

Evolutionary Design of Multilayer and Radial Basis Function Neural Network Classifiers: an Empirical Comparison

Njah Mohamed and El Hamdi Ridha

Control and Energy Management laboratory (CEM Lab)
University of Sfax, Sfax Engineering School, BP 1173, 3038 Sfax, Tunisia.

Summary

Artificial neural networks have been recognized as a powerful tool for pattern classification problems and have attracted a lot of research effort in the field of machine learning. However, optimal design of such models is known to be a notoriously hard problem. In this paper we investigate the effectiveness of a new hybrid evolutionary approach to address the optimal design of neural network based classifiers. The particularity of this approach lies in the use of two major techniques; the first one is to carry out an elite-based-reproduction strategy using either the compact genetic algorithm or a learning automata-based algorithm and the second one is the evolution itself driven by the differential evolution algorithm. The proposed approach is applied to both multilayer and radial basis function neural network classifiers. Different testing and training scenarios are presented using two classification benchmark problems, obtained from the UCI repository. Those scenarios are designed to provide an empirical comparison of performances of the two-classifier models and the most suitable elite-based-reproduction strategy used.

Key words:

Neural Network classifier, Learning, Hybrid Evolutionary Algorithm, Differential Evolution, learning automata, Compact Genetic Algorithm.

1. Introduction

Neural network based classifiers have attracted a lot of research effort in the field of machine learning and are widely used in real world applications, including handwritten characters recognition, detection of faces in images, medical diagnosis and several other tasks [1] [2] [3]. This however does not imply that a Neural Network (NN) can easily learn the underlying functional mapping between the input data and the desired output. In fact, the main drawbacks of NN are problems associated with the optimal design of the NN model.

Population-based stochastic search approaches, such as Evolutionary Algorithms (EA) [4] have attracted a lot of research effort during the last 20 years, and they are still one of the hottest research areas in the computer science research community. EA have been proposed to address the problem of the optimal design of the whole NN, since such methods are particularly useful for dealing with complex problems having large search spaces with many

local optima. A comprehensive review of these approaches can be found in [5].

In this paper, a new hybrid evolutionary approach is proposed to address the optimal design of neural network classifiers. The particularity of this approach lies in the use of two aspects: a) the development of an elitist reproductive strategy to promote the group of individuals having the best structure, to dominate the population. This is done by changing the proportions of individuals in each group according to the estimate of a probability distribution associated with each individual in the population and b) the evolutionary process itself. For the first aspect, two algorithms are investigated in this work. The first one is the compact genetic algorithm [6], whereas the second one is the learning automata-based algorithm [7]. For the second aspect, Differential Evolution (DE) strategy [8] is used to perform the evolutionary process. This paper presents, moreover, a first rigorous attempt to apply the same evolutionary proposed method for the automatic design of both multilayer and radial basis function neural network classifiers. Different testing and training scenarios are presented. Those scenarios are designed to obtain the most suitable classifier model for each used benchmark problem.

The remainder of this paper is organized as follows. Section 2 summarizes the advances in evolutionary generation of NN-Classifiers. The two elitist reproductive strategies, used in this paper, are covered in Section 3. Algorithm implementation of the proposed approach, based on the two aspects mentioned above, is given in Section 4. Simulation results and discussions of the obtained results are drawn in Section 5. Finally, the paper is concluded in Section 6 with some potential directions for future research.

2. Evolutionary Generation of NN-Classifiers

There have been extensive efforts in recent years to improve the automatic design of Artificial Neural Network models. Constructive and pruning algorithms are two most widely used non-evolutionary methods. A constructive algorithm [9] adds new nodes and connection information incrementally to a minimal NN architecture during

training, whereas a pruning algorithm [10] removes needless nodes and connections gradually starting from larger NN during training. Although this may result in good generalization models, both constructive and pruning algorithms are highly dependent on user defined training parameters.

An alternative is offered by evolutionary computing techniques [4], which have been successfully applied to finding the global optima of various multidimensional functions. Such methods are particularly useful for dealing with complex problems having large search spaces with many local optima [4]. Moreover, they are less dependent on user defined training parameters [4]. A comprehensive review of these approaches can be found in [5].

Several authors [11, 12, 13] discussed the use of Differential Evolution (DE) [8], as more promising evolutionary algorithm, for automatic design of NN models. Liu and Lampinen [13] apply a DE-based incremental training method in the search of the RBF network parameters (centers, weights and widths) which provide the best possible function approximation. The comparison with other incremental algorithms reported in the literature has shown that the DE-based RBF network growing approach combined with cycling scheme performed better in terms of the lower Mean Square Error (MSE) between the desired and actual outputs with smaller network in the tested cases. In [11], DE algorithm was implemented to train RBF networks. In each iteration, DE is used to determine centers of hidden units. Whereas, widths of hidden units were calculated by heuristic methods, and connection weights between hidden layer and output layer were obtained by SVD [14]. In [12], a new Enhanced Differential Evolution (EDE) algorithm has been developed to construct the optimal RBF network for fast restoration of distribution systems under various load levels. The proposed EDE provides more efficient fitting and forecasting capabilities than some other methods based on the same network structure.

3. Elitist Reproductive Strategies

The purpose of the elitist reproductive strategy is to promote group of individuals having the best structure, to dominate the population. To carry out such strategy, two algorithms are investigated, in this work: the compact genetic algorithm [6] and the learning automata-based algorithm [7].

3.1 Compact Genetic Algorithm (CGA)

The compact Genetic Algorithm [6] is an Estimation of Distribution Algorithm that represents a virtual population for a binary Genetic Algorithm (GA) by a vector of probabilities. The principle of CGA algorithm is to assign a probability to each sub-population (group of individuals

with the same structure). This probability translates the chance of the sub-population to be the best structure. The CGA algorithm uses a virtual population represented by the following probability vector

$$Pr = [Pr_{\theta_{min}} \dots Pr_{\theta_j} \dots Pr_{\theta_{max}}] \quad (1)$$

Where Pr_{θ_j} refers to the group of individuals with j units in the hidden layer. At the beginning of the algorithm, the different groups have the same chance to represent the optimal structure. They all have a probability of 0.5. At each generation, the probability vector is updated according to the following equations:

$$\begin{cases} Pr_{\theta_j} = Pr_{\theta_j} + (Weight_{\theta_j} \times \lambda) & \text{if } Avg_fit \leq Max_fit_{\theta_j} \\ Pr_{\theta_j} = Pr_{\theta_j} - (Weight_{\theta_j} \times \lambda) & \text{if } Avg_fit > Max_fit_{\theta_j} \end{cases} \quad (2)$$

Where $\theta_j \in \{\theta_{min} \dots \theta_{max}\}$ and $Weight_{\theta_j}$ is the update rate calculated as follows :

$$Weight_{\theta_j} = \frac{\sum_{k=1}^{N_{\theta_j}} Fit_k^{\theta_j}}{\sum_{i=1}^{T_{pop}} Fit_i} \quad (3)$$

The term $\sum_{k=1}^{N_{\theta_j}} Fit_k^{\theta_j}$ represents the sum of the fitness values of the chromosomes in the j^{th} group.

λ is a predefined threshold value. $Max_fit_{\theta_j}$ is the best fitness value in the j^{th} group and Avg_fit is the average fitness value in the whole population calculated as follows:

$$Avg_fit = \frac{1}{T_{pop}} \sum_{i=1}^{T_{pop}} Fit_i \quad (4)$$

If an element of the probability vector reaches 1, the CGA is stopped and the probability vector is set to 1 for all groups with the same structure. To complete the population, chromosomes are generated randomly.

3.2 Learning Automata-Based Algorithm

Each sub-population represents an action (candidate) for the learning automaton [7]. At each generation t , an action is considered after a selection procedure, which consists in duplicating each individual according to its average fitness value. This can easily be done by performing consecutive random draws where each individual, of the j^{th} group, has the following probability of selection:

$$p_s(\theta_j^t) = \frac{Avg_fit_{\theta_j}(t)}{\sum_{\theta_j^t = \theta_{\min}^t}^{\theta_{\max}^t} Avg_fit_{\theta_j}(t)} \quad (5)$$

$Avg_fit_{\theta_j}(t)$ is the average fitness value of the individuals of the j^{th} group at the generation t calculated as follows (Eq. 6):

$$Avg_fit_{\theta_j}(t) = \frac{\sum_{i=1}^{N_{\theta_j}(t)} Fit_{\theta_j}^i(t)}{N_{\theta_j}(t)} \quad (6)$$

$Fit_{\theta_j}^i(t)$ is the fitness value, at the generation t , of the individual of the j^{th} group.

The evaluation system generates $u(t)$ (Eq. 7) that reflects the quality of the behaviour of the action.

$$u(t) = \begin{cases} 0 & \text{if } Max_fit_{\theta_s}(t) \leq Max_fit_{\theta_s}(t-1) \\ 1 & \text{otherwise} \end{cases} \quad (7)$$

$Max_fit_{\theta_s}(t)$ and $Max_fit_{\theta_s}(t-1)$ are the best fitness values of the s^{th} selected group, respectively at t and $t-1$ generations.

According to this value $u(t)$, the probability adjustment algorithm will adjust the probability distribution of the different actions while keeping their sum equals 1 [15].

In every generation t , if $u(t)=1$, no change is introduced into the probability vector $Pr = [Pr_{\theta_{\min}}, \dots, Pr_{\theta_j}, \dots, Pr_{\theta_{\max}}]$. Else, it is updated according to the following equations (Eq. 8) [15]:

$$\begin{cases} Pr_{\theta_j}(t+1) = Pr_{\theta_j}(t) + \lambda \times (1 - Pr_{\theta_j}(t)) \\ Pr_{\theta_j}(t+1) = Pr_{\theta_j}(t) \times (1 - \lambda) \text{ for } j \neq s \end{cases} \quad (8)$$

Where $\theta_j \in \{\theta_{\min} \dots \theta_{\max}\}$ and λ is a predefined threshold value chosen between 0 and 1.

If the probability of an element of Pr approaches 1, the encoded structure of the corresponding group is considered as optimal. The algorithm is then stopped forcing all other groups to have the optimal structure found.

4. Algorithm Implementation

4.1 Representation Scheme

The encoding is the first step of an EA when it treats a particular problem. Since the NN architectures affect also

the lengths of chromosomes; a special representation is used to tackle certain particularly difficult problems for which a generic representation used in EAs might not be appropriate.

Both multilayer perceptron (MLP) and radial basis function neural (RBF) networks, which we consider in this work, share the characteristic of symmetry between hidden units. In other words, it is possible to interchange places of the units of the hidden layer while maintaining equivalent network.

As shown in Figures (1 and 3), we can divide the network into homogeneous blocks represented by units of the hidden layer.

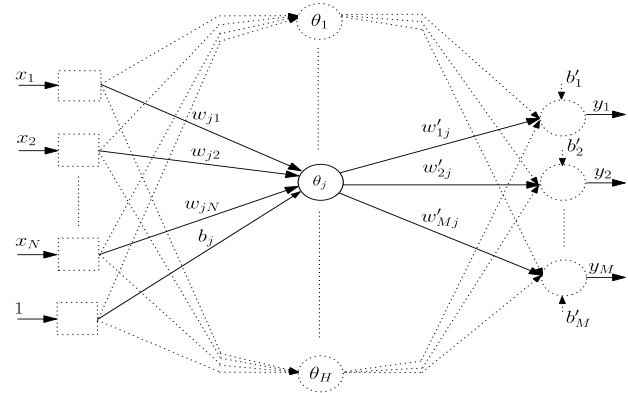


Fig. 1. Parameters associated to a hidden node θ_j (MLP)

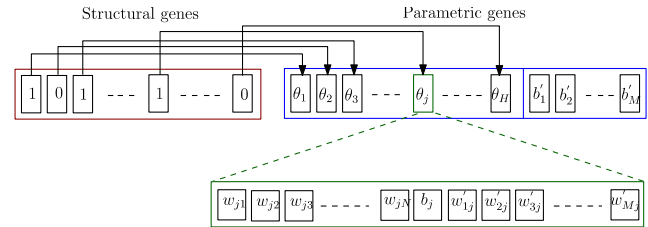


Fig. 2. Hierarchical representation of a chromosome (MLP)

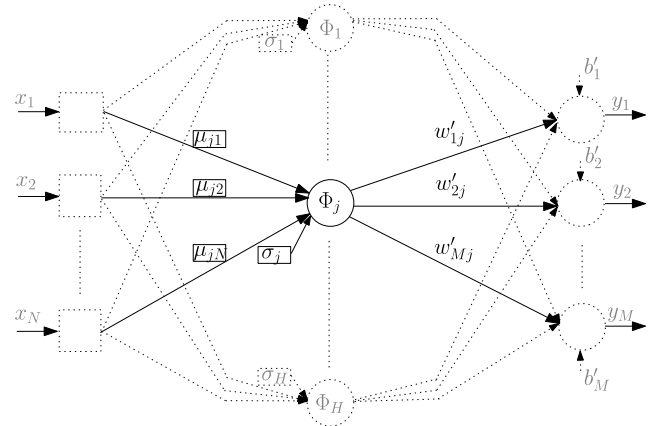


Fig. 3. Parameters associated to a hidden node θ_j (RBF)

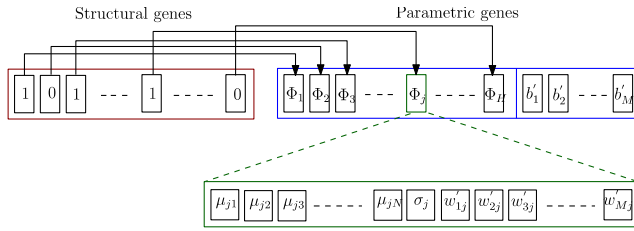


Fig. 4. Hierarchical representation of a chromosome (RBF)

Figures (2 and 4) show the genotypic representations of respectively the MLP and the RBF networks.

This hierarchical representation uses two types of coding:

- A binary mask (structural genes) representing the enabled units of the hidden layer (i.e. if an element of the mask is set to zero, the corresponding unit is not fed to the network).
- A real-valued string (parametric genes) formed by concatenating parameters of hidden layer units (i.e. connection weight vectors ($\underline{w}_j$) and bias (b_j) for MLP and the center vectors ($\underline{\mu}_j$) and spreads (σ_j) of the basis functions for RBF), and the synaptic weights ($\underline{w}_{kj}$) of the output layer nodes.

4.2 Replacement Scheme

In each generation, individuals in the population are sorted as follows: The individual having the best (the minimum) cost function is of rank 1 and the one having the highest cost function is ranked T_{POP} (T_{POP} is the size of the population). We, then, divide the population into two

sub-populations of identical sizes ($\frac{T_{POP}}{2}$). Individuals in the first half of the population, called sub-population of parents, will serve to the reproductive phase. Whereas individuals of the second half of the population, called sub-population of children, will be replaced by new individuals already created. In each generation, the number of individuals belonging to each group in the sub-population of parents that will be selected for the reproduction will depend on its probability distribution.

For this, each group j in the sub-population of parents is assigned a number Chg_{θ_j} of chromosomes that will be selected. When the probability distribution associated with each individual is based in the compact genetic algorithm, Chg_{θ_j} is calculated as follows (Eq. 9) [6]:

$$Chg_{\theta_j} = E\left(\frac{T_{POP} * Pr_{\theta_j}}{2 \sum_{Pr_{\theta_j}=\theta_{min}}^{\theta_{max}} Pr_{\theta_j}}\right) \quad (9)$$

Where E is the ceiling function.

If a learning automata-based algorithm is used to estimate the probability distribution associated with each individual,

Chg_{θ_j} is calculated as follows (Eq. 10) [16]:

$$\begin{cases} Chg_{\theta_j}(t) = E\left(\frac{T_{POP}}{2} * Pr_{\theta_j}(t)\right) & \text{if } u(t)=0 \\ Chg_{\theta_j}(t) = Chg_{\theta_j}(t-1) & \text{if } u(t)=1 \end{cases} \quad (10)$$

4.3 Evolutionary Process

The Differential Evolution (DE) strategy [8], which has gained popularity thanks to its simplicity and good observed performance, has been employed to perform the evolutionary process.

In DE, a child is generated applying the crossover operator to three parents (p_1, p_2 and p_3). The resultant child (c) is a perturbation of the main parent (p_1).

If we consider $X_i^{p_1}$, $X_i^{p_2}$ and $X_i^{p_3}$ the i^{th} parametric genes respectively of parents p_1, p_2 and p_3 , then the i^{th} parametric gene of the child c (i.e. X_i^c) is generated as follows:

$$X_i^c = \begin{cases} X_i^{p_1} + F(X_i^{p_2} - X_i^{p_3}) & \text{if } U[0,1] \leq CR \\ X_i^{p_1} & \text{else} \end{cases} \quad (11)$$

Where $F \in [0,1]$ is a predefined factor. CR is the rate of crossover and CR is a random fraction in the $[0, 1]$ interval.

Once the child parametric genes are generated, we proceed to apply the same principle of differential evolution for structural genes to determine the structure of the child to generate.

If we consider $Y_i^{p_1}$, $Y_i^{p_2}$ and $Y_i^{p_3}$ the i^{th} structural genes respectively of parents p_1, p_2 and p_3 , the i^{th} structural gene of the child c (i.e. Y_i^c) is generated as follows [17]:

$$Y_i^c = \begin{cases} 0 & \text{if } (Y_i^{p_1} + F(Y_i^{p_2} - Y_i^{p_3})) \leq 0.5 \\ 1 & \text{else} \end{cases} \quad (12)$$

5. Simulation Results and Discussion

The proposed approach was developed and implemented using Xcode version 5 under a Mac os x 10.11.5 workstation.

To examine the performance of the proposed approach for both MLP and RBF classifiers, we consider two benchmark classification problems, selected from the UCI repository¹.

The first used dataset is the Wisconsin Breast Cancer Diagnosis (WBCD). It represents medical measurements, which represents the result of the efforts made at the University of Wisconsin Hospital for accurately diagnosing breast masses based solely on a Fine Needle Aspiration (FNA) test.

The dataset consists of 699 samples taken from Fine Needle Aspirates (FNA) of human breast tissue. Each record in the database has nine integer-valued attributes, which represent cytological characteristics of breast FNA. To these attributes is assigned an integer value between one and ten, with one being the closest to benign (non-cancerous) and ten the most malignant (cancerous). The dataset contains two classes referring to benign and malignant samples. Table 1 gives a brief description of this dataset. We removed the sixteen instances with missing values from the dataset to construct a new dataset with 683 instances.

Table 1. Description of the WBCD dataset

Class	Data distribution	
	Nb of instances	Missing attributes
Benign	458	14
Malignant	241	2
Total	699	16

The second dataset, that we consider in this work, is the IRIS, which has been used extensively for evaluating the performance of pattern classification algorithms. This data set contains 150 samples of dimension four that are Sepal Width, Sepal Length, Petal Width, and Petal Length. These samples can be divided into three classes (Iris Setosa, Iris Versicolour, and Iris Virginica) representing different IRIS subspecies. Setosa class is far from the other two, which have overlap of their features. Figure 5 shows two scatter plots of the IRIS dataset for sepal length vs. sepal width (left) and for petal length vs. petal width (right).

In this paper, we denote :

- M1 : the proposed method of multilayer neural network classifiers design using the compact genetic algorithm as an elitist reproductive strategy.
- M2 : the proposed method of multilayer neural network classifiers design using a learning automata-based algorithm as an elitist reproductive strategy.
- M3 : the proposed method of radial basis function neural network classifiers design using the compact genetic algorithm as an elitist reproductive strategy.
- M4 : the proposed method of radial basis function neural network classifiers design using a learning automata-based algorithm as an elitist reproductive strategy.

Tables 2 and 3 provide, for different values of λ , the optimal number of hidden units found respectively by M1 and M2 for the Iris dataset on 30 runs.

$iter_{\alpha}(\max)$ is the maximum number (on 30 runs) of required iterations to converge to the appropriate network structure.

According to tables 2 and 3, if the value of λ is relatively high ($\lambda > 0.1$) the number of hidden units obtained by the two methods is arbitrary. In fact, the decision about the winner is, in such case, premature and the generated NN depend on the random initialization of network parameters. On the contrary, if the value of λ is too low, the premature convergence of the method is avoided and the number of hidden units obtained focuses on a definite value, but the convergence is very slow.

A choice of $\lambda = 0.01$ usually gives a good compromise (between the values of λ relatively high and those relatively low), at least for the applications we tested.

The other point to be drawn from the results of Tables 2 and 3 is that the convergence to the appropriate structure of the method M1 is faster compared to the method M2.

Table 4 shows the results obtained by the different methods (M1, M2, M3 and M4) for the Iris and Breast Cancer datasets.

$iter_{\alpha}$ is the number of iterations required by the method to converge to the appropriate network structure, while $iter_c$ denotes the total number of iterations. $\%acc$ is the accuracy of the designed classifier, evaluated in terms of percentage of correct classifications.

¹ <http://archive.ics.uci.edu/ml/datasets.html>

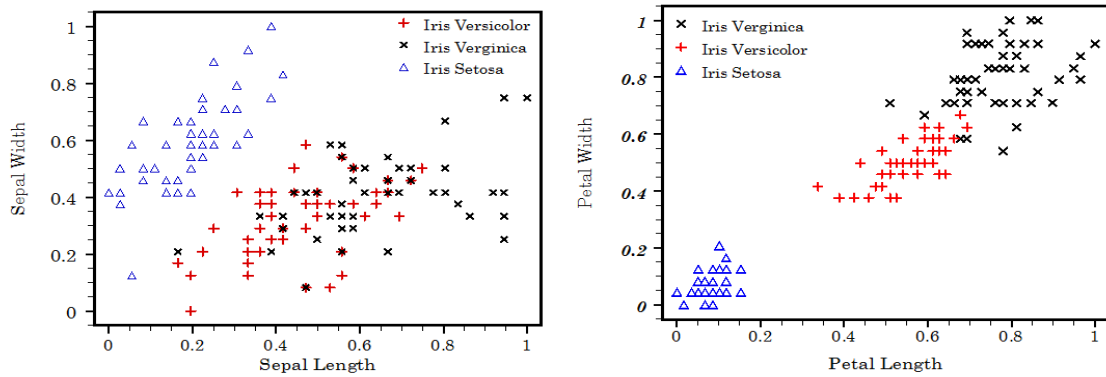


Fig. 5. Scatter plots of the Iris data set

Table 2. Convergence of the method M1 for different values of λ (Iris dataset)

λ	Number of hidden units obtained							$iter_{\alpha}(\max)$
	3	4	5	6	7	8	9	
0.5	-	-	3 times	6 times	5 times	5 times	11 times	44
0.1	1 time	4 times	4 times	7 times	5 times	4 times	5 times	86
0.05	2 times	-	-	13 times	-	-	5 times	117
0.01	-	-	2 times	26 times	-	2 times	-	165
0.001	-	-	-	27 times	-	-	3 times	309

Table 3. Convergence of the method M2 for different values of λ (Iris dataset)

λ	Number of hidden units obtained							$iter_{\alpha}(\max)$
	3	4	5	6	7	8	9	
0.5	3 times	2 times	3 times	2 times	5 times	6 times	9 times	37
0.1	1 time	4 times	4 times	7 times	5 times	4 times	5 times	102
0.05	2 times	-	-	13 times	-	-	5 times	267
0.01	1 time	-	2 times	23 times	-	2 times	2 times	470
0.001	1 time	-	-	28 times	-	-	1 time	962

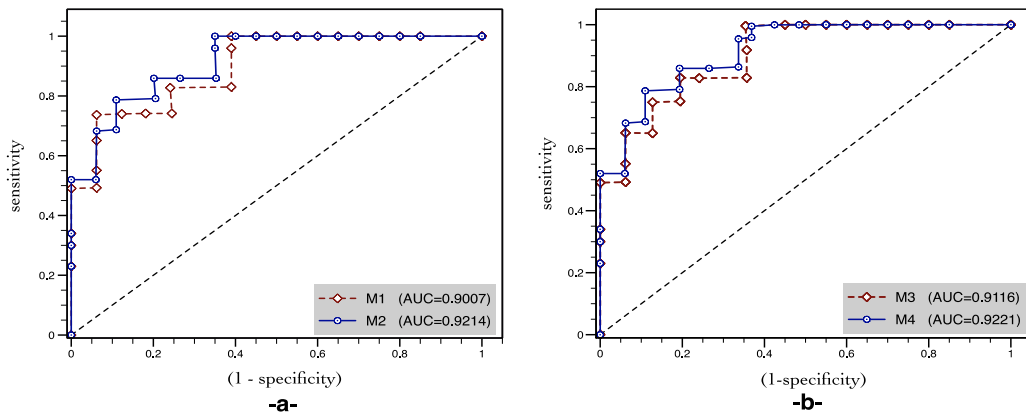


Fig. 6. ROC curves generated by both methods for multilayer (a) and radial basis function (b) neural network classifiers

As shown in table 4, the number of hidden units θ_H

(respectively ϕ_H) found by M1 and M2 (respectively M3 and M4) for multilayer (respectively for radial basis

function) neural network classifiers are identical except for breast cancer dataset using radial basis function model (i.e. M3 and M4). However, in terms of accuracy, both designed models are identical. The results reported in this table show also that radial basis function neural network model achieves slightly better performance in terms of accuracy (98,65% as opposed to 97,98% for multilayer classifier) for the breast cancer dataset.

Table 4. Classification results of the used classification problems

Iris				
Method	$\theta_H(\phi_H)$	%acc	iter _{cs}	iter _c
M1	6	97.33	152	367
M2	6	97.33	440	698
M3	21	97.33	279	397
M4	23	97.33	787	889
Breast Cancer Diagnosis				
Method	$\theta_H(\phi_H)$	%acc	iter _{cs}	iter _c
M1	6	97.98	188	461
M2	6	97.98	554	693
M3	4	98.65	419	517
M4	4	98.65	561	709

In terms of computational time, methods using the compact genetic algorithm as an elitist reproductive strategy (M1 and M3) are faster to converge to the appropriate structure than methods using the learning automata-based algorithm (M2 and M4). Once the appropriate structure is obtained, these methods (M2 and M4) do not require a high number of iterations (compared to M1 and M3) to converge to the final solution.

Another comparison between the different methods (M1, M2, M3 and M4) was carried out using a Receiver operating characteristics (ROC) graph, which is a very useful tool for visualizing and evaluating classifiers [18]. An ROC graph is a two-dimensional depiction of classifier performance in which the true positive rate (*sensitivity*) is plotted on the Y-axis and the false positive rate (1 - *specificity*) is plotted on the X-axis.

An important point about ROC graphs is that they are insensitive to changes in class distribution. If the proportion of positive to negative instances changes in a test set, the ROC curves will not change [18].

Figure 6 shows the ROC graph of the developed classifier using breast cancer dataset. Resulting ROC curves are obtained by varying the output threshold value of the activation function, so that only binary classification dataset is used.

In general, the area under the ROC curve, abbreviated AUC, is a powerful index for assessing the classification performance of the classifier. The closer the value AUC is to 1, the better is the performance of the model. Therefore, models obtained by methods using the learning automata-based algorithm, as an elitist reproductive

strategy, (M2 and M4) are slightly better than models obtained by methods using the compact genetic algorithm, as an elitist reproductive strategy (M1 and M3). Moreover, if the same elitist reproductive strategy is used, radial basis function model achieves slightly better performance than multilayer based classifier.

6. Conclusion

In this paper, a new hybrid evolutionary approach has been presented and applied to address the optimal design of neural network classifiers.

The proposed approach is divided into two major steps; the first one is to carry out the elite-based-reproduction strategy using either the compact genetic algorithm or a learning automata-based algorithm and the second one is the evolution itself driven by the differential evolution algorithm. The proposed approach was tested for both multilayer and radial basis function neural network classifiers. Simulation results on two benchmark classification problems, indicate that models given by methods using the learning automata-based algorithm, as an elitist reproductive strategy perform slightly better than models given by methods using the compact genetic algorithm. Moreover, if the same elitist reproductive strategy is used, radial basis function model achieves slightly better performance than multilayer based classifier.

An interesting issue that rises from this work is how to extend the proposed approach using multi-objective optimization to emphasize the trade-off between complexity and accuracy of the NN based classifier.

References

- [1] Abaynarh M., Elfadili H., Zenkour K. and Zenkour L. "Neural Network Classifiers for Off-line Optical Handwritten Amazighe Character Recognition," Int. J. of Computer Science and Network Security, vol.12, pp. 28-36, 2012.
- [2] Nuryanti M. S., Harsa A. M. S. and Othman N. H. "Neural Networks to Evaluate Morphological Features for Breast Cells Classification," Int. J. of Computer Science and Network Security, vol. 8, pp. 51-58, 2008.
- [3] Chaitali V., Nikita B. and Darshana M., A Survey on Various Classification Techniques for Clinical Decision Support System. International Journal of Computer Applications, Vol. 116 - No. 23, 2015
- [4] Castellani M., Evolutionary generation of neural network classifiers : An empirical comparison. Neuro computing, 99, pp. 214-229, 2013.
- [5] Azzini A. and Tettamanzi A. G. B., Evolutionary ANNs: a state of the art survey. Intelligenza Artificiale Vol. 5, pp. 19-35, 2011.
- [6] Lin C.-J. and Xu Y.-J. A hybrid evolutionary learning algorithm for TSK-type fuzzy model design Mathematical and computer Modeling, Vol. 43, pp. 563-581, 2006.

- [7] Njah M., Alimi M.A, Chtourou M. and Tourki R. Automatic Synthesis of Neural Networks Using Learning Automata IEEE International Conference on Systems, Man and Cybernetics, Vol. 5, 2002.
- [8] Storn R., Differential evolution research - Trends and open questions. In : Advances in differential evolution. Studies in computational intelligence. Vol. 143, pp. 1-31, 2008.
- [9] Sharma S. K. and Chandra P., Constructive Neural Networks: A review. International Journal of Engineering Science and Technology, Vol. 2(12), pp. 7847-7855, 2010.
- [10] Engelbrecht, A.P., A new pruning heuristic based on variance analysis of sensitivity Information. IEEE transactions of Neural Networks, Vol. 11(6), pp. 1386-1399, 2001.
- [11] Bing Yu and Xingshi He, Training Radial Basis Function Networks with Differential Evolution Granular Computing, 2006 IEEE International Conference. pp. 369-372, 2006.
- [12] Huang Y.C., Chen S.J., Huang C.M., Evolving Radial Basic Function Neural Network for Fast Restoration of Distribution Systems Industrial Electronics and Applications (ICIEA), pp. 401-406, 2011.
- [13] Liu J. and Lampinen J., A Differential Evolution Based Incremental Training Method for RBF Networks Proceeding GECCO'05 Proceedings of the 7th annual conference on Genetic and evolutionary computation. pp. 881-888. ACM, 2005.
- [14] Muller N., Magaia L. and Herbst B.M., Singular Value Decomposition, Eigenfaces, and 3D Reconstructions SIAM Review Vol. 46(3), pp.518-545, 2004.
- [15] Thatachar and Ramakrishman A hierarchical system of learning automata, IEEE trans. On Sys. Man and Cybernetics, pp. 236-241, 1981.
- [16] El Hamdi, R., Njah, M. and Chtourou, M. A hybrid evolutionary design of neuro-fuzzy systems. 7th International Multi- Conference on Systems Signals and Devices (SSD), pp. 1 - 6, 2010.
- [17] Cruz-Ramirez M., Fernandez J. C., Sanchez-Monedero J. and Hervás-Martínez C., Memetic pareto differential evolutionary artificial neural networks to determine growth multi-classes in predictive microbiology. Evolutionary Intelligence, Vol. 3, pp. 187-199, 2010.
- [18] Fawcett T., An Introduction to ROC Analysis. Pattern Recognition Letters, vol. 27, pp. 861-874, 2006.



Njah Mohamed received the Engineering diploma in Electrical Engineering from the National Engineering School (ENIS), University of Sfax in 1990, the DEA degree in Automatic Control from the ESSTT, University of Tunis in 1995 and the PhD degree in Electrical Engineering from ENIT University Elmanar Tunis in 2003. He was at SITEX textiles Company as an industrial informatics engineer from

1990 to 2001. He was a university assistant at FSEG Mahdia from 2001 to 2004 and assistant professor at ENIG Gabès from 2004 to 2009. Since 2009, he is assistant professor at ENIS. His research interests include neural networks and evolutionary algorithms.



El Hamdi Ridha received his engineering diploma in Electrical Engineering- Automatic Control from the National Engineering School of Gabès, University of Gabès (Tunisia) in 2004, his Master of Science (M.Sc.) in Automatic Control and Intelligent Techniques from University of Gabès in 2006 and his Ph.D. in Electrical Engineering from University of Sfax. Currently, he is an assistant

professor at National Engineering School of Gabès. His research interests include artificial intelligence especially hybrid computational methods.