ECG Data Classification Using Clustering Algorithm and MLP Neural Network

Sharafeddin Toumajpour, Bahram Mohamad Maghsoudi
Department of Mechnical Engineering, Payame Noor University (PNU) P.O. Box19395-3697 Tehran, Iran
sharafeddin.toumajpour@gmail.com
Faculty of engineering islamic azad university of aliabad katool branch, b.m.maghsodi@gmail.com

Abstract- Electrocardiogram (ECG) which reveals the rhythm and function of the heart is an important non-invasive clinical tool for cardiologists to diagnose various heart diseases. The objective of this paper is to diagnose ECG arrhythmia classification based on the extracted features. Feature extraction and selection are critical to the quality of classifiers founded through data mining methods. To extract useful information and diagnose the ECG arrhythmia, a hybrid of clustering algorithm and multi layer Perceptron (MLP) neural network is developed. In clustering section, we used hybrid evolutionary optimization algorithm based on combining modify imperialist competitive algorithm (MICA) and K-means (K), which is called K-MICA. In the clustering section, the input data is first clustered by a K-MICA. Then the Euclidean distance of each ECG signal is computed from the determined clusters. These Euclidean distances of ECG signals are used as input of MLP neural network. In the test stage, 10-fold cross validation method has been applied to the MIT-BIH arrhythmia database for evaluating the capability of the proposed method. The simulation results show that the proposed method has high recognition accuracy.

Index Terms— Arrhythmia, MLP, ECG, K-MICA, Euclidean distance.

1 INTRODUCTION

The analysis of the ECG has been widely used for diagnosing many cardiac diseases. The development of accurate and quick methods for automatic ECG classification is vital for clinical diagnosis of the heart diseases. An arrhythmia is any abnormal cardiac rhythm [1], [2]. Among the various abnormalities related with functioning of the human heart, Premature Ventricular Contraction (PVC) is one the most important arrhythmias [3], [5].

In the literature, several methods have been proposed for the automatic classification of ECG signals. In [6], 2nd, 3rd and 4th order cumulants of the ECG beat calculated and modeled by linear combinations of Hermitian basis functions. Then, the parameters of each cumulant model used as feature vectors to classify five different ECG beats namely as Normal, PVC, APC, RBBB and LBBB using 1-Nearest Neighborhood (1-NN) classifier. Finally, after classifying each model, a final decision making rule applied to these specified classes and the type of ECG beat defined. In [7], the authors compared the performances of three approaches. The first approach used principal components of segmented ECG beats, the second approach used principal components of error signals of linear prediction model, whereas the third approach used principal components of Discrete Wavelet Transform (DWT) coefficients as features. These features from three approaches were independently classified using feed forward neural network (NN) and Least Square-Support Vector Machine (LS-SVM).

In [8], the authors described feature extraction methods using higher order statistics (HOS) of wavelet packet decomposition (WPD) coefficients for the purpose of automatic heartbeat recognition. The method consisted of three stages. First, the wavelet package coefficients (WPC) were calculated for each different type of ECG beat. Then, higher order statistics of WPC were derived. Finally, the obtained feature set was used as input to a classifier, which was based on k-NN algorithm. In [9], discrete wavelet transform used to extract the morphological features of ECG signals and a multi-class support vector machine (SVM)-based classifier employed to classify them. Then a genetic algorithm used for optimization of the relevant parameters of system. These parameters were: wavelet filter type for feature extraction, wavelet decomposition level, and classifier's parameters. In [10], the authors proposed a new power spectral-based hybrid genetic algorithm-support vector machines (SVMGA) technique to classify five types of electrocardiogram (ECG) beats, namely normal beats and four manifestations of heart arrhythmia.

In this paper, we have proposed an automated method for recognition of PVC heartbeats from a normal beat and the other ones. The proposed automated method for the classification of cardiac arrhythmias is based on clustering and classification. More details regarding the proposed method are described in next sections.

The rest of paper is organized as follows. Section 2 explains the general scheme of the proposed method. Section 3 and Section 4 describe the main parts of the proposed method, i.e., clustering technique and classifier module, respectively. Section 5, shows some simulation results and finally Section 6 concludes the paper.

2. General structure of the proposed method

In this paper, we have proposed an automated method for recognition of PVC heartbeats from a normal beat and the other ones. The proposed method includes two main modules: clustering module and the classifier module. Fig. 1 shows the general scheme of this method. In the clustering module, first the input ECG data will be clustered by K-MICA clustering technique. Then the Euclidean distance of each arrhythmia is computed from the determined clusters. It is used as the characteristic feature. It is obvious that the Euclidean distance of each ECG signal from its own cluster center is smaller than the value of Euclidean distance of that signal from other cluster

centers. Subsequently, each signal is shown as a 1*3 vector, which one of the rows is a small value and the remaining rows have a large value. Application of this approach causes that the dimension of the classifier is decreased. Also we have a more efficient feature that is better than the raw data. The classifier module determines the membership of the patterns using the computed distance. Next sections present the main modules.

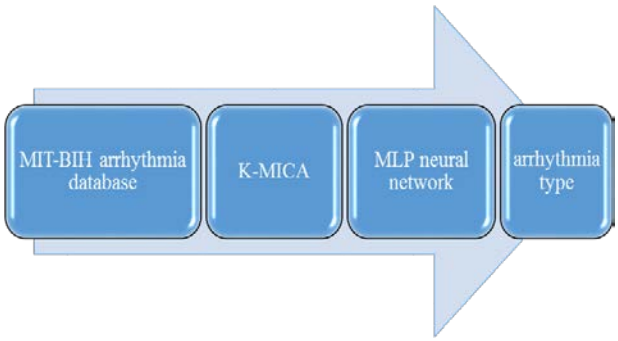

Fig. 1. General scheme of the proposed method

$$Population = \begin{bmatrix} X_1 \\ X_2 \\ .... \\ X_{N_{initial}} \end{bmatrix} \quad (2)$$

$$X_i = Country_i = [center_1, center_2, ..., center_k] \quad , \quad i = 1, 2, ..., N_{initial} \quad (3)$$

$$center_j = [x_1, x_2, ..., x_d] \quad , \quad j = 1, 2, ..., k$$

$$H = k \times d \quad (4)$$

$$X_0 = [X_0^1, X_0^2, ..., X_0^H]$$

$$x_0^j = rand(.) \times (x_{\max}^j - x_{\min}^j) + x_{\min}^j \quad , \quad j = 1, 2, ..., H \quad (5)$$

$$X_i = [x_i^1, x_i^2, ..., x_i^H] \quad , \quad i = 1, 2, ..., N_{initial}$$

$$x_i^j = 4 \times x_{i-1}^j \times (1 - x_{i-1}^j) \quad , \quad j = 1, 2, ..., H \quad (6)$$

$$x_j^{\min} < x_j < x_j^{\max} \quad (7)$$

3. K-MICA clustering technique

Clustering is an important problem that must often be solved as a part of more complicated tasks in pattern recognition, image analysis and other fields of science and engineering. Clustering procedures partition a set of objects into clusters such that objects in the same cluster are more similar to each other than objects in different clusters according to some predefined criteria.

K-means is used for its easiness and simplicity for applications [11]. However, it has some drawbacks. First, its result may depend on initial values. Also, it may converge to local minimum. Recently, numerous ideas have been used to alleviate this drawback by using global optimization algorithms such as GA [12] hybrid PSO-SA [13], hybrid PSO-ACO-K [14] and HBMO [15]. In this paper, we used method that delivered in [16] that called Hybrid K-MICA. According to original ICA, first primary population generates and then empires with their possessions take place. Applying K-means to each empire causes us to improve the initial population of colonies. It makes the hybrid algorithm converges more quickly and prevents it from falling into local optima. The outputs of K-means form the initial empires of modified ICA.

To improve the income of algorithm, it is better, that the powerless imperialist would not be removed when it loses all possessions. This imperialist is one of the best answers and that can be contributed in imperialistic competition as a weak colony or to be given to the powerful empire. To apply the ICA for clustering, the following steps have to be taken [16]:

Step 1: Generate an initial population

An initial population of input data is generated by chaos initialization as follows:

Where $center_j$ is jth cluster center for ith country. X_i is one of the countries. $N_{initial}$ is the number of population and d is the dimension of each cluster center. $x_j^{\max}$ And $x_j^{\min}$ (each feature of center) are the maximum and minimum value of each point referring to the jth cluster center which are in order. K is the number of clusters. H is the number of state variables. X_0 is an initial solution.

Step 2: Calculate objective function value

Suppose that we have N sample feature vectors. The objective function is evaluated for each country as follows:

Step 2-1: i=1 and $Objec$=0

Step 2-2: select the ith sample.

Step 2-3: calculate the distances between the ith sample and $Center_j$ (j=1,2,...,K).

Step 2-4: add the value of Objec with the minimum distance calculated in Step 2-3

($Objec$=$Objec$+min($|Center_i - Y_m|$, i=1,2,...,K)).

Step 2-5: if all samples have been selected, go to the next step, otherwise $i = i + 1$ and return to step 2-2.

Step 2-6: $Cost(X)$=$Objec$.

The objective function is calculated mathematically as below:

$$Cos(X) = \sum_{m=1}^{N} \min(|Center_i - Y_m|, \quad i=1,2,...,K) \quad (8)$$

(N=number of input data)

Step 3: Sort the initial population based on the objective function values.

The initial population is ascended based on the value of their objective function.

Step 4: Select the imperialist states.

Countries that have the minimum objective function are selected as the imperialist states and the remaining ones form the colonies of these imperialists.

Step 5: Divide colonies among imperialist.

Based on power of each imperialist the colonies are divided among them. The power of each imperialist calculated as followings:

$$C_n = \max\{\cos t\} - \cos t_n \quad (9)$$

$$P_n = \left| \frac{C_n}{\sum_{i=1}^{N_{imp}} C_i} \right| \quad (10)$$

$$C_n^{norm} = round(P_n(N_{col})) \quad (11)$$

In above equations, $\cos t_n$ is the cost of nth imperialist and C_n the normalized cost of each one. The normalized power of each imperialist introduced as P_n, then the initial number of colonies for each empire will be C_n^{norm} where N_{col} and N_{imp} are number of all colonies and imperialists.

Step 6: Use K-means algorithm for each empire.

Step 7: Move colonies toward their imperialist states as described in Section 3.

Step 8: Use mutation to change the direction of colonies. It is mentioned in modified ICA.

Step 9: Check the cost of all colonies in each empire.

During the previous steps cost of each colony might have changed. Check the cost of all colonies of an empire if there is one that have a lower cost than its relevant imperialist, exchange their position.

Step 10: Check total cost of each empire.

Cost of each empire depends on power of both imperialist and its colonies. It is calculated as follows:

$$TC_n = \cos t(imperialist_n) + \xi mean\{\cos t(colonies.of.empire_n)\} \quad (12)$$

TC_n is the total cost of nth empire, ξ is an attenuation coefficient between 0 and 1 to reduce the effect of colonies cost.

Step 11: Do imperialistic competition

All empires according their power (total cost), try to get colonies of weakest empire.

$$TC_n^{norm} = \max\{TC_i\} - TC_n \quad (13)$$

$$P_{P_n} = \left| \frac{TC_n^{norm}}{\sum_{i=1}^{N_{imp}} TC_i^{norm}} \right| \quad (14)$$

where TC_n^{norm} is normalized total cost of nth empire and the possession probability of each empire is P_{P_n}.

The roulette wheel can be used for stochastic selection of the winner empire which will dominate the weakest colony of weakest empire. To perform Roulette Wheel algorithm, it is necessary to calculate cumulative probability as follows:

$$C_{P_n} = \sum_{i=1}^{n} P_{P_n}$$

According to this equation cumulative probability for n=1 is equal to its probability, while for the last n it corresponds to one.then a random number with uniform distribution generates and compares with all C_{P_n}.

Each sector with higher probability will have more chance to be chosen. Therefore the winner empire will specify. as it is mentioned to use Roulette Wheel algorithm, computing cumulative distribution function is essential. To reduce this time consuming step an approach has been presented as below:

$$P = [P_{P_1}, P_{P_2}, ..., P_{P_{N_{imp}}}] \quad (15)$$

$$R = [r_1, r_2, ..., r_{N_{imp}}] \qquad r_1, r_2, ..., r_{N_{imp}} \approx U(0,1) \quad (16)$$

$$D = P - R = [D_1, D_2, ..., D_{N_{imp}}]$$

$$= [P_{P_1} - r_1, P_{P_2} - r_2, ..., P_{P_{N_{imp}}} - r_{N_{imp}}] \quad (17)$$

P is the vector of possession probability of all empires and R is a vector with uniformly distributed random numbers. Maximum index in D shows Winner Empire that gets the colony.

After realizing the winner empire, the weakest colony of the weakest empire will be given to the winner one. Then we should subtract one of the populations of this weak empire and add one to the winner's population.

Step 12: Remove weakest empire.

If there is any empire without colony, eliminate it. Replace one of the weakest colonies of best empire (low cost) with this imperialist.

Step 13: Apply chaotic local search (CLS) to search around the global solution. ICA has gained much attention and widespread applications in different fields. However, it often converges to local optima. In

order to avoid this shortcoming, a CLS algorithm is used to search around the global solution in the paper. CLS is based on the logistic equation as follows:

$$Cx_i = [Cx_i^1, Cx_i^2, ..., Cx_i^H]_{1\times H}, \quad i = 0,1,2,...,N_{chaos} \quad (18)$$

$$Cx_{i+1}^j = 4 \times Cx_i^j \times (1 - Cx_i^j), \quad j = 1,2,...,H$$

$$Cx_0^j = rand(.)$$

$$Cx_i^j \in [0,1], \quad Cx_0^j \notin \{0.25, 0.5, 0.75\}$$

In the CLS, the best solution is considered as an initial solution X_{ds}^0 for CLS. X_{cls}^0 s scaled into (0,1) according to the following equation:

$$X_{cls}^0 = [X_{ds,0}^1, X_{ds,0}^2, ..., X_{cls,0}^H]_{1\times H} \quad (19)$$

$$Cx = [Cx_0^1, Cx_0^2, ..., Cx_0^H,]$$

$$Cx_0^j = \frac{x_{cls,0}^j - x_{\min}^j}{x_{\max}^j - x_{\min}^j}, \quad j = 1,2,...,H$$

The chaos population for CLS is generated as follows:

$$X_{cls}^{i} = [X_{cls,i}^{1}, X_{cls,i}^{2}, ..., X_{cls,i}^{H}]_{1\times H} \quad (20)$$

$$i = 1,2,...,N_{chaos}$$

$$x_{cls,i}^{j} = cx_{i-1}^{j} \times (x_{\max}^{j} - x_{\min}^{j}) + x_{\min}^{j}$$

$$j = 1,2,...,H$$

where cx_i^j indicates the jth chaotic variable, N_{chaos} is the number of individuals for CLS. $rand(.)$ is a random number between zero and one.

The objective function is evaluated for all individuals of CLS. One country selected randomly is replaced with the best solution among them.

Step 14: Check number of empire.

If there is just one empire remained, stop. Else go to step 7.

4. MLP NEURAL NETWORK

An MLP neural network consists of an input layer (of source nodes), one or more hidden layers (of computation nodes) and an output layer. The recognition basically consists of two phases: training and testing. In the training stage, weights are calculated according to the chosen learning algorithm. The issue of learning algorithm and its speed is very important for the MLP model. In this study the following learning algorithms are considered.

4.1. Back propagation with momentum (BP with momentum)

The BP algorithm makes use of gradient descent with a momentum term to smooth out oscillation [17]. Eq. (21) gives the weight update for BP with momentum:

$$\Delta W_{ij}(t+1) = -\varepsilon \frac{\delta E}{\delta W_{ij}}(t) + \mu \frac{\delta E}{\delta W_{ij}}(t-1) \quad (21)$$

where w_{ij} represents the weight value from neuron j to neuron i, ε is the learning rate parameter, and E represents the error function. It adds an extra momentum parameter, μ , to the weight changes.

4.2. Resilient back propagation (RPROP) algorithm

RPROP considers the sign of derivatives as the indication for the direction of the weight update [17]. In doing so, the size of the partial derivative does not influence the weight step. The following equation shows the adaptation of the update values of Δ_{ij} (weight changes) for the RPROP algorithm. For initialization, all are set to small positive values:

$$\Delta_{ij}(t) = \begin{cases} \eta^{+} \times \Delta_{ij}(t-1); & if \frac{\delta E}{\delta W_{ij}}(t-1)\frac{\delta E}{\delta W_{ij}}(t) > 0 \\ \eta^{-} \times \Delta_{ij}(t-1); & if \frac{\delta E}{\delta W_{ij}}(t-1)\frac{\delta E}{\delta W_{ij}}(t) < 0 \\ \eta^{0} \times \Delta_{ij}(t-1); & otherwise \end{cases} \quad (22)$$

where $\eta^0 = 0, 0 < \eta^- < 1 < \eta^+, \eta^{-,0,+}$ are known as the update factors. Whenever the derivative of the corresponding weight changes its sign, this implies that the previous update value is too large and it has skipped a minimum. Therefore, the update value is then reduced (η^-) ,as shown above. However, if the derivative retains its sign, the update value is increased (η^+) .This will help to accelerate convergence in shallow areas. To avoid over-acceleration, in the epoch following the application of (η^+) , the new update value is neither increased nor decreased (η^0) from the previous one. Note that the values of Δ_{ij} remain non-negative in every epoch. This update value adaptation process is then followed by the actual weight update process, which is governed by the following equations:

$$\Delta W_{ij}(t) = \begin{cases} -\Delta_{ij}; & if \frac{\delta E}{\delta W_{ij}}(t) > 0 \\ +\Delta_{ij}; & if \frac{\delta E}{\delta W_{ij}}(t) < 0 \\ 0 \quad ; & otherwise \end{cases} \quad (23)$$

The values of the training parameters adopted for the algorithms were determined empirically.

4.3. Levenberg–Marquardt (LM) algorithm

The LM algorithm [18] uses the approximation to the Hessian matrix in the following Newton-like update:

$$w_{ij}(t+1) = w_{ij}(t) - \left[J^T J + \mu I \right]^{-1} J^T e \quad (24)$$

where J is the Jacobian matrix, e a vector of network errors and μ a constant.

4.4. scaled conjugate gradient algorithm (SCG),

The scaled conjugate gradient algorithm (SCG), developed by Moller [19], was designed to avoid the time-consuming line search. This algorithm combines the model-trust region approach (used in the Levenberg-Marquardt algorithm, described in Levenberg-Marquardt), with the conjugate gradient approach. See [19] for a detailed explanation of the algorithm.

4.5. Conjugate gradient back propagation with Fletcher-Reeves updates (CGBFR)

More detail regarding the Conjugate gradient back propagation with Fletcher-Reeves updates can be found in [20].

4.6. BFGS quasi-Newton back propagation (BFGSQB)

Newton's method is an alternative to the conjugate gradient methods for fast optimization. In optimization, quasi-Newton methods (a special case of variable metric methods) are algorithms for finding local maxima and minima of functions. Quasi-Newton methods are based on Newton's method to find the stationary point of a function, where the gradient is 0. Newton's method assumes that the function can be locally approximated as

a quadratic in the region around the optimum, and uses the first and second derivatives to find the stationary point. In higher dimensions, Newton's method uses the gradient and the Hessian matrix of second derivatives of the function to be minimized.

In quasi-Newton methods the Hessian matrix does not need to be computed. The Hessian is updated by analyzing successive gradient vectors instead. Quasi-Newton methods are a generalization of the secant method to find the root of the first derivative for multidimensional problems. In multi-dimensions the secant equation is under-determined, and quasi-Newton methods differ in how they constrain the solution, typically by adding a simple low-rank update to the current estimate of the Hessian. More detail regarding the BFGS quasi-Newton back propagation can be found in [21].

4.7. One-step secant back propagation (OSS)

The one step secant (OSS) method is an attempt to bridge the gap between the conjugate gradient algorithms and the quasi-Newton (secant) algorithms. This algorithm does not store the complete Hessian matrix; it assumes that at each iteration, the previous Hessian was the identity matrix. This has the additional advantage that the new search direction can be calculated without computing a matrix inverse.

More detail regarding the OSS can be found in [22].

5. Simulation Results

In this section, the performance of proposed classifier is evaluated. For this purpose, we have used the practical and real world data [23]. The used dataset contains 10000 examples of ECG signals. In order to compare the accuracy of proposed system, the k-fold cross validation technique is used. The k-fold cross validation technique was employed in the experiments, with k=10. The data set was thus divided into 10 portions, with each part of the data sharing the same proportion of each class of data. 9 data portion were used in the learning phase, while the remaining part was applied in test phase. The MLP-training methods were run 10 times to permit each part of the data to take turn as a testing data. The recognition accuracy rate is computed by summing the individual accuracy rate for each run of testing, and then dividing of the total by 10. All the obtained results are the average of 50 independent runs. The computational experiments for this section were done on Intel core i7 with 8 GB RAM using Dell computer. The computer program was performed on MATLAB (version R2012) environment.

5.1. Performance comparison of different training algorithms with row data

First we have evaluated the performance of the recognizer with row data. The training parameters and the configuration of the MLP used in this study are shown in Table 1. The MLP classifiers were tested with various neurons for a single hidden layer and the best networks are selected.

TABLE 1. MLP ARCHITECTURE AND TRAINING PARAMETER

Network structure	Value of parameters
Number of layers	2
Number of neurons in output layer	3
Training algorithm	Back-propagation with momentum
	RPROP
	LM
	SCG
	CGBFR
	BFGSQB
	OSS
Initial weights	Random
Transfer function in hidden layer	Tangent-sigmoid
Transfer function in output layer	Linear

Table 2 shows the recognition accuracy (RA) of different systems. In this table, NNHL means the number neurons in the hidden layers. The obtained results are the average of 50 independent runs. As it is depicted in Table 2, using various training algorithms and raw data, the highest accuracy is 96.37%, which is achieved by LM training algorithms.

Table 2
Recognition accuracy different systems using row data

Training algorithm	RA (%)	NNHL	Run time (Sec)	Standard deviation
Back-propagation with momentum	94.81	19	6	8.3
RPROP	95.65	20	4	7.2
LM	**96.37**	**20**	**5**	**6.5**
SCG	96.28	20	4	7.8
CGBFR	95.28	14	8	6.3
BFGSQB	94.15	24	11	8.3
OSS	95.33	15	8	9.8

5.2. Performance of proposed method

In next section we apply K-MICA for finding the optimum cluster centers. The parameters of the clustering algorithm used in this study are shown in Table 3. These values were selected for the best performance after several experiments.

Table 3. K-MICA parameters for clustering

Parameters	Values
N_{pop}	30
N_{imp}	8

γ	0.4
ζ	0.1
β	15
Max iteration	1000

Table 4 shows the recognition accuracy of different systems. As it is depicted in Table 3, using various training algorithms and proposed features as input of MLP, the highest accuracy is 99.38%, which is achieved by RPROP training algorithms. Comparison between Table 1 and Table 3 shows the efficiency of proposed feature.

Table 4. Recognition accuracy different systems using proposed feature

Training algorithm	RA (%)	NNHL	Run time (Sec)	Standard deviation
Back-propagation with momentum	97.63	16	3	2.1
RPROP	**99.38**	**20**	**1**	**0.6**
LM	98.52	16	2	2.5
SCG	99.03	20	1	1.2
CGBFR	98.81	18	2	7.4
BFGSQB	98.93	22	6	3.8
OSS	98.67	18	3	4.6

5.3. Performance comparison of different clustering algorithms

The performance of the classifiers has been compared with different clustering algorithms. For this purpose, fuzzy C-mean clustering (FCM) algorithm, K-mean clustering algorithm and genetic algorithm clustering [24] techniques are considered. Tables 6 to 7 show the RA of different systems. From these tables and table 3, it can be seen that K-MICA clustering based system has better recognition accuracy (99.38%)

Table 5. Recognition accuracy different systems using proposed feature (FCM clustering)

Training algorithm	RA (%)	NNHL	Run time (Sec)	Standard deviation
Back-propagation with momentum	97.27	15	3	3.1
RPROP	**98.78**	**13**	**1**	**1.8**
LM	98.31	18	2	2.8
SCG	97.99	22	1	1.2
CGBFR	98.36	12	2	7.8
BFGSQB	98.26	15	6	3.2
OSS	97.53	10	3	6.3

Table 6. Recognition accuracy different systems using proposed feature (K-Means clustering)

Training algorithm	RA (%)	NNHL	Run time (Sec)	Standard deviation
Back-propagation with momentum	97.21	14	3	2.8
RPROP	**98.96**	**19**	**1**	**1.5**
LM	98.37	15	2	3.6
SCG	98.55	10	1	4.2
CGBFR	98.27	13	2	5.2
BFGSQB	97.82	15	6	4.9
OSS	97.76	18	3	5.8

Table 7. Recognition accuracy different systems using proposed feature (Genetic algorithm clustering)

Training algorithm	RA (%)	NNHL	Run time (Sec)	Standard deviation
Back-propagation with momentum	97.03	10	3	2.8
RPROP	**99.04**	**17**	**1**	**1.2**
LM	98.37	21	2	1.9
SCG	98.27	20	1	2.7
CGBFR	97.29	15	2	3.9
BFGSQB	98.25	15	6	5.2
OSS	98.33	19	3	4.9

6. Conclusion

Accurate and fast approaches for automatic ECG data classification are vital for clinical diagnosis of heart disease. This study has investigated the design of an automatic and accurate system for recognition of ECG arrhythmias. In this study, the usage of Euclidean distance of patterns from cluster centers are proposed as efficiency input of classifier. The proposed input (Euclidean distance of signals from cluster center) are applied to MLP neural network that using MLP neural network with RP learning algorithm, the highest rate of accuracy 99.38% is achieved .for surveying the effect of clustering algorithm over system accuracy, various kind of clustering algorithms are performed which best performance is related to K-MICA algorithm.

References

[1] Looi J., Wong C., Lee M., Khan A., Webster M., Kerr A.: Usefulness of ECG to differentiate Takotsubo cardiomyopathy from acute coronary syndrome. International Journal of Cardiology, 199, 132-140 (2015).

[2] Do D., Hayase J., Tiecher R., Bai Y., Hu X., Boyle N.: ECG changes on continuous telemetry preceding in-hospital cardiac arrests. Journal of Electrocardiology, 48, 1062-1068 (2015).

[3] Padhy S., Sharma L., Dandapat S.: Multi lead ECG data compression using SVD in multi resolution domain. Biomedical Signal Processing and Control, 23, 10-18 (2016).

[4] Arango C., aguilar J., Vettorazzi M., Martínez-Bovi R.: ECG concentrations, luteal structures, return to cyclicity, and postabortion fertility in embryo transfer recipient mares. Theriogenology, 84, 1003-1013(2015).

[5] Atwood D., Wadlund D.: ECG Interpretation Using the CRISP Method: A Guide for Nurses. AORN Journal, 102, 396-408 (2015).

[6] S. Karimifard and A. Ahmadian, A robust method for diagnosis of morphological arrhythmias based on Hermitian model of higher or-

der statistics, BioMedical Engineering OnLine 10:22, 2011.

[7] R. J. Martis, U. R. Acharya, K.M. Mandana, A.K. Ray and C. Chakraborty, Application of principal component analysis to ECG signals for automated diagnosis of cardiac health, Expert Systems with Applications 39, 11792–11800, 2012.

[8] Y. Kutlua and D. Kuntalpb, Feature extraction for ECG heartbeats using higher order statistics of WPD coefficients, computer methods and programs in biomedicine 105, 257–267, 2012.

[9] A. E. Zadeh and A. Khazaee, High Efficient System for Automatic Classification of the Electrocardiogram Beats, Annals of Biomedical Engineering, 2011.

[10] A.Khazaee, A. Ebrahimzadeh, Classification of electrocardiogram signals with support vector machines and genetic algorithms using power spectral features, Biomedical Signal Processing and Control 5, pp. 252–263, 2010.

[11] A.K. Morales, F.R. Erazo. A search space reduction methodology for data mining in large databases. Engineering Applications of Artificial Intelligence 2009; 22: 57–65.

[12] K. Krishna, M. Murty. Genetic k-means Algorithm. IEEE Transactions on Systems, Man and Cybernetics B Cybernet 1999; 29: 433–439

[13] T. Niknam, B. Amiri, J. Olamaie, A. Arefi. An efficient hybrid evolutionary optimization algorithm based on PSO and SA for clustering. Journal of Zhejiang University Science 2009; 10: 512–519.

[14] B. Firouzi, M.S. Sadeghi, T. Niknam. A new hybrid algorithm based on PSO, SA, and K-means for cluster analysis. International Journal of Innovative Computing Information and Control 2010; 6: 1–10.

[15] M. Fathian, B. Amiri. A honey-bee mating approach on clustering. The International Journal of Advanced Manufacturing Technology 2007; 38: 7-8, 809–821.

[16] T.Niknam, E. Taherian Fard, N. Pourjafarian, A. Rousta. An efficient hybrid algorithm based on modified imperialist competitive algorithm and K-means for data clustering. Engineering Application of Artificial Intelligence 2011; 24: 306-317

[17] S. Haykin . Neural networks: a comprehensive foundation. New York: MacMillan; 1999.

[18] Hagan M, Menhaj M. Training feed-forward networks with the Marquardt algorithm. IEEE Trans. Neural Networks 1994;5:989–93.

[19] M. Moller. A Scaled Conjugate Gradient Algorithm for Fast Supervised Learning. 1990

[20] Scales, L.E., Introduction to Non-Linear Optimization, New York, Springer-Verlag, 1985

[21] The Numerical Algorithms Group. "Keyword Index: Quasi-Newton". NAG Library Manual, Mark 23. Retrieved 2012-02-09.

[22] Battiti, R., "First and second order methods for learning: Between steepest descent and Newton's method," Neural Computation, Vol. 4, No. 2, 1992, pp. 141-166.

[23] https://www.physionet.org/physiobank/database/mitdb.

[24] C.A. Murthy, N. Chowdhury. In search of optimal clusters using genetic algorithms. Pattern Recognition Letters, 1996; 17: 825–832.