BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **15**, No 1

Sofia • 2015 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.1515/cait-2015-0003

A Distributed Adaptive Neuro-Fuzzy Network for Chaotic Time Series Prediction

Margarita Terziyska

Institute of Information and Communication Technologies, 1113 Sofia
Email: mterziyska@bas.bg

Abstract: *In this paper a Distributed Adaptive Neuro-Fuzzy Architecture* (*DANFA*) *model with a second order Takagi-Sugeno inference mechanism is presented. The proposed approach is based on the simple idea to reduce the number of the fuzzy rules and the computational load, when modeling nonlinear systems. As a learning procedure for the designed structure a two-step gradient descent algorithm with a fixed learning rate is used. To demonstrate the potentials of the selected approach, simulation experiments with two benchmark chaotic time systems − Mackey-Glass and Rossler are studied. The results obtained show an accurate model performance with a minimal prediction error.*

Keywords: *Distributed fuzzy neural network, fuzzy-neural models, nonlinear identification, Takagi-Sugeno fuzzy inference, NARX.*

1. Introduction

Neural networks and fuzzy logic proved to be universal approximators, which can estimate any nonlinear function to a prescribed accuracy. For identification of the complex nonlinear processes, different kinds of neuro-fuzzy architectures are also used. These structures have an advantage over traditional statistical estimation and adaptive control approaches. They estimate a function without the need of a detailed mathematical description of the functional dependency between inputs and outputs. Combining the neural networks and fuzzy systems in one unified framework has become popular in the last few years. The fusion of both combines the learning and computational ability of neural networks with the human like IF-THEN thinking and reasoning of a fuzzy system. This could be compared with the human brain [1] – the neural network concentrates on the structure of human brain, i.e., on the “hardware”, whereas the fuzzy logic system concentrates on the “software”.

A lot of architectures have been proposed in literature that combine fuzzy logic and neural networks. Some of the most popular are ANFIS [6] and DENFIS [4]. They are all composed of a set of if-then rules. In principle, the number of fuzzy rules depends exponentially on the number of inputs and membership functions. If n is the number of inputs in a fuzzy-neural system and m is the number of the membership functions, then the number of the generated fuzzy rules is m^n. Thus, the huge number of generated rules requires determination of a large number of parameters during the learning procedure. For instance, for a fuzzy inference system with 10 inputs, each one with two membership functions, the grid partitioning leads to $1024(=2^{10})$ rules, which is an extremely large number of rules for any practical applications.

In order to reduce the number of fuzzy rules without loss of accuracy, different fuzzy clustering approaches, such as fuzzy C-means [7, 8] and K-means [9] can be used. Besides, subtractive clustering and hyperplane clustering are proposed in [10, 11]. Evolving fuzzy systems [4, 13], such as DENFIS, includes evolving clustering and dynamically forms bases of fuzzy rules generated during the past instance of the learning process. The new AnYa neuro-fuzzy structure also belongs to the evolving fuzzy systems. This architecture works with the so-called cloud instead of fuzzy sets. This removes the need for training of the membership functions parameters. However, apriori data is needed to form the clouds [2].

Another possibility to reduce the number of fuzzy rules gives the self-constructing and self-organizing fuzzy-neural network structures [14, 15]. In this type of structures, during the training procedure, inactive rules are being removed, which consequently leads to reduction in the number of trained parameters.

In order to deal with the rule-explosion problem, hierarchical fuzzy neural networks could be used, but they employ a very complex learning method [5]. A method that compresses a fuzzy system with an arbitrarily large number of rules into a smaller fuzzy system by removing the redundancy in the fuzzy rule base is presented in [3]. As a result of this compression, the number of on-line operations during the fuzzy inference process is significantly reduced without compromising the solution. A review of most of the existing rule base reduction methods for fuzzy systems, a summary of their attributes and an introduction of the advanced techniques for formal presentation of the fuzzy systems based on Boolean matrices and binary relations, which facilitate the overall management of complexity, are presented in [12].

A simple method to reduce the number of the fuzzy rules is presented in the author’s previous works. In [16] the Semi Fuzzy Neural Network Model (SFNNM) is described in details, which has a small fuzzy rules base and also a reduced number of parameters (linear and nonlinear) that must be determined during the learning procedure.

In this paper a Distributed Adaptive Neuro-Fuzzy Architecture (DANFA) with a reduced number of the fuzzy rules is proposed. The main idea of the designed

DANFA structure is to distribute the input space through different fuzzy neural structures. The learning procedure is based on a two-step gradient descent method, for scheduling of the rules premise and consequent parameters. The potentials of the presented approach are evaluated in simulation experiments with two common benchmark chaotic systems – Mackey-Glass and Rossler.

2. Classical fuzzy neural network

In this section the so-called Classical Fuzzy Neural Network (CFNN) with a second order Takagi-Sugeno inference mechanism is described. It is named so in order to distinguish it from the proposed in the next section DANFA model. The structure of CFNN model is shown in Fig. 1.

Layer 1. This layer accepts the input variables and then the nodes in this layer only transmit the input values to the next layer directly.

Layer 2. Each node in this layer does fuzzification via a Gaussian membership function:

$$\mu^{(n)}_{X_{p,m}} = \exp\frac{-(X_p - C_{X_{p,m}})^2}{2\sigma^2_{X_{p,m}}}, \tag{1}$$

where X_p are the input values, $C_{Xp,m}$ and $\sigma_{Xp,m}$ are the center and the standard deviation of the Gaussian membership function.

Layer 3. This layer is a kind of a rules generator for it forms the fuzzy logic rules. Their number depends on the number of inputs p and the number of their fuzzy sets m, and it is calculated according to the expression $N = m^p$. In this layer, each node represents a fuzzy in the following form:

$$R^{(i)}: \text{if } x_1 \text{ is } \widetilde{A}^{(i)}_1 \text{ and } x_p \text{ is } \widetilde{A}^{(i)}_p \text{ then } f_y(k), \tag{2}$$

$$\begin{aligned} f^{(i)}_y(k) &= a^{(i)}_1 y(k-1) + a^{(i)}_2 y(k-2) + \ldots + a^{(i)}_{ny} y(k-n_y) + \ldots \\ &+ b^{(i)}_1 u(k) + b^{(i)}_2 u(k-1) + \ldots + b^{(i)}_{nu} u(k-n_u) + b^{(i)}_0. \end{aligned} \tag{3}$$

Layer 4. At the fourth layer an implication operation is realized:

$$\mu^{(n)}_{yq}(k+j) = \mu^{(n)}_{X_{1,m}}(k+j) * \mu^{(n)}_{X_{2,m}}(k+j) * \ldots * \mu^{(n)}_{X_{p,m}}(k+j). \tag{4}$$

Layer 5. At the fifth last layer the removal decision is made which consists of determining the value of the model output by the expression

$$\hat{y}(k+j) = \frac{\sum_{i=1}^{q} f^{(i)}_y(k+j)\mu^{(i)}_y(k+j)}{\sum_{i=1}^{q} \mu^{(i)}_y(k+j)}. \tag{5}$$

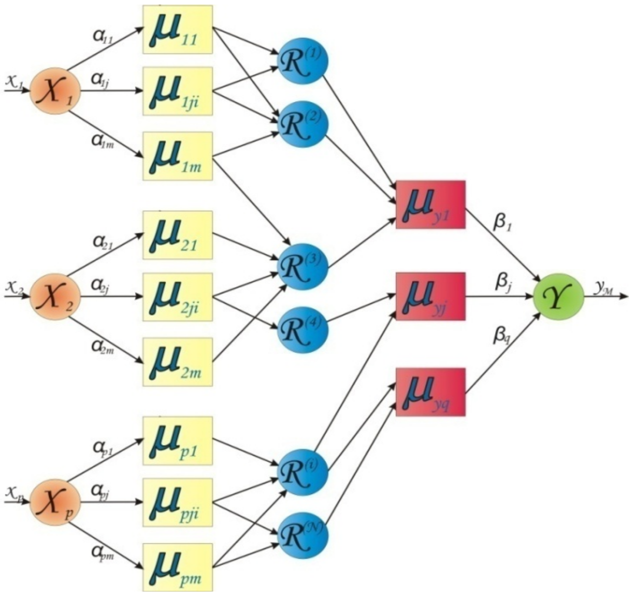

Fig. 1. Structure of the CFNN

3. DANFA model structure

The structure of the proposed DANFA model with a second order Takagi-Sugeno inference mechanism is shown in Fig. 2. DANFA model is a modification of the CFNN model with a second order Takagi-Sugeno inference mechanism, described in the previous section. Actually, DANFA model is a network from p CFNNs. DANFA model is a six-layer architecture with Takagi-Sugeno inference mechanism. The main idea behind it is to distribute the input signals in order to separate the fuzzy neural structures as can be seen in Fig. 2. In this way a network of neural fuzzy structures is obtained. Each of these neural-fuzzy structures acts as a separate sub-model and the global DANFA model is a set of p on the number of the sub-model. The output signal of a global DANFA model is computed as a sum of the output signals of the p CFNN models, which are obtained by using the equations (1)-(5). Thus, the global DANFA model output parameter is calculated by the following expression:

(6) $$\hat{y}_M(k+j) = \hat{y}_{M1}(k+j) + \hat{y}_{M2}(k+j) + \ldots + \hat{y}_{Mp}(k+j),$$

where $\hat{y}_{Mr}$ for $r = 1/p$ is obtained as follows:

(7) $$\hat{y}_{Mr}(k+j) = \frac{\sum_{i=1}^{q} f_r^{(i)}(k+j)\mu_r^{(i)}(k+j)}{\sum_{i=1}^{q} \mu_r^{(i)}(k+j)}.$$

In this paper it is chosen to construct the global DANFA model from two sub-models, i.e., the global DANFA model represents a network from two CFNNs. This is described by the block scheme given in Fig. 3. Thus, the output of DANFA model can be calculated by

$$\hat{y}_M = \hat{y}_{M1} + \hat{y}_{M2} = \frac{\sum_{i=1}^{q} f_u{}^{(i)} \mu_u{}^{(i)}}{\sum_{i=1}^{q} \mu_u{}^{(i)}} + \frac{\sum_{i=1}^{q} f_y{}^{(i)} \mu_y{}^{(i)}}{\sum_{i=1}^{q} \mu_y{}^{(i)}}, \tag{8}$$

where $\hat{y}_{M1}$ and $\hat{y}_{M2}$ are the output parameters respectively to CFNN1 and CFNN2 (see Fig. 3); f_u and f_y are the Sugeno output functions; μ_u and μ_y are the corresponding membership degrees of the quantization levels and can be obtained by the expressions:

$$f_u{}^{(i)} = b_{1u}{}^{(i)} u(k) + b_{2u}{}^{(i)} (k-1) + \ldots + b_{n_u u}{}^{(i)} u(k-n_u) + b_{ou}{}^{(i)}, \tag{9}$$

$$\mu_u{}^{(i)} = \mu_{u1j}{}^{(i)} * \mu_{u2j}{}^{(i)} * \ldots * \mu_{upj}{}^{(i)}, \tag{10}$$

$$f_y{}^{(i)} = a_{1y}{}^{(i)} y(k-1) + a_{2y}{}^{(i)} y(k-2) + \ldots + a_{n_y y} (k-n_y) + a_{oy}{}^{(i)}, \tag{11}$$

$$\mu_y{}^{(i)} = \mu_{y1j}{}^{(i)} * \mu_{y2j}{}^{(i)} * \ldots * \mu_{ypj}{}^{(i)}. \tag{12}$$

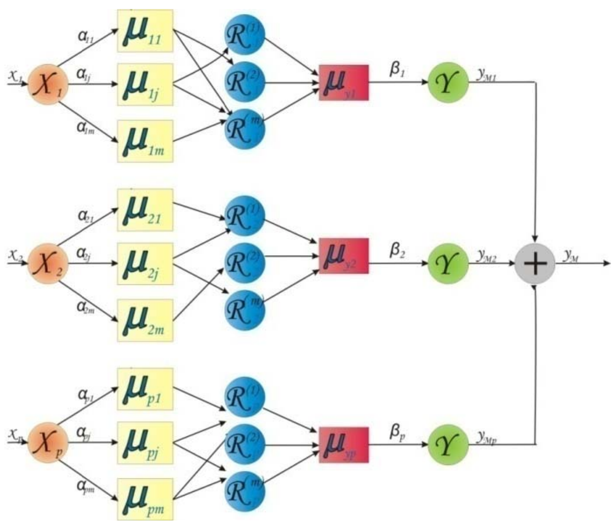

Fig. 2. Structure of the proposed DANFA architecture

From Fig. 3 it is clear that CFNN1 is a fuzzy neural network which cares for the vector regressor *u* and CFNN2 is a fuzzy neural network which cares for the vector regressor *y*. Each of these fuzzy-neural networks (CFNN1 and CFNN2) have two input variables with three fuzzy sets, respectively $N = 3^2 = 9$ rules.

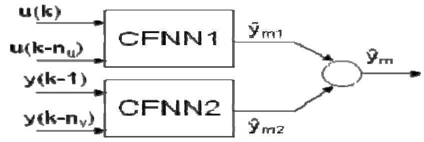

Fig. 3. Block scheme of the proposed DANFA structure

The main advantage of DANFA is that it operates with a small number of rules and has respectively a smaller number of parameters for learning. The formula for calculating the number of rules in DANFA model is $N= pn^m$, where p is the number of used CFNNs, n is the number of input variables, m is their fuzzy set. It is clear, that if one wants to realize a NARX model with 4 inputs and 3 fuzzy sets, then the CFNN model needs 81 fuzzy rules, while the DANFA model needs only 18 fuzzy rules. Also, during the learning procedure in CFNN model the values of 1053 parameters are computed, while in DANFA model the values of only 156 parameters are computed.

The learning algorithm for DANFA model is very simple. It is based on minimization of the instant error measurement function between the real plant output and the process output, calculated by the DANFA model

(13) $$E(k) = \frac{(y(k) - \hat{y}(k))^2}{2} = \frac{(y(k) - (\hat{y}_{M1} + \hat{y}_{M2}))^2}{2},$$

where $y(k)$ denotes the measured real plant output and $\hat{y}(k)$ is the sum of the calculated by the two fuzzy neural networks output parameters $\hat{y}_{M1}$ and $\hat{y}_{M2}$. The algorithm performs a two-steps gradient learning procedure. Assuming that β_{uij} is an adjustable i-th coefficient for Sugeno function f_u into j-th activated rule for CFNN1, then the general parameter learning rule for the consequent parameters is

(14) $$\beta_{uij}(k+1) = \beta_{uij}(k) + \eta\left(-\frac{\partial E}{\partial \beta_{uij}}\right).$$

After calculating the partial derivatives, the final recurrent predictions for each adjustable coefficient β_{uij} and the free coefficient are obtained by the following equations:

(15) $$\beta_{uij}(k+1) = \beta_{uij}(k) + \eta\varepsilon(k)\overline{\mu}_u^{(j)}(k)x_{1i}(k),$$

(16) $$\beta_{0uj}(k+1) = \beta_{0uj}(k) + \eta\varepsilon(k)\overline{\mu}_u^{(j)}(k).$$

The output error E can be used back directly to the input layer, where the premise adjustable parameters are available (center – c_{ij} and the deviation – σ_{ij} of a Gaussian fuzzy set). The error E is propagated through the links composed by the corresponding membership degrees, where the link weights are units. Hence, the learning rule for the second group of adjustable parameters in the input layer can be expressed by the same learning rule:

(17) $$c_{uij}(k+1) = c_{uij}(k) + \eta\varepsilon(k)\overline{\mu}_u^{(i)}(k)[f_u^{(i)} - \hat{y}_M(k)]\frac{[x_{1i}(k) - c_{uij}(k)]}{c_{uij}^2(k)},$$

(18) $$\sigma_{uij}(k+1) = \sigma_{uij}(k) + \eta\varepsilon(k)\overline{\mu}_u^{(i)}(k)[f_u^{(i)} - \hat{y}_M(k)]\frac{[x_{1i}(k) - \sigma_{uij}(k)]^2}{\sigma_{uij}^3(k)}.$$

To obtain the parameters for CFNN2, similar formulas are used, namely:

(19) $$\beta_{yij}(k+1) = \beta_{yij}(k) + \eta\varepsilon(k)\overline{\mu}_y^{(j)}(k)x_{2i}(k),$$

(20) $$\beta_{0yj}(k+1) = \beta_{0yj}(k) + \eta\varepsilon(k)\overline{\mu}_y^{(j)}(k),$$

$$c_{yij}(k+1)=c_{yij}(k)+\eta\varepsilon(k)\bar{\mu}_y^{(i)}(k)[f_y^{(i)}-\hat{y}_M(k)]\frac{[x_{2i}(k)-c_{yij}(k)]}{c_{yij}^2(k)}, \tag{21}$$

$$\sigma_{yij}(k+1)=\sigma_{yij}(k)+\eta\varepsilon(k)\bar{\mu}_y^{(i)}(k)[f_y^{(i)}-\hat{y}_M(k)]\frac{[x_{2i}(k)-\sigma_{yij}(k)]^2}{\sigma_{yij}^3(k)}. \tag{22}$$

4. Simulation results

Two benchmark chaotic systems (Mackey-Glass and Rossler chaotic time series) are chosen to demonstrate the ability of the proposed DANFA model. The series will not converge or diverge, and the trajectory is highly sensitive to initial conditions. The learning rate η has a fixed value of 0.04 for each of the two fuzzy neural networks.

Mackey-Glass (MG) chaotic time series is described by the time-delay differential equation:

$$x(i+1)=\frac{x(i)+ax(i-s)}{(1+x^c(i-s))-bx(i)}, \tag{23}$$

where $a = 0.2$, $b = 0.1$, $c = 10$; the initial conditions are $x(0) = 0.1$ and $s = 17$ s.

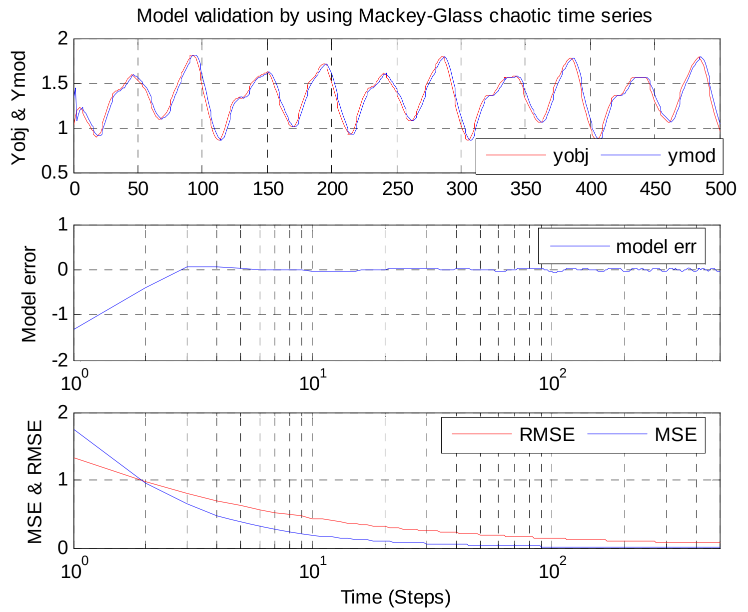

Fig. 4. Model validation by using Mackey-Glass chaotic time series

Results on model validation by using Mackey-Glass chaotic time series are shown in Fig. 4. As it can be seen, the proposed model structure predicts accurately the generated time series, with a minimum prediction error and fast transient response of the RMSE, reaching a value closer to zero. The value of the MSE in the

50-th time step is 0.045. For the sake of clarity the model errors, MSE and RMSE are presented in logarithmic scale.

Another test of the DANFA model proposed is made with the help of Rossler chaotic time series. These series are described by three coupled first-order differential equations:

$$\frac{dx}{dt} = -y - z, \quad \frac{dy}{dt} = x + ay, \quad \frac{dz}{dt} = b + z(x - c). \tag{24}$$

where a = 0.2, b = 0.4, c=5.7; initial conditions are x_0 = 0.1, y_0 = 0.1, z_0 = 0.1. The results are given in Fig. 4.

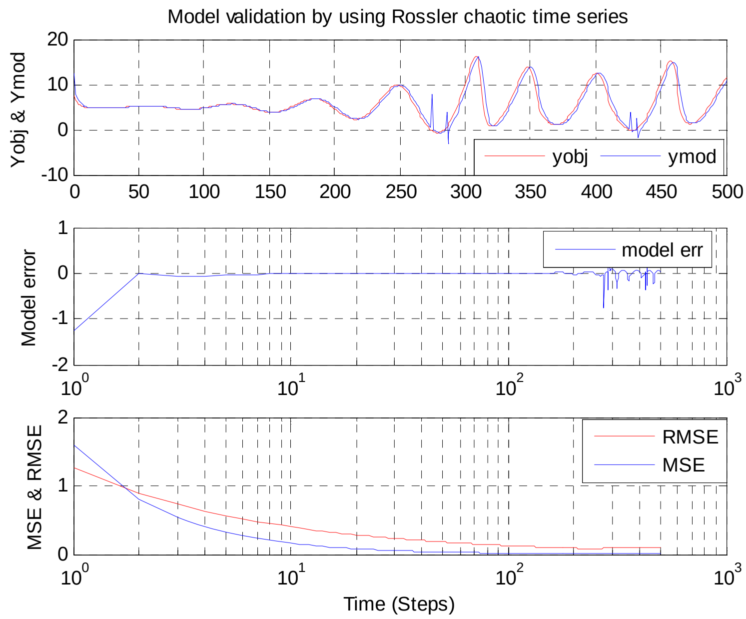

Fig. 5. Model validation by using Rossler chaotic time series

For clarity, the results obtained with CFNN are not represented in Figs 4 and 5. Comparison between CFNN and DANFA models is still made and it is shown in Table 1 below. The results in Table 1 are for Mackey-Glass chaotic time series prediction. As it can be seen, the absolute value of the current prediction error (Pred_err in Table 1) obtained with DANFA model is smaller than this obtained with CFNN model. The greater accuracy of DANFA model can be easily explained. In (3) the free parameter b_0 plays the role of a disturbance filter. This can easily be proved if (3) is transformed into the following form:

$$f_y^{(i)}(k) = a_1^{(i)} y(k-1) + a_2^{(i)} y(k-2) + \ldots + a_{ny}^{(i)} y(k-n_y) + \ldots \\ + b_1^{(i)} u(k) + b_2^{(i)} u(k-1) + \ldots + b_{nu}^{(i)} u(k-n_u) + d^{(i)}(k). \quad (25)$$

This conversion is valid in $T(q^{-1}) = 1$. In (25) $d(k)$ is unknown disturbance that is defined by

$$d(k) = \frac{T(q^{-1})}{\Delta(q^{-1})} \upsilon(k). \quad (26)$$

Thus, the CFNN model works with one disturbance filter, while DANFA model has two filters (see (9) and (11)) and this is the reason the latter to be more accurate.

Table 1. Comparison between CFNN and DANFA model

Steps	DANFA model			CFNN model		
	Pred_err	MSE	RMSE	Pred_err	MSE	RMSE
50	–0.019613	0.053553	0.23141	–0.04702	0.0091595	0.095705
100	–0.067288	0.027756	0.1659	–0.15122	0.0077459	0.088011
150	–0.01297	0.01893	0.13759	–0.051306	0.0079593	0.089513
200	–0.058875	0.014637	0.12098	–0.12744	0.077992	0.088313
250	–0.02409	0.012033	0.1097	–0.051261	0.0077583	0.088081
300	-0.04841	0.010343	0.10156	–0.12544	0.0079122	0.08895
350	–0.039738	0.0090874	0.095328	–0.10179	0.0077017	0.087759
400	–0.045243	0.0081819	0.090454	–0.11864	0.007838	0.088471
450	–0.03445	0.0074437	0.086277	–0.095164	0.007659	0.087516
500	–0.040763	0.0068844	0.082973	–0.11223	0.007769	0.088143

5. Conclusions

In this paper the architecture and learning procedure of DANFA model are presented. The proposed model is a modification of CFNN model with a second order Takagi-Sugeno inference mechanism and it has all its features. The main idea in DANFA is to distribute the inputs and thus to reduce the number of the used fuzzy rules. Two benchmark chaotic systems (Mackey-Glass and Rossler chaotic time series) are chosen to demonstrate the ability of this model. The proposed DANFA model predicts accurately the generated time series, with a minimal prediction error and fast transient response of RMSE, reaching values closer to zero. The main advantage of DANFA is that it operates by a small number of rules and has respectively a smaller number of parameters for learning. Thus, it carries out the modelling of nonlinear systems with considerably less calculation in comparison with the CFNN model. Moreover, DANFA has other advantages – it is more accurate than CFNN model and it does not require any apriori data and is not bounded by additional procedures, such as clustering. This makes it suitable for real time applications, such as predictive controllers.

Acknowledgments: The research work reported in the paper is partly supported by the project **AComIn** "Advanced Computing for Innovation", grant 316087, funded by the FP7 Capacity Programme (Research Potential of Convergence Regions). The author would like to acknowledge the support of Project DO1-192 of the National Research Fund, Ministry of Education and Science of Bulgaria, as well.

References

1. C h e n n a k e s a v a, R. A. Fuzzy Logic and Neural Networks: Basic Concepts and Applications. New Age International, Pvt., Ltd., Publishers, 1 December 2008.
2. A n g e l o v, P. Autonomous Learning Systems. Wiley, 2013.
3. G e g o v, A. Complexity Management in Fuzzy Systems. – Studies in Fuzziness and Soft Computing, Vol. **211**, 2007.
4. K a s a b o v, N. K., Q u n S o n g. DENFIS: Dynamic Evolving Neural-Fuzzy Inference System and Its Application for Time-Series Prediction. – IEEE Transactions on Fuzzy Systems, Vol. **10**, April 2002, No 2.
5. Y u, W., M. A. M o r e n o-A r m e n d a r i z, F. O. R o d r i g u e z. System Identification Using Hierarchical Fuzzy Neural Networks with Stable Learning Algorithm. – Journal of Intelligent & Fuzzy Systems, Vol. **18**, 2007, pp. 171-183.
6. J a n g, J.-S. R. ANFIS: Adaptive-Network-Based Fuzzy Inference System. – IEEE Transactions on System, Man, and Cybernetics, Vol. **23**, May 1993, No 5, pp. 665-685.
7. C h u a n g, K.-S. et al. Fuzzy c-Means Clustering with Spatial Information for Image Segmentation. – Computerized Medical Imaging and Graphics, Vol. **30**, 2006, pp. 9-15.
8. Ž a l i k, K. R. Fuzzy C-Means Clustering and Facility Location Problems. – Artificial Intelligence and Soft Computing, 2006.
9. G a l l u c c, L., et al. Graph Based *k*-Means Clustering. – Signal Processing, Vol. **92**, September 2012, Issue 9, pp. 1970-1984.
10. M e h r a b i a n, A. R., et al. Neuro-Fuzzy Modeling of Superheating System of a Steam Power Plant. – Artificial Intelligence and Applications, 13-16 February, 2006.
11. P a n e l l a, M. A Hierarchical Procedure for the Synthesis of ANFIS Networks. – Advances in Fuzzy Systems, Vol. **2012**.
12. G e g o v, A., N. G o b a l a k r i s h n a n. Advanced Inference in Fuzzy Systems by Rule Base Compression. Mathware & Soft Computing, Vol. **14**, 2007, pp. 201-216.
13. K a s a b o v, N. K., D. F i l e v. Evolving Intelligent Systems: Methods, Learning & Applications. – In: Proc of International Symposium of Evolving Fuzzy Systems, September 2006.
14. A l l e n d e-C i d, H., et al. Self-Organizing Neuro-Fuzzy Inference System. – In: Proc. of Iberoamerican Congress on Pattern Recognition CIARP, 2008, pp. 429-436.
15. F e r r e y r a, A., J. d e J e s u s R u b i o. A New On-Line Self-Constructing Neural Fuzzy Network. – In: Proc. of 45th IEEE Conference on Decision & Control, San Diego, CA, USA, 13-15 December 2006.
16. T e r z i y s k a, M., L. D o u k o v s k a, M. P e t r o v. Implicit GPC Based on Semi Fuzzy Neural Network Model. – In: Proc. of 7th IEEE International Conference Intelligent Systems IS'2014, 24-26 September 2014, Warsaw, Poland, Vol. **1**. Mathematical Foundations, Theory, Analyses, ISBN 978-3-319-11312-8, pp. 695-706.