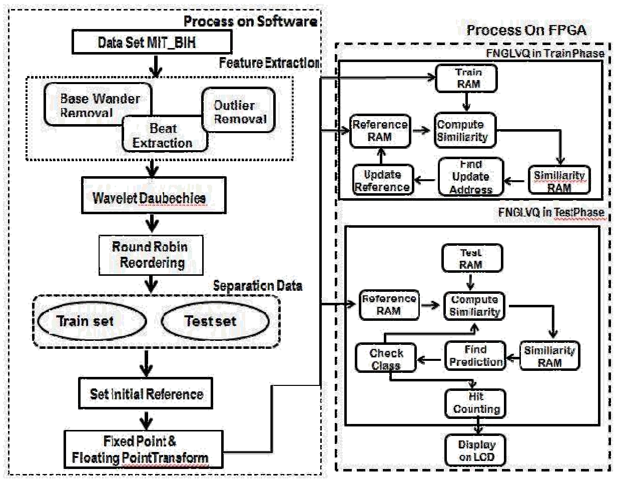

Figure 2. Arrhythmia Processing Framework

should be shifted left 16 times. Afterwards, the result should be shifted right 16 times to retain the original width. The remainder part of the circuit can be safely omitted later. This process is illustrated on the Equation 7.

$$Q = \frac{X \ll 16}{Y} \tag{6}$$

$$Q = Q \gg 16 \tag{7}$$

Resources allocation for each arithmetic operator, delay needed, and the number of created instances can be seen in Table 1.

System design consideration. The design of FN-GLVQ has already considered both training and testing functions of FN-GLVQ. The internal Block RAM stores the dataset operations and the Distributed RAM stores the vector references. Different from Block RAM, Distributed RAM can store the data as Look up Tables. For every update process, the update value will be written on the Reference RAM.

Our previous work includes designing Wavelet - FN-GLCQ to detect arrhythmia [31] and also to detect and estimate the level of Trichloroethylene inside a mouse liver [32]. Some fundamental modification of this implementation includes the additional design of sigmoid derivative core, expanding the data range from 12 bit to 32 bit, and the removal of operations that uses floating point computations. The learning phase state machine design can be seen on Figure 3.

GLVQ implementation. In this research, we have also implemented GLVQ in FPGA. FN-GLVQ can be considered as an extension version of GLVQ, by applying little change on updating rules and substituting Euclidean distance computation with the Fuzzy Similarity. Therefore, the main architecture is quite similar. However, GLVQ architecture is simpler and is more resource efficient. To implement GLVQ, we downgrade FN-GLVQ by discarding unnecessary components based on the algorithm.

Functionality verification. The functionality verification process that we have taken can be seen in Figure 4. This process starts by running behavioral simulation for each atomic component with different functionalities. The component is fed with virtual input, then the result output is monitored and the delay is computed. Post place and route (PAR) simulation is executed to know the exact delay taken during component execution. In most cases, running post PAR simulation for a large component may take excessive time. Hence, after verifying the behavioral correctness, it could be obtained that only the

Table 1. Realized Arithmetic Circuit Instances

Unit	Delay (Clock)	Slices	Instances
Divisor	20	1167	1
Sigmoid Core	8	471	1
Adder	1	32	56
Subtractor	1	32	25
Multiplier	1	32	4

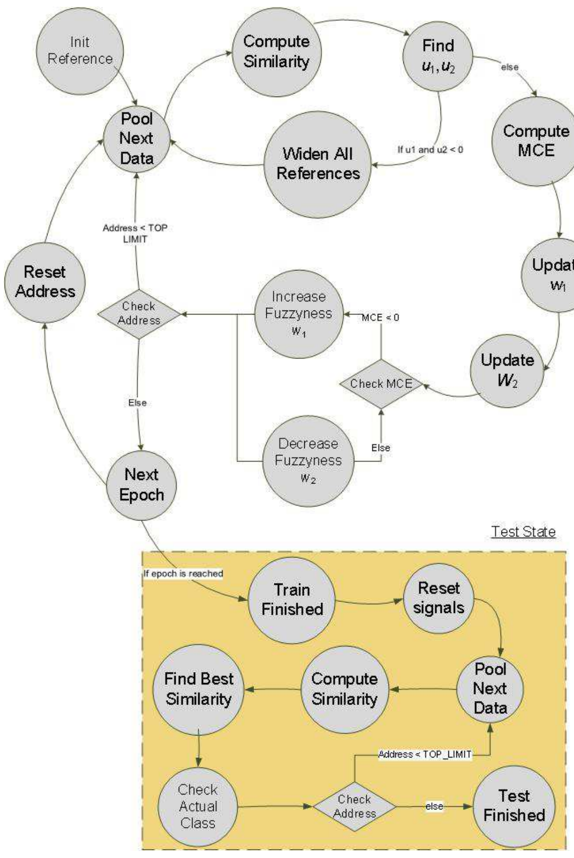

Figure 3. FN-GLVQ Train State Machine

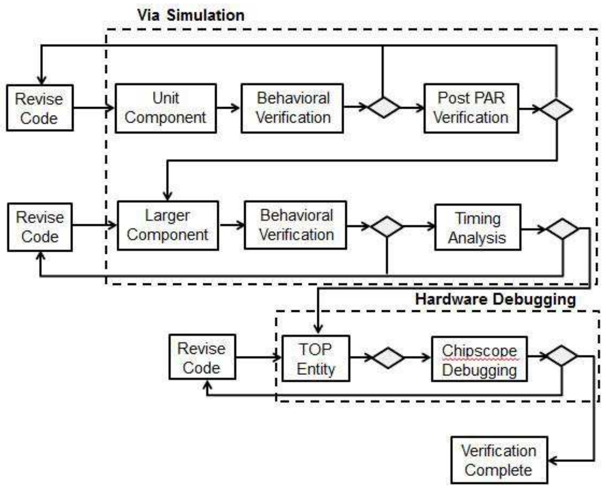

Figure 4. Functional Verification Check

exact delay is needed and this is achieved via timing analysis. The correctness was checked again via the real time debugging method using Xilinx Chipscope.

3. Results and Discussion

The experiment of this research was held with five scenarios. The first scenario is performed to find the best features among four variants of Daubechies wavelet: db2, db4, db6, and db8. The second scenario is carried out to verify the capability of the proposed classifier, as well as its predecessor in recognizing ECG beat from MITDB. The third scenario is performed to show the robustness of the proposed classifier against the unbalance dataset. The fourth scenario is statistical tests. The last scenario focuses on the verification correctness of the designed FPGA logic through the accuracy report.

Performance evaluation on software side. This experiment equally separates the processed dataset into two groups, training and testing. A maximum of 2000 data for each class were selected to reduce the data distribution over the class categories 2 (Table 2). A special setup was done beforehand with the dataset sequence. This setup arranges the dataset sequence in the training process, where it requires a round robin class. This only focuses on one reference vector for each class category.

Test wavelet daubechies. As pointed out in section 2.4, we chose Daubhecies as the mother wavelet throughout this experiment because of its performance superiority among others, especially for arrhythmia recognition. The trial was conducted to determine the best Daubechies configuration which is gave the best performance with regards to the Daubechies variant and the Daubechies decomposition level (Table 3). Examined Daubechies variants are db2, db4, db6 and db8, while decomposition levels are examined up to level 5. Table 3 shows the number of features for each Daubechies variant/level. The learning parameter for the classifier can be seen in Table 4. The experiment will be performed by the means of GLVQ and FN-GLVQ.

To simplify the examination, only six dominant classes of Arrhythmia without outlier were considered. The data were then split evenly for training and testing respectively. The testing procedures were performed 10 times for each data and then the average of all those attempts is calculated to gain the overall accuracy for each configuration.

From the above test scenario, the overall accuracy were obtained as seen in Table 5. Table 5 shows that by using db8, both classifiers can obtain better performance among other variants. In addition, from Table 5 we can also see that the optimum decomposition levels for db8 were Level 1 with 157 features and Level 2 with 86 features.

Classifier performance evaluation. In order to evaluate our learning method, 10-Fold Cross Validation with ratio 50:50 between the training set and the testing set were performed for both the GLVQ as well as FN-GLVQ methods (Table 6). Learning parameters were then obtained from passing a series of tests, as can be seen in Table 6. The number of features used is 86.

From Table 7 it could be seen that our proposed method, the FN-GLVQ, produces a better performance in terms of accuracy compared to its predecessor, GLVQ. FN-GLVQ produces 95.52% in average for K = 10, which is better than GLVQ, 93.36%.

Table 2. The Number of Beats per Class Arrhythmia for Experiment

Beat code	Beat name	Num data	Trainset	Test set
‘NOR’	‘Normal Beat’	2000	1000	1000
‘LBBB’	‘Left Bundle Branch Block Beat’	2000	1000	1000
‘RBBB’	‘Right Bundle Branch Block Beat’	2000	1000	1000
‘PVC’	‘Premature ventricular Contraction’	2000	1000	1000
‘P’	‘Paced Beat’	1898	949	949
‘AP’	‘Atrial Premature Beat’	1236	618	618
‘fVN’	‘Fusion of Ventricular and Normal Beat’	412	206	206
‘fPN’	‘Fusion of Paced and Normal Beat’	105	53	52
‘NE’	‘Nodal (junctional) premature Beat’	148	74	74
‘aAP’	‘abberated Atrial Premature Beat’	61	31	30
‘VE’	‘ventricular Escape Beat’	103	52	51
‘NP’	‘Nodal (junctional) premature Beat’	37	19	18
		12000	6002	5998

Table 3. The Number of Features for Each Level of Wavelet Daubechies

Level			Daubechies	Wavelet
	db2	db4	db6	db8
L1	151	153	155	157
L2	77	80	83	86
L3	40	43	47	50
L4	21	25	29	32
L5	12	16	20	23

Table 4. Parameter Used for Wavelet Test

	GLVQ	FN-GLVQ
α	0.05	0.05
epoch	150	150
Inisial bobot	random	random

Table 5. The Results of Test Accuracy between GLVQ and FN-GLVQ

Level	GLVQ			
	db2	db4	db6	db8
L1	95.24	96.51	96.27	96.68
L2	96.05	96.38	96.06	96.3
L3	93.6	93.22	93.43	93.67
L4	89.95	92.55	93.3	92.77
L5	86.99	81.39	85.75	85.07
Avg	92.37	92.01	92.96	92.9
Level	FN-GLVQ			
	db2	db4	db6	db8
L1	97.49	98.13	97.86	98.09
L2	97.98	98.09	97.99	97.96
L3	97.52	97.74	97.38	97.64
L4	96.41	96.56	96.95	96.72
L5	93.89	94.35	91.67	94.28
Avg	96.66	96.66	96.37	96.94
GLVQ	92.37	92.01	92.96	92.9
FN-GLVQ	96.66	96.66	96.37	96.94
Avg	94.51	94.49	94.67	94.92

Table 6. Parameter for the Learning Process

Alg.	α	Max Epoch
GLVQ	0.075	150
FN-GLVQ	0.01	150

Evaluation on the unbalanced dataset. The neural network in general works best for balanced data, meaning the distribution among each class do not differ much. From this motivation, we tested the algorithm against Arrhythmia data after outlier removal with an unbalance condition, as shown in Table 2. The learning parameters used for this scenario were the same as the previous scenarios, and the number of features used is 86. The subsequent result for the unbalanced data test for 12 classes can be seen in Table 8.

From Table 8 it can be seen that FN-GLVQ obtains a better accuracy compared to other algorithms. It reaches an accuracy of 96.35%. To evaluate the performance of a classifier against unbalance datasets, we have measured the Recall, Precision, F-Measure, Sensitivity, Specificity, and its G-Mean value. The average recall of FNLGVQ gives a better result, which were spread evenly for each class category, up to 86.23%, ±4% higher than GLVQ. Meanwhile, F-Measure value shows that FN-GLVQ also gives a better result, up to 89.31%, ±7% higher than GLVQ. G-Mean value also shows that FN-GLVQ gives better performance, 92.33%, ±4% higher than GLVQ. Table 9 shows distribution of recognition level for each class category.

Statistical tests. The two algorithms will be tested statistically using 3 kinds of statistical t-test: k-hold out paired t-test, k-fold cross validation paired t-test, and Dietterichs 5x2-fold cross validation paired t-test. All these statistical tests were based on cross validation method. After several tests, this method will select the smallest estimated risk [33]. To conduct this statistical test, we used 6 and 12 classes with 86 features dataset, and also using the best parameters have been obtained in the previous scenario.

Table 7. Performance Result using 10-Fold Cross Validation

K	GLVQ	FN-GLVQ
1	93.25	95.57
2	93.19	95.74
3	93.56	95.59
4	93.16	95.39
5	93.46	95.75
6	93.32	95.52
7	92.98	95.30
8	94.45	95.53
9	93.30	95.65
10	92.96	95.19
avg	93.36	95.52

Table 8. Performance Result for Unbalanced Dataset

Algorithm	GLVQ	FN-GLVQ
Max Epoch	150	150
α	0.075	0.01
Total Acc	94.08	96.35
Recall	82.12	86.23
Precision	84.46	93.49
F-Measure	82.73	89.31
Sensitivity	82.12	86.23
Specificity	99.42	99.64
G-Mean	88.68	92.33

Table 9. Performance Results for the Unbalance Dataset for Each Class Category (%)

Classes	N	GLVQ	FN-GLVQ
C#0	1000	89.20	95.30
C#1	1000	95.70	98.00
C#2	1000	97.60	99.50
C#3	1000	91.20	93.70
C#4	949	99.89	99.89
C#5	618	98.87	99.68
C#6	206	88.35	92.72
C#7	52	94.23	86.54
C#8	74	93.24	68.92
C#9	30	13.33	73.33
C#10	51	62.75	54.90
C#11	18	61.11	72.22

K-Hold out paired t-Test. In this test, the data will be split into two subsets, the training subset which from 2/3 of the data and the testing subset which from 1/3 of the data. The classifier testing accuracy will be denoted as P_A and P_B respectively. The test will be conducted K times, where the number K that is commonly used is 30. The testing subscripts from the tests are tagged with (i), (i) = 1, …, *K*. Therefore, at the end of the testing, we will have a set of K, $P^{(1)} = P_A^{(1)} - P_B^{(1)}$ to $P^{(K)} = P_A^{(K)} - P_B^{(K)}$. This is assuming that the set is an independently drawn sample from a normal distribution. From the null hypotheses (H_0: where there was no significant difference in accuracy, or equal accuracies), the following statistics has a t-distribution with *K-1* degrees of freedom.

$$t = \frac{\bar{P}\sqrt{K}}{\sqrt{\sum_{i=1}^{K}(P^{(i)} - \bar{P})^2/(K-1)}} \qquad (8)$$

where $\bar{P} = (1/K)\sum_{i=1}^{K} P^{(i)}$. The calculated t will decide whether we reject the H_0 (therefore, accepting that there are significant differences between the two classifier models) or not. This is done by comparing the tabulated value for the chosen level of significance and $K - 1$ degrees of freedom and the calculated t. If the t is greater, that rejects H_0. However, because there is an assumption that the independently drawn sample is invalid, Dietterich argues that it could lead to deceptive results. A drawback is found where there are overlapping data sets in each of the *K* runs when training the classifier model and estimating testing accuracy and making the differences dependent [37]. Generally in the comparison of the algorithm, the level of significancy 0.05 was used. If the value of $|t| > z$ with z is the tabular value is obtained, then the H_0 can be rejected, and accept that there is a significant difference between the two algorithms were compared.

In this test, the number of iterations performed K = 30 where the dataset is partitioned into two equal (50:50) for training and testing, while the initialization weights using the method of internal random dataset. The result of the test can be seen in Table 10. Tabulation of data obtained are shown in Table 10. From the data tabulation K-hold Score at the Tables 10, *t*6 obtained for 6 classes is -76.672141, while the t_{12} for 12 classes obtained -70.245154. Therefore the level of significancy K-hold was used for testing is 0.05 (95%) and 29 (*K*-1) degrees of freedom. The values was obtained from the tabulation table is 2.045. For 6 classes, the null hypothesis H_0 can be rejected, because $|t_6| > 2.045$. And for the 12 classes, the value $|t_{12}| > 2.045$, so H_0 can be rejected.

Table 10. K-Hold Test Result P_A = GLVQ and P_B = FN-GLVQ

Classes	List of accuracy			
	6 classes		12 classes	
	P_A	P_B	P_A	P_B
0	97.32	98.82	94.52	97.28
1	97.49	98.8	94.3	97.33
2	97.43	98.76	94.42	97.12
3	97.56	98.83	94.23	97.25
4	97.56	98.83	94.60	97.10
5	97.50	98.78	94.47	97.53
6	97.59	98.87	94.13	97.42
7	97.36	98.87	94.30	97.47
8	97.50	98.83	94.50	97.28
9	97.50	98.71	94.58	97.47
10	97.58	98.89	94.15	97.43
11	97.50	98.92	94.17	97.32
12	97.41	98.78	94.50	97.33
13	97.49	98.85	94.52	97.25
14	97.54	98.78	94.35	97.23
15	97.50	98.76	94.43	97.30
16	97.50	98.83	94.68	97.07
17	97.47	98.83	94.58	97.38
18	97.49	98.78	94.78	97.43
19	97.49	98.82	94.48	97.18
20	97.56	98.85	94.42	97.40
21	97.58	98.92	94.47	97.18
22	97.11	98.80	94.20	97.27
23	97.56	98.90	94.48	97.12
24	97.49	98.87	94.28	97.08
25	97.52	98.76	94.15	97.30
26	97.52	98.87	94.23	97.10
27	97.43	98.80	94.40	97.53
28	97.52	98.82	94.23	97.37
29	97.49	98.82	94.55	97.30
AVG	97.49	98.83	94.4	97.29

K-Fold Score	
6 Classes	12 Classes
pBar: -0.013357	pBar: -0.028906
t6: -76.672141	t12: -70.245154

After we have conducted the normal tests using the Shapiro-Wilk normality test, there is an anomaly at the *p*-value for *p*-value P_{A6}. The *p*-value for P_{A6} is below the value 0.005, this indicates that the data P_{A6} is not normal. So we need to do a more sensitive statistical test, such as Wilcoxon signed rank. Given the *p*-value for P_{A6}, P_{B6}, P_{A12}, P_{B12}, is 1.524e-05, 0.6734, 0.3234, and 0.3267 respectively. By using an application R, for data 6 classes, obtained *p*-value = 1.807e-06. With a *p*-value <0.05, we can conclude that H_0 can also be rejected.

From these results, we could see the difference in accuracy of the two algorithms. The difference is very significant and it could be concluded *FN-GLVQ* algorithm gives better results compared to GLVQ.

K-Fold cross validation paired t-Test. The K-fold cross validation is an alternative from the previous procedure. This procedure avoids the flaw of the previous procedure, which is the testing data overlap 8. In this test, the dataset used is partitioned into 10 (K = 10), then, it is iterated 10 times with the composition of the training data to testing data is 90:10. Afterwards, weights were initialized using the internal random manner from the dataset. After the trial tabulation, the data obtained are shown in Table 11.

Table 11. K-fold Cross Validation Test Result P_A = GLVQ and P_B = *FN-GLVQ*

Classes	List of accuracy			
	6 classes		12 classes	
	P_A	P_B	P_A	P_B
0	96.77	98.47	93.5	96.5
1	95.78	98.65	96.17	97.08
2	97.76	98.92	94.33	96.42
3	97.76	98.56	94.42	96.33
4	97.67	98.56	94.42	96.33
5	96.77	98.38	93.42	96.25
6	97.04	97.85	94.67	96.25
7	97.4	99.01	94.17	96.00
8	95.69	98.11	93.33	95.75
9	97.67	98.65	93.58	96.67
AVG	97.03	98.52	94.13	96.33

K-Fold Score	
6 Classes	12 Classes
pBar: -0.014901	pBar: -0.022000
t6: -6.664332	t12: -10.255489

From the tabulation data K-Fold Score in Table 11 t_6 obtained for 6 classes is -6.664332, while the t_{12} for 12 classes obtained is -10.255489. Therefore, the level of K-Fold significance is 0.05 (95%) and 9 (K - 1) degrees of freedom. The value that is obtained from the tabulation data is 2.262 for 6 classes, the null hypothesis H0 can be rejected, because $|t_6| > 2.262$. And for the 12 classes, the value $|t_{12}| > 2.262$, so H_0 can be rejected.

However, after we have done the normal tests using the Shapiro-Wilk normality test, there is an anomaly at the *p*-value P_{A6} and P_{A12}. Given the P-Value for P_{A6}, P_{B6}, P_{A12}, is 0.04851, 0.6884, 0.04248, and 0.9324 respectively. This indicates that the data P_{A6} and P_{A12} is not normal. We also utilized the Wilcoxon signed rank to prove that the null hypothesis can be rejected. By using an application R, for data of 6 classes, we have obtained the *p*-value = 0.005889, while the results for 12 classes is 0.001953. Both results showed that the charge of *p*-value below the value <0.05, we can concluded that H_0 also can be rejected. From these results, it can be concluded that the difference in accuracy of the two algorithms is very significant, and therefore it could be summed that the *FN-GLVQ* algorithm gives better results compared to GLVQ.

Dietterichs 5 2-Fold cross validation paired t-Test. This procedure repeats the two-fold cross-validation procedure tested five times [34]. The data will be split into two parts, training and testing halves. This is done for each cross-validated run. Suppose we have two classifier models, classifier A and classifier B. The two classifiers will be trained on the first half the data, and tested on the second half of the data. The observed accuracy will be $P^{(1)} = P_A^{(1)} - P_B^{(1)}$ respectively. If the training and testing B halves is swapped, the estimates will be $P^{(2)} = P_A^{(2)} - P_B^{(2)}$. The AB mean and variance of the difference estimation is:

$$\bar{P} = \frac{P^{(1)} + P^{(2)}}{2};\ s^2 = (P^{(1)} - \bar{P})^2 + (P^{(2)} - \bar{P})^2 \qquad (9)$$

Let $P_i^{(1)}$ denote the difference $P^{(1)}$ in the *i*th run, and $S_i^{(2)}$ denote the estimated variance for run *i*; *i* = 1; …; 5. The proposed t statistics becomes [34]:

$$\bar{t} = \frac{P^{(1)}}{\sqrt{(1/5)\sum_i^5 = 1 s_i^2}} \qquad (10)$$

Only one from the 10 values difference obtained will be used in the calculation statistics. In this method, the charge *t* approaches t distribution with the 5 degree of freedom.

In this test, the dataset is partitioned into 2-Fold with 5 iterations as described above. The initial weights was initiated using internal random manner from the dataset. The data tabulation obtained from the tests is shown in Table 12 and Table 13.

Table 12. Dietterichs 5x2 Folds

The Tabulation of 6 Class Data				
K	P_{A1}	P_{B1}	P_{A2}	P_{B2}
K:0	96.39	98.13	96.37	98.29
K:1	96.57	98.06	96.44	98.47
K:2	96.73	98.17	96.35	98.4
K:3	96.8	98.13	96.3	98.37
K:4	96.61	98.24	96.37	98.24

From Table 13 Dietterichs Score, the value of t_6 is -4.846367, and for 12 classes, the value of t_{12} is -3.910283. The significance level used for K-Hold test is 0.05 (95%) with 5 degree of freedom. The tabulation value obtained from the table is 2.571 for 6 classes; therefore, H_0 can be rejected, because the value of $|t_6| > 2.571$. For the value of 12 classes the value is also above the 2.571 ($|t_{12}| > 2.571$). Normality test shows the data is normal. So this also shows that the difference in accuracy of both algorithms is very significant and can be concluded that the *FN-GLVQ* gives better results as compared to GLVQ.

Evaluation on hardware implementation. Several processes to reduce the data were conducted. First, only six dominant classes of Arrhythmia are considered. Afterwards, the reminder of the data is further decomposed using Daubechies 8, until 24 features were remained. Finally, stratified sampling was applied until only 198 instances are left with initial class distribution intact. This whole process is repeated a few times until five sub-sample sets are left. Table 14 shows the data instances after the final selection step where it is stored into Block RAM.

FN-GLVQ was implemented with fixed point component to process the subsamples. The fixed point number is selected to be implemented in FPGA's implementation because it is more simple and consuming lesser resources.

In FPGA implementation, the performance is gradually degraded when the learning time goes over 64 epochs. This happens because the fixed point losses many precision when the number is divided by a near zero value. This problem can be anticipated with increasing the bit length. Yet, it would also increase resources consumption. Therefore, learning time is set at 64 epoch with the learning rates of 0.03125, 0.0625, 0.125, 0.25 are to be tested. The testing result of the FN-GLVQ's FPGA implementation can be seen at Table 15. The time diagram of the implementation can be seen at Figure 5.

The FN-GLVQ's FPGA result shows that the classifier gave around 73%-75% with the best accuracy on α= 0.0625. This result would be compared with high level implementation and GLVQ's FPGA implementation to compare the FN-GLVQ's FPGA performance. Meanwhile The FN-GLVQ's and GLVQ's FPGA is implemented with the fixed point number, the high level language implementation is implemented with floating point number. The GLVQ's FPGA and high level implementation is tested with α = 0.0625, based on the best α on FN-GLVQ's FPGA implementation with the same set of dataset to be tested.

Table 16 shows the accuracy of FN-GLVQ implementation, GLVQ implementation and the high level language implementation of FN-GLVQ which is tested with similar subsample data. This result showed FN-GLVQ still performs better for all dataset with moderate significance when compared to GLVQ. The average differences between both implementations reached up to 7%. The result confirms the same conclusion when it was experimented on the software. Thus, the FN-GLVQ's FPGA implementation had worse results compared to the high level implementation's results. The average differences between both implementations reached up to 6%. That case happened because the data type which is used in high level implementation has wider range that the FPGA implementation.

Table 13. Dietterichs 5x2 Folds 12 Classes

The Tabulation of 12 Classes				
K	P_{A1}	P_{B1}	P_{A2}	P_{B2}
K:0	93.08	95.73	93.45	95.22
K:1	92.88	95.9	93.56	95.22
K:2	93	95.72	93.51	95.43
K:3	92.81	95.7	93.43	95.28
K:4	92.83	95.62	93.4	95.2

Dietterichs Score	
6 Classes	12 Classes
$t_6 = -4.846367$	$t12 : -3.910283$

Table 14. Subsample data

	class 1	class 2	class 3	class 4	class 5
Instances	40	40	40	40	38

Table 15. The Classifier Performance on FPGA

Set n	0.03125	0.0625	0.125	0.25
set 1	75.13%	72.40%	70.77%	67.21%
set 2	78.14%	73.77%	78.14%	77.05%
set 5	74%	73.5%	70%	66.47%
set 8	75.2%	80.4%	78%	75.2%
set 10	73.33%	79.17%	80%	82.67%
mean	75.16%	75.85%	75.38%	73.72%

Table 16. The Classifier Performance on FPGA

	set 1	set 2	set 5	set 8	set 10	Mean
FN-GLVQ	75.14	78.14	74	75.2	73.33	75.16
GLVQ	65.38	62.82	69.51	73.08	70.73	68.3
High level	82.83	80.81	81.82	80.81	80.81	81.41

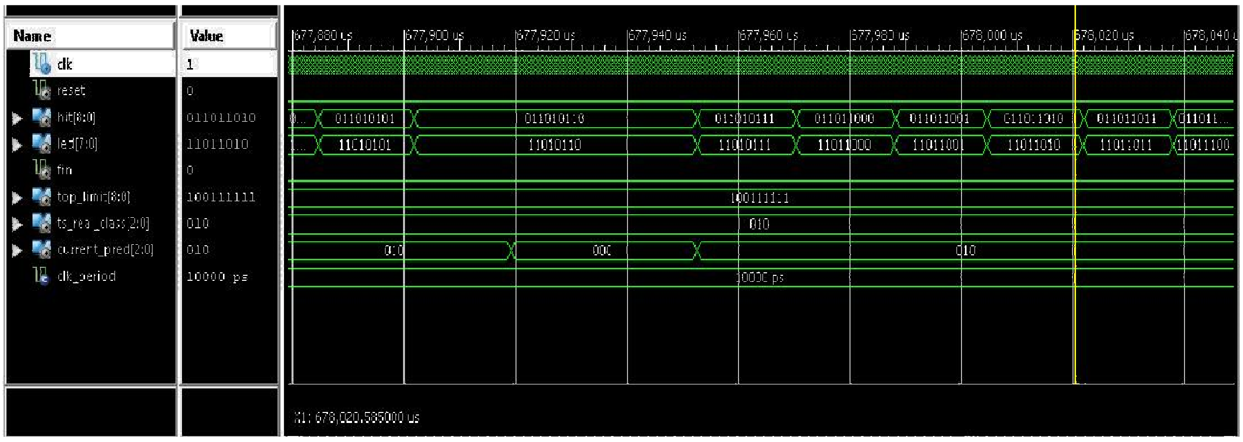

Figure 5. Time Diagram of FN-GLVQ's FPGA Implementation

Resource consumption analysis. Resources utilization to realize FN-GLVQ can be seen on Table 17 along with each number representation format. Synthesizer was configured to optimize speed as the resources has been optimized via the designed hardware implementation logic. The operation division, that utilizes the division unit, consumes 20% of the board's resources. This is the most expensive operation; therefore, single instance of division unit was created where each division operation needs to take turn, which is manageable with the designed state machine.

Running time analysis. In this scenario, a dataset of 366 instances were applied to a classifier which is configured to run at 64 epochs, configured with $_{2}{}^{1}{}_{4}$ learning rate. An epoch of GLVQ update process took about 5.89 ms which is constant for the whole processing. The total running time was 768 ms. An epoch of FN-GLVQ update process took about 12.34 ms. The total running time based on average chance of update process activation taken 1169.89 ms. Processing time was also tested on the software. Based on the experiment, the time taken was 576 ms. Table 18 summarized the running time log.

Table 17. Device Utilization Summary Post Place and Route

Logic Utilization	Fixed Point	Floating Point
Number of Slice Flip Flops	15.00%	32.00%
Number of 4 input LUTs	96.00%	89.00%
Number of occupied Slices	98.00%	95.00%
Total Number of 4 input LUTs	98.00%	90.00%
Number of bonded IOBs	6.00%	6.00%
Number of BUFGMUXs	16.00%	16.00%
Number of DCMs	25.00%	25.00%
Number of MULT18X18SIOs	70.00%	60.00%
Number of RAMB16BWEs	95.00%	95.00%

Table 18. Running Time Log

GLVQ	FN-GLVQ	Software
768 ms	1169.89 ms	576 ms

4. Conclusions

We have shown the mathematical derivation of FN-GLVQ from GLVQ. The overall processing framework applied to the data has also been explained. Three statistical testing were conducted on GLVQ and FN-GLVQ algorithms on Arrhythmia of 6 classes and 12 classes, 86 features data show that FN-GLVQ gives a significantly better result than GLVQ algorithms. Benchmarked against MITDB, FN-GLVQ successfully improved recognition rate over GLVQ with a considerate margin. We have begun initial development of wearable device by prototyping FN-GLVQ into FPGA where the design was tested with actual hardware. Subsequent result also showed the developed layout functionally correct and exhibit close performance with software result with average margin of 6%.

This research still works in progress, with the final goal is to build fully a automated system capable for both measuring ECG signal and Arrhythmia classification. A separate analog device for capturing ECG signals from a living patient had been built, the design of FPGA based FN-GLVQ has already shown. Still, a lot of work need to be done specifically data preprocessing and feature extraction which can utilize the FPGA implementation.

Acknowledgement

This work is supported by "Hibah Riset Kolaborasi Internasional" No. 1403/H2.R12/HKP.05.00/2014 by Direktorat Perguruan Tinggi 2014. It is also supported by "Penelitian Unggulan Perguruan Tinggi" No.0478/ UN2.R12/HKP.05.00/2015 by "Kemenristek Dikti".

References

[1] M. Faezipour, A. Saeed, S. Bulusu, M. Nourani, H. Minn, L. Tamil, Inf. Tech. Biomed. IEEE Trans. 14/5 (2010) 1153.

[2] O. Sayadi, M. Shamsollahi, G. Clifford, Biomed. Eng. IEEE Trans. 57/2 (2010) 353.

[3] J. Oresko, Z. Jin, J. Cheng, S. Huang, Y. Sun, H. Duschl, A. Cheng, Inf. Tech. Biomed. IEEE Trans. 14/3 (2010) 734.

[4] F. Melgani, Y. Bazi, Inf. Tech. Biomed. IEEE Trans. 12/5 (2008) 667.

[5] T. Ince, S. Kiranyaz, M. Gabbouj, Biomed. Eng. IEEE Trans. 56/ 5 (2009) 1415.

[6] I.M.A. Setiawan, E.M. Imah, W. Jatmiko, Int. Conf. Adv. Comput. Sci. Inf. Sys. (ICACSIS), Jakarta, Indonesia, 2011, p.385.

[7] E.M. Imah, I.M.A. Setiawan, A. Febrian, W. Jatmiko, Int. Symp. Micro-Nano Mechatronics and Human Science (MHS), Nagoya, Japan, 2011, p.355.

[8] K. Takahashi, D. Nishiwaki, Proc. Seventh Int. Conf. Sofia, 2003, p.268.

[9] K. Budiarjo, B. Handoko, W. Jatmiko, ISA Trans. Sci. Eng. Meas. Autom, Elsevier, 41 (2002) 395.

[10] T. Kohonen, Proc. IEEE, 78/ 9 (1990) 1464.

[11] E.M. Imah, I.M.A. Setiawan, A. Febrian, W. Jatmiko, Int. Symp. Micro-Nano Mechatronics and Human Science (MHS), Nagoya, Japan, 2011, p.355.

[12] I. Jekova, G. Bortolan, I. Christov, Med. Eng. Phys. 30/2 (2008) 248.

[13] G. de Lannoy, D. Franois, J. Delbeke, M. Verleysen, in: A.L.N. Fred, J. Filipe, H. Gamboa (Eds.), Feature Relevance Assessment in Automatic Inter-Patient Heart Beat Classification, INSTICC Press, 2010, p.13.

[14] S.M. Isa, M.I. Fanany, W. Jatmiko, A.M. Arymurthy, 5th Int. Conf. Bioinformatics Biomed. Eng. (iCBBE), Wuhan, China, 2011, p.4.

[15] I. Romero, N. Grubb, G. Clegg, C. Robertson, P. Addison, J. Watson, Biomed. Eng. IEEE Trans. 55/11 (2008) 2658.

[16] T. Mar, S. Zaunseder, J. Martinez, M. Llamedo, R. Poll, Biomed. Eng. IEEE Trans. 58/8 (2011) 2168.

[17] E. Matul, W. Jatmiko, T. Basarudin, IEEE Int. Conf. Syst. Man and Cybernetics (SMC), Seoul, Korea, 2012.

[18] S.M. Isa, W. Jatmiko, A.M. Arymurthy, IEEE Int. Conf. Syst. Man and Cybernetics, Seoul, Korea, 2012, p.226.

[19] A. Armato, E. Nardini, A. Lanata, G. Valenza, C. Mancuso, E. Scilingo, D. De Rossi, Intelligent Systems Design and Applications, 2009. ISDA ’09. Ninth Int. Conf. 2009, p. 660.

[20] M.E. Suyana, M. Fajar, M. Iqbal, W. Jatmiko, I.M. Agus, IEEE Int. Conf. Syst. Man and Cybernetics (SMC), Seoul, Korea, 2012, p.2711.

[21] G. Moody, R. Mark, Eng. Med. Biol. Mag. IEEE, 20/3 (2001) 45.

[22] A.L. Goldberger, L.A. Amaral, L. Glass, J.M. Hausdorff, P.C. Ivaniv, R.G. Mark, J.E. Mietus, G.B. Moody, C.K. Peng, H.E. Stanley, Circulation, 101/23 (2000) e215.

[23] Z. Qibin, L. Zhang, Proc. 2005 Int. Conf. Neural Networks and Brain (ICNN&B '05), Beijing, 2005.

[24] L. Senhadji, G. Carrault, J. Bellanger, G. Passariello, Eng. Med. Biol. Mag. 14/2 (1995), p.167.

[25] A. Sato, K. Yamada, Adv. Neural Inf. Proc. Syst. 7 (1995) 423.

[26] P. Schneider, Ph.D. Dissertation, Rijksuniversiteit Groningen, Duitsland, 2010.

[27] M.E. Suyana, M. Fajar, M. Iqbal, W. Jatmiko, I.M. Agus, IEEE Int. Conf. Syst. Man and Cybernetics (SMC), Seoul, Korea, 2012, p.2711.

[28] B. Rosner, Technometrics, 25/2 (1983) 165.

[29] K. Basterretxea, J. Tarela, I. del Campo, IEE Proc. Circuit, Devices and Syst. 151(2004) 18.

[30] G.D.S. Jean-Pierre Deschamps, Gery Jean Antoine Bioul, Synthesis of Arithmetic Circuits FPGA, ASICS, and Embedded Systems. John Willey and Sons, 2006, ch. Chapter 6.

[31] W. Jatmiko, P. Mursanto, A. Febrian, M. Fajar, W.T. Anggoro, R.S. Rambe, M.I. Tawakal, F.F. Jovan, M. Eka, Int. Conf. Adv. Comput. Sci. Inf. Syst. (ICACSIS), Nagoya, Japan, 2011, p. 349.

[32] A. Febrian, M. Fajar, M.I Tawakal, E.M. Imah, W. Jatmiko, D.H. Ramdani, A. Bowolaksono, P. Mursanto, Int. Conf. Adv. Comput. Sci. Inf. Syst. (ICACSIS), Jakarta, Indonesia, 2011, p. 119.

[33] S. Arlot, A. Celisse, Stat. Surv. 4 (2010) 40.

[34] L.I. Kuncheva, Combining Pattern Classifiers: Methods and Algorithms, Ser. Willey Interscience. John Willey and Sons, 2004, ch. 1.