THE PUBLISHING HOUSE
OF THE ROMANIAN ACADEMY

PROCEEDINGS OF THE ROMANIAN ACADEMY, Series A,
Volume 6, Number 2/2005, pp. 000-000

ADAPTATION SCHEMES FOR A NEURO-FUZZY PREDICTOR

Lavinia Eugenia FERARIU[*], Horia Nicolai TEODORESCU*, Lucian Iulian FIRA[**]

[*]Technical University of Iasi, Faculty of Automatic Control and Computer Engineering
*Technical University of Iasi, Faculty of Electronics and Communications, Institute for Theoretical Computer Science, Iasi
[**]Institute for Theoretical Computer Science, Iasi
Corresponding author: Horia Nicolai TEODORESCU, Bd. Carol I, no 11, Iasi, Romania, e-mail: hteodor@etc.tuiasi.ro

The paper presents, in a comparative study, six hybrid configurations that can assist the adaptation of the neuro-fuzzy system parameters. The suggested configurations involve various combinations of the gradient descent algorithm, the k-means clustering procedure and the genetic algorithms. The neuro-fuzzy system architecture is adopted from literature. The applicability of the methodology is investigated with respect to the time series prediction.

Keywords: Neuro-fuzzy systems, time series prediction, gradient algorithm, genetic algorithms

1. INTRODUCTION

A multitude of research studies indicates that the neuro-fuzzy systems represent a powerful tool for time series prediction [2, 4, 7, 13, 14, 15]. The neuro-fuzzy approach is a combination between two soft computing techniques, which are both suitable for modeling expert behavior [8]. Fuzzy systems and neural networks do not require any mathematical model of the problem to solve and are fault-tolerant to small changes of their inputs or parameters.

The neuro-fuzzy systems can take advantage both from the learning abilities of the neural networks [6] and the reasoning characteristics of the fuzzy systems. The combination should be able to learn linguistic rules and membership functions or to optimize the existing ones. Usually, the membership functions of the involved fuzzy systems are described by parameters, which are optimized during a training stage. On the other hand, because the neuro-fuzzy systems are based on linguistic rules, the learning result can be interpreted as a fuzzy system, thus avoiding the black box behavior of the neural networks. Moreover, at initialization, one can easily integrate prior knowledge, for improving the efficiency of the learning process. Various neuro-fuzzy architectures and learning algorithms were investigated in the related literature [8]. The gradient algorithms have the drawback of frequently converging to local minima. On the other hand, pure genetic algorithms are slow.

The paper investigates the potential of schemes using various combinations of the gradient descent algorithm, the k-means clustering procedure and the genetic algorithms for neuro-fuzzy predictors.

The evolutionary algorithms represent efficient optimizations tools, able to cope with multimodality, noise, discontinuity and time-variance. At each iteration, they work on a population of possible solutions, named individuals or chromosomes. The most adapted ones are encouraged to produce new solutions (offspring), using mechanisms similar to biological recombination. The offspring compete with their parents and the most adapted ones survive to the next generation. After a large number of generations, the population contains many duplicates of the same individual, selected as result of the algorithm.

The performances of the genetic search are compared with the performances of the k-means clustering algorithm and the techniques based on the uniform distribution, in the case of center selection, respectively with the performances of the gradient descent procedure (with momentum) in the case of the output singleton adaptation. We analyze six configurations, recommended as support for providing an accurate one step ahead prediction.

The paper is organized as follows. Section 2 presents the architecture of the neuro-fuzzy system,

*Memeber of the Romanian Academy

adopted from [9, 13]. The adaptation schemes used for the selection of centers and singletons are described in Section 3. Their applicability to time series prediction is discussed in Section 4, based on several experimental results achieved by simulation. Section 5 contains a discussion of the complexity issues and the Section 6 is devoted to final remarks.

2. NEURO-FUZZY SYSTEM ARCHITECTURE

The architecture of neuro-fuzzy prediction system was introduced in [9, 13]. It was successively tested for the prediction of the genomic time series in [5, 10, 11, 12]. This hybrid topology permits to interpret the changes produced by the learning process from the point of view of both neural networks and fuzzy systems.

The neuro-fuzzy predictor consists in a network of M cells represented by single input single output 0-type Sugeno fuzzy systems (Fig. 1).

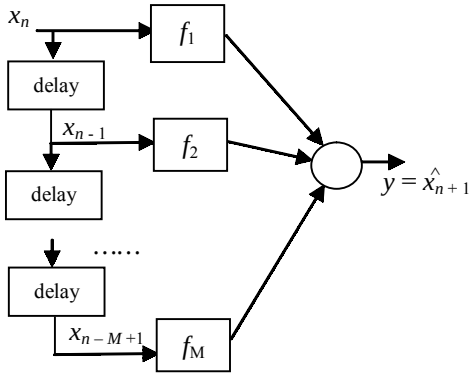

Fig. 1. The architecture of the neuro-fuzzy system. Here x_n,…, x_{n-M+1} represent the actual and the delayed samples of the input signal, $\hat{x}_{n+1}$ indicates the predicted value, f_1,..., f_M specify the Sugeno fuzzy systems.

The predictor has the topology of a feedforward finite response filter with the fuzzy cells acting as nonlinear multipliers of the input delayed samples [11]. The Sugeno fuzzy systems are based on $N = 7$ Gaussian membership functions. The equation describing the input-output transfer performed by the predictor [10]:

$$y = \sum_{k=0}^{M-1} f_{i+1}(x_{n-k}) = \sum_{k=0}^{M} \frac{\sum_{l=1}^{N} \beta_{kl} \cdot e^{-\frac{(x_{n-k}-a_{kl})^2}{\sigma_{kl}^2}}}{\sum_{l=1}^{N} e^{-\frac{(x_{n-k}-a_{kl})^2}{\sigma_{kl}^2}}}, \quad (1)$$

where M denotes the number of Sugeno fuzzy systems (here, $M = 5$), N indicates the number of membership functions for each Sugeno fuzzy system ($N = 7$), β_{kl} represent the values of the output singletons, a_{kl} and σ_{kl}^2 denotes the centers and the spreads of the Gaussian membership functions, respectively.

3. TECHNIQUES FOR NEURO-FUZZY PARAMETERS ADAPTATION

Data-driven methodologies are considered for the selection of centers and output singletons, as indicated below. The neuro-fuzzy parameters are determined according to supervised learning algorithms. For each adaptation scheme, the accuracy of the prediction is evaluated subject to the Mean Squared prediction Error (MSE), computed for the whole normalized training data set:

$$E = \frac{1}{T} \sum_{n=1}^{T} (x_{n+1}^d - \hat{x}_{n+1})^2 . \quad (2)$$

Here, T indicates the interval time used for the analysis, $\hat{x}_{n+1}$ and x_{n+1}^d denote, respectively, the predicted and target value at sample $n+1$.

The schemes suggested for the neuro-fuzzy predictor adaptation are briefly described in Table 1. All involved learning algorithms used for the adaptation of centers/singletons are indicated in sequel. The hybrid configurations are supplementary discussed in Section 4, with respect to time series prediction problem.

Table 1. The description of the adaptation schemes

Scheme	Selection of centers	Selection of singletons
*S*1	Uniformly distributed in [-1,1] interval	Gradient with momentum
*S*2	Uniformly distributed in [-1,1] interval	Genetic algorithm
*S*3	k-means clustering algorithm	Gradient with momentum
*S*4	k-means clustering algorithm	Genetic algorithm
*S*5	Genetic algorithm	Gradient with momentum
*S*6	Genetic algorithm	Genetic algorithm

The selection of centers is done by means of uniform selection, k–means clustering, or genetic algorithms.

The uniform selection considers a uniform distribution of the centers over the discourse universe. For Gaussian membership functions, the universe of discourse could be $(-\infty,+\infty)$. If a normalization of the data is used, the centers have to be distributed inside the normalization interval (e. g. [-1, 1]). Consequently, in the case of $N=7$ membership functions, the values of the centers result as follows:

$$a_{kl} \in \{-1,-0.66,-0.33,0,0.33,0.66,1\},\ l=1,..,7,\ k=1,..5. \tag{3}$$

The *k*-means clustering algorithm delimitates, over the considered training data set, *k* centers corresponding to *k* different groups. The values of the centers represent the means of the resulted clusters. The method can produce a non-uniform partition of the input space, according to the real grouping of the learning data. One uses $k=N=7$.

The centers of the membership functions are also determined using a genetic algorithm. The optimization problem demands the minimization of the mean squared error (2). Each chromosome directly encodes, as real values, the parameters of the membership Gaussian functions, namely the centers and the spreads. The most adapted individuals are encouraged to produce new solutions and to survive. At each generation, the offspring are generated using intermediary recombination and uniform mutation. The genetic search has to be performed for a large number of generations, in order to provide a good exploration of the search space. As the genetic algorithm is not formulated as a clustering algorithm, it is expected to achieve a less accurate prediction. Moreover, the genetic search is implemented to accomplish a more difficult task, namely to select both centers and spreads for the membership functions.

For the adaptation of the singletons, gradient-based algorithm with momentum and genetic techniques are considered.

The use of the gradient-based algorithm is allowed, due the fact that the membership functions are Gaussian type. The prediction error (2) is minimized, based on the adaptation equations (4):

$$\beta[n+1]=\beta[n]+\Delta\beta[n]+\alpha\Delta\beta[n],\ \text{with}\ \Delta\beta=-\eta\frac{\partial E}{\partial\beta}, \tag{4}$$

Here, $\Delta\beta$ denotes the variation produced for the adaptation of parameters β at the successive iterations, η indicates the learning rate ($\eta\in(0,1)$) and $\frac{\partial E}{\partial\beta}$ represents the gradient.

The implemented procedure is based on the momentum technique. According to this technique, the actual adaptation step takes into account the changes produced at the previous iterations. In that situation, the algorithm is able to perform a filtering over the error surface and to guarantee higher stability [1, 3].

The genetic algorithm determines the singletons, subject to the minimization of the objective function (2). The singletons are directly encoded using real numbers. A chromosome encodes a whole set of singletons. At each generation, the genetic algorithm works on a population of *NIND* chromosomes. The

initial population is randomly distributed over the permitted search space. The individual evaluation considers ranking-based fitness assignment. The recombination pool is filled in considering the stochastic universal sampling method. The intermediary recombination and the uniform mutation maintain the diversity of the genetic material and guarantee a robust exploration of the search space. The genetic search is implemented according to an island model. As consequence, the population is separated into several subpopulations that evolve quasi-independently. Once at NOMIGR generations, an exchange of information is allowed between the subpopulations. A schematic description of the genetic algorithm used is indicated below [3]:

Create the initial population (*NIND* individuals).
Evaluate the chromosomes and compute the fitness values.
Loop for *MAXGEN* generations:
  For each subpopulation:
    Select the parents for the reproduction pool.
    Apply the crossover and the mutation operators.
    Evaluate the offspring and compute the fitness values.
    Insert the offspring into the population.
    Once at *NOMIGR* generations, exchange individuals with the other subpopulations.
    Compute the fitness values.
Determine the best individual.
End of the algorithm.

4. EXPERIMENTAL RESULTS

A chaotic time series is used as benchmark for prediction. The series is generated using the 3-order discrete dynamic system [13], described in equation (5):

$$x_{n+1} = \frac{5x_n}{1+x_n^2} - 0.5x_n - 0.5x_{n-1} + 0.5x_{n-2}, \quad (5)$$

subject to the initial conditions $x_0 = 0.773$, $x_1 = 0.234$, $x_2 = 0.973$. Fig. 1 illustrates the dynamic behavior of the resulted time series, over the first 500 iterations, after the normalization to the [-1, 1] interval.

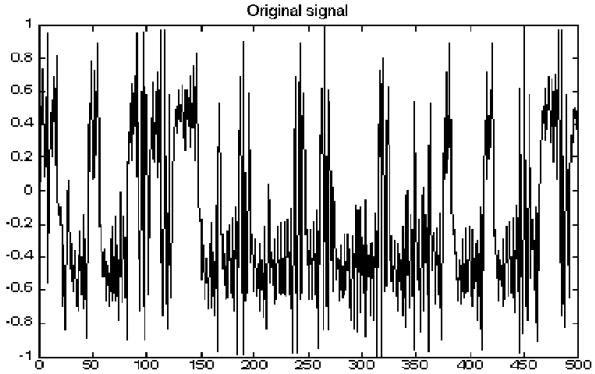

Fig. 2. The time series obtained for 500 iterations, after the normalization to the [-1, 1] interval.

Several experiments were carried out in order to verify the adaptation schemes *S*1- *S*6 presented in Table 1. A goal of this research work is to compare the genetic adaptation schemes with other adaptation procedures, as a preliminary study for the genetic design of dynamic neuro-fuzzy architectures.

The experimental results are illustrated in Tables 2-7, both for the training data set (iterations 1…180 of the chaotic time series) and for the testing data set (iterations 181…220 of the chaotic time series). The solutions indicated in Tables 2-7 are those providing the best accuracy of prediction, achieved for each adaptation scheme.

The configuration *S*1 (Table 2) considers the uniform selection of the centers inside the [-1, 1] interval and the gradient descent algorithm for the singletons adaptation. This configuration was successfully tested

before in [5, 10, 11, 12]. The learning rate is maintained constant during the training process ($\eta = 0.1$). The momentum rate, denoted with α, is set to 0.7. The adaptation algorithm stops when the absolute value of the difference that is considered between the mean squared errors resulted at two successive iterations is less than $\varepsilon = 0.00001$. At the end of the learning procedure, an accurate neuro-fuzzy system is achieved (*MSE* = 0.0084 for the training data set), characterized by satisfactory generalization capabilities (*MSE* = 0.0171 for the testing data set).

The configuration *S*2 (Table 3) uses a genetic algorithm for the singletons adaptation (instead of the gradient descent algorithm applied by *S*1). The genetic algorithm considers: *NIND* = 500 chromosomes into the population, with *LIND* = 35 genes (5 Sugeno fuzzy systems × 7 output singletons for each fuzzy system); the individuals are separated into *SUBPOP* = 10 subpopulations and the migration is applied once at *NOMIGR* = 5 generations. A half of the offspring survives to the next generation (*GGAP* = 0.5). The evolutionary loop is considered for *MAXGEN* = 100 generations.

The configuration *S*3 (Table 4) selects the centers using the k-means clustering algorithm (instead of the uniform distribution tested for *S*1). The *MSE* values obtained after the adaptation are 0.0030 for the train period and 0.0038 for the test period.

Table 2. Configuration *S*1-Parameter values for the adaptation algorithm and performances

Adaptation model	centers: uniformly distributed in [-1,1] interval; singletons: gradient with momentum.
The parameter values for the adaptation algorithm	singletons: $\varepsilon = 0.00001$, $\eta = 0.1$, $\alpha = 0.7$.
Results	train: *MSE*=0.0084, test, test: *MSE*=0.0171.

Table 3. Configuration *S*2-Parameter values for the adaptation algorithm and performances

Adaptation model	centers: uniformly distributed in [-1,1] interval; singletons: genetic algorithm.
The parameter values for the adaptation algorithm	singletons: *NIND*=500; *SUBPOP*=10; *NOMIGR*=5; *LIND*=35; *MAXGEN*=100; *GGAP*=0.5.
Results	train: *MSE*=0.0085, test: *MSE*=0.0103.

Table 4. Configuration *S*3-Parameter values for the adaptation algorithm and performances

Adaptation model	centers: k-means clustering algorithm; singletons: gradient with momentum.
The parameter values for the adaptation algorithm	centers: $k = 7$; singletons: $\varepsilon = 0.00001$, $\eta = 0.1$, $\alpha = 0.7$.
Results	train: *MSE*=0.0030, test: *MSE*=0.0038.

For the configuration *S*4 (Table 5), the selection of the centers is performed using the k-means clustering algorithm. The singletons are adapted using a genetic search, with the genetic parameter values previously mentioned at *S*2. The *MSE* values achieved after the adaptation stage are 0.0040 for the train period and 0.0061 for the test period.

In the case of configuration *S*5 (Table 6), the selection of the centers is done considering a genetic algorithm. The genetic algorithm parameters (denoted similarly as indicated for *S*2) are set as follows: *NIND*=100; *SUBPOP*=10; *NOMIGR*=5; *LIND*=14; *MAXGEN*=75; *GGAP*=0.5. The singleton adaptation uses the gradient algorithm described at the configuration *S*1. The *MSE* values obtained after the adaptation stage are 0.0073 for the train period and 0.0407 for the test period.

For the configuration *S*6 (Table 7), the selections of the centers and singletons are accomplished using genetic algorithms. In this case, the *MSE* values obtained after the adaptation result 0.0079 for the train period and 0.0087 for the test period.

Table 5. Configuration *S*4-Parameter values for the adaptation algorithm and performances

Adaptation model	centers: k-means clustering algorithm; singletons: genetic algorithm.
The parameter values for the adaptation algorithm	centers: $k = 7$, singletons: *NIND*=500; *SUBPOP*=10; *NOMIGR*=5; *LIND*=35; *MAXGEN*=100; *GGAP*=0.5; *PC*=0.7; *PM*=0.1;
Results	train: *MSE*=0.0040, test*: MSE*=0.0061.

Table 6. Configuration *S*5-Parameter values for the adaptation algorithm and performances

Adaptation model	centers: genetic algorithm; Singletons: gradient with momentum.
The parameter values for the adaptation algorithm	centers*: NIND*=100; *SUBPOP*=10; *NOMIGR*=5; *LIND*=14; *MAXGEN*=75; *GGAP*=0.5; singletons: $\varepsilon = 0.00001$, $\eta = 0.1$.
Results	train: *MSE*=0.0073, test: *MSE*=0.0407.

Table 7. Configuration *S*6-Parameter values for the adaptation algorithm and performances

Adaptation model	centers: genetic algorithm; singletons: genetic algorithm.
The parameter values for the adaptation algorithm	centers: *NIND*=100; *SUBPOP*=10; *NOMIGR*=5; *LIND*=14; *MAXGEN*=75; *GGAP*=0.5; *PC*=0.7; *PM*=0.1; singletons: *NIND*=500; *SUBPOP*=10; *NOMIGR*=5; *LIND*=35; *MAXGEN*=100; *GGAP*=0.5; *PC*=0.7; *PM*=0.1.
Results	train: *MSE*=0.0079, test: *MSE*=0.0087

The experimental results indicate that the adaptation scheme *S*3 offers the best results, both for the train and the test period. One confirms the ability of the clustering algorithms to select convenient centers and the computational efficiency of the gradient-based technique. The later one is able to exploit advantageously the local information in the case of convex objective functions. The genetic search, as a general approach, can be surpassed by these dedicated methods. Its main advantage is related to flexibility and to the reduced amount of a priori information requested for implementation.

Table 8. Comparison of the performances of the six configurations

Scheme	Parameters		Performances	
	Centers	Singletons	Train	Test
1	Uniform	Gradient	0.0084	0.0171
2	Uniform	GA	0.0085	0.0103
3	k-means	Gradient	0.0030	0.0038
4	k-means	GA	0.0040	0.0061
5	GA	Gradient	0.0073	0.0407
6	GA	GA	0.0079	0.0087

A comparison of the performances of the six configurations is presented in Table 8. While all six configurations offer quite accurate predictions (the train error being less than 0.01), some differences occur with respect to the generalization capabilities of the designed neuro-fuzzy systems. The worst results are obtained for configurations *S*5 and *S*1. In these cases, the error for the test period is larger than twice the error achieved for the train period.

For the *S*3 adaptation scheme, more details are given in Fig. 3-8. Fig. 3 illustrates the iterative improvement of MSE during the gradient-based algorithm, applied for the singletons adaptation. The stop condition is satisfied after about 300 iterations. Fig. 4 describes the performances achieved by the neuro-fuzzy system subject to the training data set (samples 1 to 180). An accurate prediction is performed, characterized by a reduced correlation of the error series (less than 0.1), as it is revealed in Fig. 5. The histogram of the errors, illustrated in Fig. 6, is almost Gaussian. The error histogram is an indication of the quality of the prediction process. While the normal distribution of the errors is no guarantee that the errors represent white noise, such a distribution is considered desirable [12]. In Fig. 7, the test period is presented (samples 181 to 220). The prediction quality is validated by a small correlation of the error series (less than 0.1), as indicated in Fig. 8 (bottom panel).

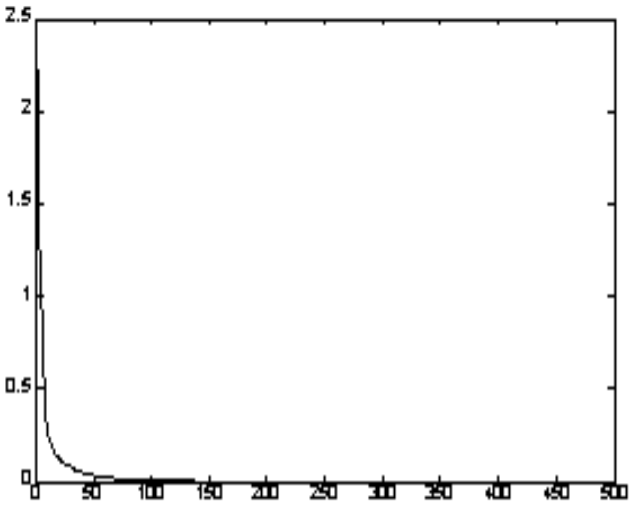

Fig. 3. The learning curve for the singleton adaptation.

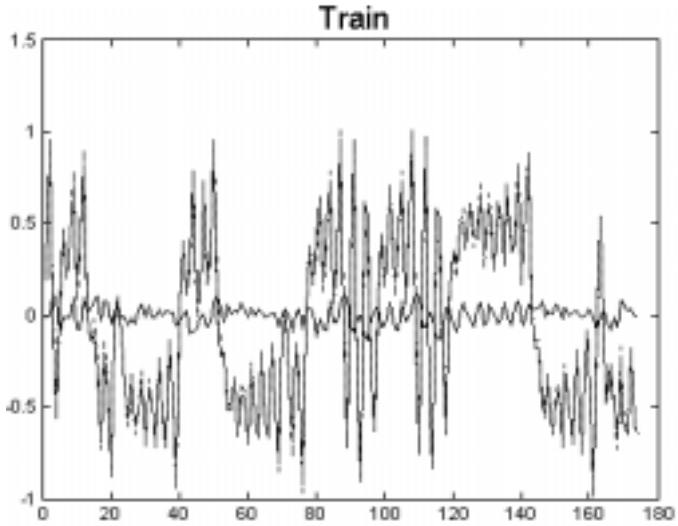

Fig. 4. The signals for the training period: the desired signal is represented with solid thin line, the predicted signal is represented with dotted thin line and the prediction error is drawn with thick solid line.

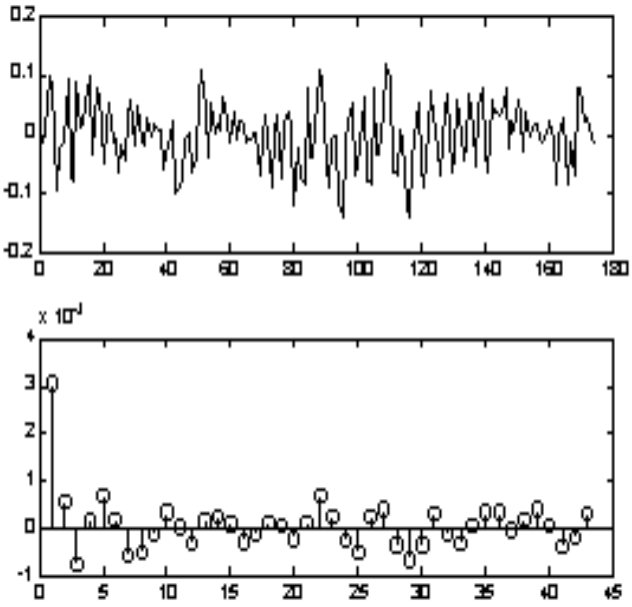

Fig. 5. The error (top) and the self-correlation of the error (bottom)-training data set.

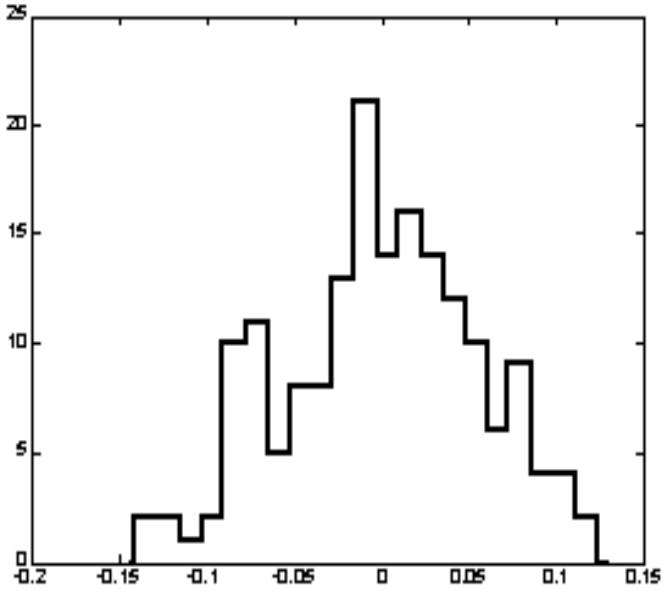

Fig. 6. The histogram of the prediction error-the training data set.

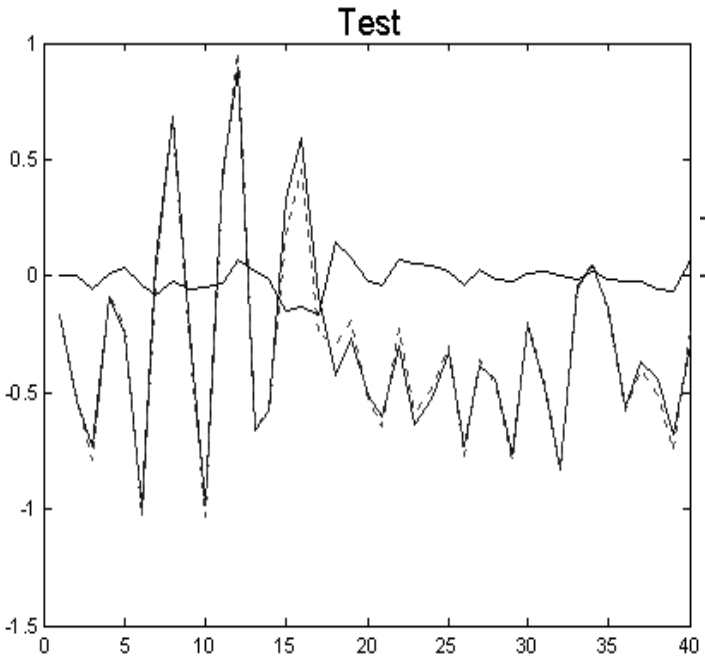

Fig. 7. The signals for the test period: the desired signal is represented with solid thin line, the predicted signal is represented with dotted thin line and the prediction error is drawn with thick solid line.

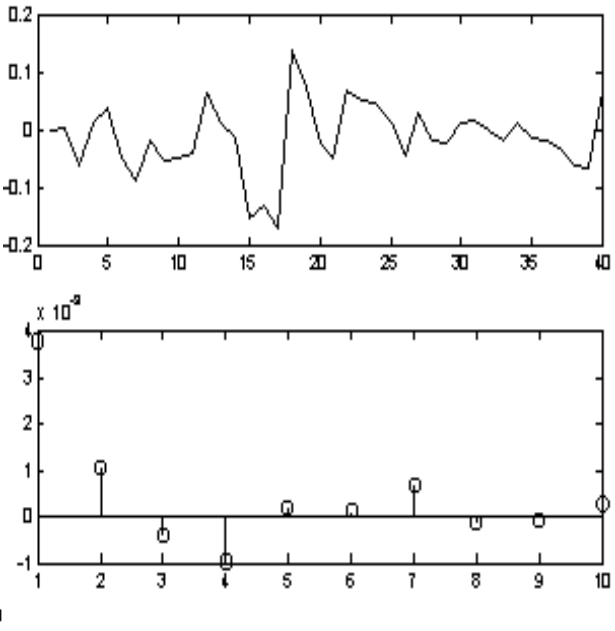

Fig. 8. The error (top) and the self-correlation of the error (bottom)-testing data set.

The comparative study suggests several new directions for future work. The one-stage genetic adaptation of centers and singletons implies a difficult optimization task. A better approach might be to divide the training process into two stages. At the first stage might use a genetic adaptation of the centers (formulated as a clustering algorithm) in combination with a gradient-based computation of singletons. The second stage might use the centers previously determined and can apply a genetic algorithm for a fine-tuning of the output singletons.

5. DISCUSSION OF COMPLEXITY ISSUES

The presented results should be considered preliminary results because we have not performed an analysis of the complexity of the training. Moreover, the stop condition in the algorithms has not been consistent. Indeed, in our analysis, the stop condition has not been the same for all the schemes and has not been related to the evolution of the error on a validation set. (We used as stop condition a maximum generation's number for all genetic algorithms and a minimum level of decreasing for all gradient algorithms). Further work is needed to determine the evolution of the error on the training set in relation to the error on the validation set. It is known that before some threshold of the training error, both the validation and the training errors decrease, while after the threshold, the training error decreases but the validation error increases [6]. The threshold is used as the best stop condition for the training. The time to reach the threshold is considered the learning time (see Fig. 9). However, determining the threshold is a tedious task and has not been performed in this research.

Prediction is a multiple-objective task. While the prediction error is the main quality indicator, it is not the only one. No predictor structure and related training algorithm can be considered a good choice until the costs of implementation and the costs of utilization are determined. The implementation costs relate to the time of developing the software, the resources (mainly memory) required, and the time of training on a given problem. When the system is hardware implemented, the circuitry complexity (for both training and utilization) and cost are also pertinent. The utilization cost relates chiefly to the running time to make a

prediction, after the training. Because we use a single neuro-fuzzy system, whose structure is not changed during the adaptation, the utilization cost is irrelevant here. In this discussion, we skip the developing costs and the resources costs. Moreover, we recall that there is no connection between the complexity of the system – typically assumed equal to the number of neurons or fuzzy systems in the system – and the learning complexity.

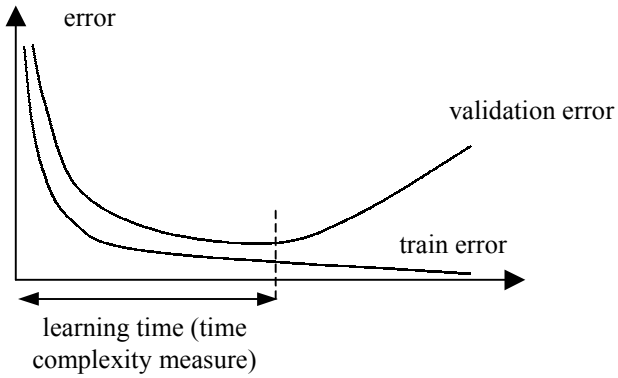

Fig. 9. Standard definitions of the stop condition and of the learning time [6]

The time of training, on a given problem, relates to the complexity of the problem and the complexity of the meta-problem of training. There are various points of view in the literature on measures of the complexity of training adaptive systems and specifically predictors. A popular view is that the complexity of the learning is proportional to the time of training scaled with a measure of the complexity of the problem to be learned by the trained system.

The experimental approach to learning complexity is the direct measurement of the learning time, for various "benchmark" problems and for several dimensions of the input data sets. This approach requires huge work, yet it is the only to demonstrate the benefits of a specific algorithm for a given task. This approach will be addressed in a future research. There are several theoretical results on the complexity of the training for neural networks and other similar systems. Theoretical solutions to the complexity problem is important because its answer can shed light on how the error on the training set is related to the error on the validation sets, without any tedious tests. However, the currently existing results relate only to binary classification and to classical neural networks. Further research is needed to extend the existing theoretical results to neuro-fuzzy systems like the one used in this paper.

6. CONCLUSIONS

The paper investigates the potential of the genetic adaptation schemes, within the framework of neuro-fuzzy prediction problems. The performances are compared with those corresponding to other optimization/clustering techniques.

Using a prediction benchmark time series, the neuro-fuzzy predictor performances have been checked for six different configuration strategies. The best prediction is achieved for the configuration that selects the membership functions centers using the *k*-means clustering algorithm and adapts the singletons with a gradient descent algorithm.

However, the genetic search offers a quite accurate prediction, with satisfactory generalization capabilities, especially when applied for the singletons adaptation. Therefore, it can represent a good alternative in the case of time-variance, discontinuity. Further research is needed to determine the complexity of the learning with these schemes.

ACKNOWLEDGMENT

The CNCSIS Grant 149/2005 “System for the analysis and prediction of genomic sequences based on neuro-fuzzy data-mining methods” has supported part of the research for this paper. This research is partly performed for the Romanian Academy prioritary grant "Cognitive systems and applications" ("Sisteme cognitive şi aplicaţii"); however, there was no financial support from this Grant for this research.

REFERENCES

1. AYOUBI, M., *Nonlinear System Identification Based on Neural Networks with Locally Distributed Dynamics and Application to Technical Process*, Ph.D. Thesis, VDI Verlag GmbH, Reihe **8**, 591, Düsseldorf, 1996.
2. BABUŠKA, R., VERBRUGGEN, H., *Neuro-fuzzy methods for nonlinear system identification*, Annual Reviews in Control, **27**, pp. 73–85, 2003.
3. BACK, T., FOGEL, D., MICHALEWICZ, Z., *Evolutionary Computation 2. Advanced Algorithms and Operators*, Institute of Physics Publishing, USA, 2000.
4. CUEVAS, E., ZALDIVAR, D., ROJAS, R., *Neurofuzzy prediction for visual tracking. Technical Report B-03-16*, http://page.mi.fu-berlin.de/~zaldivar/files/tr-b-03-16.pdf, (Accessed 17.05.2005).
5. FIRA, L. I., TEODORESCU, H. N., *Genome Bases Sequences Characterization by a Neuro-Fuzzy Predictor*, Proceedings of the IEEE-EMBS 2003 Conference, Cancun, Mexico, pp. 3555-3558, 2003.
6. HAYKIN, S. *Neural Networks-A Comprehensive Foundation*, McMillan College Publishing Company, New York, 2nd Edition, 1999.
7. LIAO, Y., MOODY, J., WU, L., *Applications of Artificial Neural Networks to Time Series Prediction*, Handbook of Neural Network Signal Processing, CRC Press, USA,2001.
8. NAUCK, D., KLAWONN, F., KRUSE, R., *Foundations of Neuro Fuzzy Systems* , John Wiley and Sons, USA, 1997.
9. TEODORESCU, H. N., YAMAKAWA, T., *Neuro-Fuzzy Systems: Hybrid configurations*, Fuzzy Logic. Implementation and applications, M.J. Patyra, D. Mlynek (Eds), Chichester, Wiley & Teubner, pp. 267-298, 1996.
10. TEODORESCU, H. N., FIRA, L. I., *A Hybrid Data-Mining Approach in Genomics and Text Structures*, Proceedings of the Third IEEE International Conference on Data Mining ICDM '03, Melbourne, Florida, USA, pp. 649-652, 2003.
11. TEODORESCU, H. N., FIRA, L. I., *Predicting the Genome Bases Sequences by means of distance sequences and a Neuro-Fuzzy Predictor*, Fuzzy Systems & A.I.-Reports and Letters, **7**, pp. 23-33, 2003.
12. TEODORESCU, H. N., FIRA, L. I., *On the Predictability of the Genomic Time Series*, Proceedings of the Third European Conference on Intelligent Systems and Technologies ECIT 2004, Iasi, Romania, CDROM, 2004.
13. YAMAKAWA, T., UCHINO, E., MIKI, T., KUSANAGI, H., *A Neo Fuzzy Neuron and Its Applications to System Identification and Prediction of the System Behaviour*, Proceedings of the 2nd International Conference on Fuzzy Logic and Neural Networks, Iizuka, Japan, pp. 477-483, 1992.
14. WALLER, J., TOIVONEN, H., *A Neuro-Fuzzy Model Predictive Controller Applied to a Ph-Neutralization Process*, Proceedings of IFAC World Congress on Automatic Control, Barcelona, Spain, 2002.
15. WEIGEND, A., GERSHENFELD, N., *Time Series Prediction: Forecasting the Future and Understanding the Past*, Santa Fe Institute, Addison-Wesley Publishing Company, USA, 1994.

Received, May 21, 2005