UDC 004.032.26

Ye. Bodyanskiy, O. Boiko

EVOLVING MULTILAYER NEURO-FUZZY SYSTEM AND ITS LEARNING

Abstract. In this work a neuro-fuzzy system with all its parameters adjustment is proposed. It is used as a node of the evolving multilayer system. The system architecture can evolve in online mode as the synaptic weights, centers and widths parameters of the neuro-fuzzy nodes are adjusted improving approximation properties of the system.

Keywords: evolving system, computational intelligence, neuro-fuzzy system, multilayer neural network, data mining

Introduction

Nowadays hybrid systems of computational intelligence are widely used for solving different Data Mining tasks: pattern recognition, identification, emulation, etc. In practice it's a common situation when data come sequentially in online mode. This task is concerned by an intensively developing field, known as Dynamic Data Mining and Data Stream Mining [1]. The most effective systems for solving these tasks are evolving computational intelligence systems (ECIS) [2-6], that adjust their architecture during learning process. It should be noticed that the base of the majority of the known ECIS are multilayered neuro-fuzzy systems of TSK- and ANFIS-type [4-8].

Conventionally "learning" is defined as a synaptic weights adjustment process with the given learning criterion optimization. The quality of learning can be significantly improved by adjusting not only synaptic weights of the system but also its architecture, and in the case of NFSs – also its membership functions parameters.

In this paper an evolving multilayer neuro-fuzzy system is proposed. This system is trained using simple learning procedures and it adjusts all its parameters, improving its approximation properties.

The evolving multilayer neuro-fuzzy system architecture

The architecture of the evolving multilayer neuro-fuzzy system is shown in Figure 1. To the zero (receptive) layer of the system a

$(n\times 1)$-dimensional vector of input signals $\mathbf{x}(k)=(x_1(k), x_2(k), ..., x_n(k))^{\mathrm{T}}$ is fed (here $k=1, 2, ..., N$ is either the current discrete time index or the observation number in training set). Then this vector is fed to the first hidden layer that contains $n_1=c_n^2$ nodes, each having two inputs.

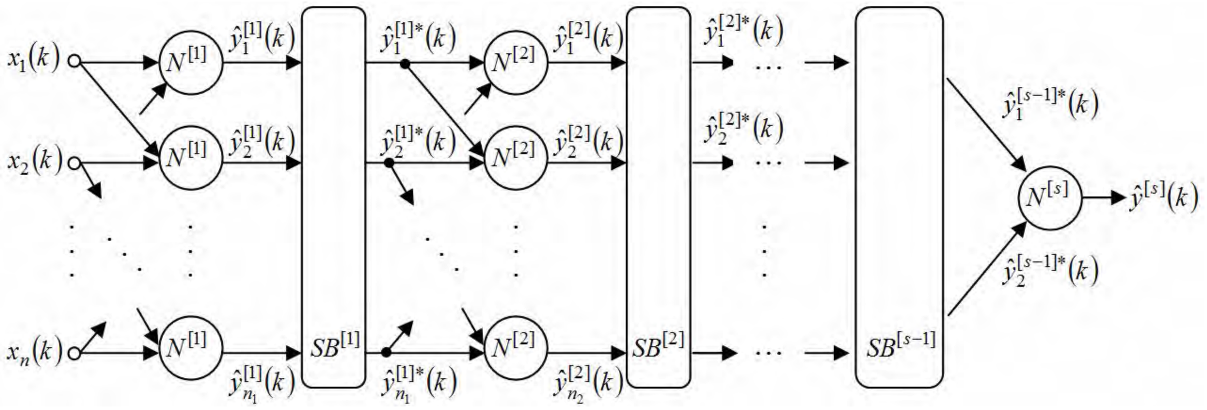
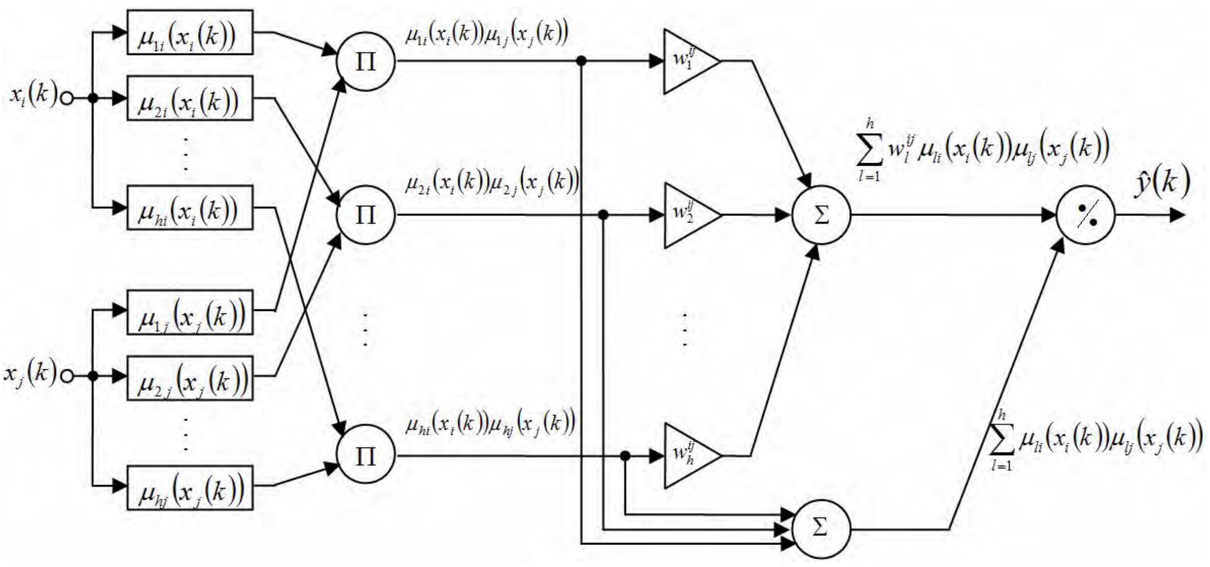


Fig. 1. Evolving multilayer neuro-fuzzy system

Among the output signals $\hat{y}_l^{[1]}(k)$ ($l=1, 2, ..., 0{,}5n(n-1)=c_n^2$) of the first layer nodes $N^{[1]}$ the selection block of the first layer $SB^{[1]}$ selects n_1^* ($n_1^*\le n$) most precise signals in the sense of accepted criterion, usually by the mean squared error $\sigma^2_{y_l^{[1]}}$. Then from these n_1^* best outputs n_2 pairwise combinations $\hat{y}_l^{[1]*}(k), \hat{y}_p^{[1]*}(k)$ are formed (usually $n\le n_2\le 2n$). These signals are fed to the second hidden layer, formed by nodes $N^{[2]}$, similar to the neurons $N^{[1]}$. Among the output signals $\hat{y}_l^{[2]}(k)$ of the second hidden layer the selection block of the second hidden layer $SB^{[2]}$ selects only signals that by accuracy are better than $\hat{y}_1^{[1]*}(k)$, i. e. that are better than the best signal of the first hidden layer. The third hidden layer forms signals that are more accurate than the best signal $\hat{y}_1^{[2]*}(k)$ and so on. The process of the system evolution continues until only two signals $\hat{y}_1^{[s-1]*}(k)$ and $\hat{y}_2^{[s-1]*}(k)$ are formed on the output of the selection block $SB^{[s-1]}$. These two signals are then fed to the single output node $N^{[s]}$, that calculates the output signal $\hat{y}^{[s]}(k)$ of the system.

Neuro-fuzzy network with all parameters adjustment as a node of the evolving multilayer system

The architecture of the node proposed as a neuron of the considered evolving multilayer system is shown in Figure 2.



$x_i(k)$
$x_j(k)$
$\mu_{1i}(x_i(k))$
$\mu_{2i}(x_i(k))$
$\mu_{hi}(x_i(k))$
$\mu_{1j}(x_j(k))$
$\mu_{2j}(x_j(k))$
$\mu_{hj}(x_j(k))$
Π
$\mu_{1i}(x_i(k))\mu_{1j}(x_j(k))$
$\mu_{2i}(x_i(k))\mu_{2j}(x_j(k))$
$\mu_{hi}(x_i(k))\mu_{hj}(x_j(k))$
w_1^{ij}
w_2^{ij}
w_h^{ij}
Σ
$\sum_{l=1}^{h} w_l^{ij}\mu_{li}(x_i(k))\mu_{lj}(x_j(k))$
$\sum_{l=1}^{h} \mu_{li}(x_i(k))\mu_{lj}(x_j(k))$
$\%$
$\hat{y}(k)$

Fig. 2. Node of the evolving multilayer system

The node architecture is in fact a Wang–Mendel neuro-fuzzy system [30,31] with two inputs $x_i(k)$ and $x_j(k)$, five sequentially connected layers for information processing and one output $\hat{y}(k)$. To the input of node a two-dimensional vector of input signals $\mathbf{x}(k)=(x_i(k),x_j(k))^{\mathrm{T}}$ is fed. The first layer of a node provides fuzzification of the input variables. The elements of the first layer compute membership levels $0<\mu_{li}(x_i(k))\leq 1$, $0<\mu_{lj}(x_j(k))\leq 1$, $l=1,2,...,h$. To avoid appearing of "gaps" in the fuzzified space while using scatter partitioning of input space [7] the bell-shaped constructions with non-strictly local receptive support are usually used as membership functions. Mostly the Gaussians are used as membership functions of the first layer

$$\mu_{li}(x_i(k))=\exp\left(-\frac{(x_i(k)-c_{li}(k))^2}{2\sigma_{li}^2(k)}\right),\ \mu_{lj}(x_j(k))=\exp\left(-\frac{(x_j(k)-c_{lj}(k))^2}{2\sigma_{lj}^2(k)}\right),\quad (1)$$

where $c_{li}(k)$, $c_{lj}(k)$ are parameters that define the centers of the membership functions, $\sigma_{li}(k)$, $\sigma_{lj}(k)$ are width parameters of these functions.

The second layer contains h multiplication units and forms two-dimensional radial basis activation functions $\mu_{li}(x_i(k))\mu_{lj}(x_j(k))$. This layer provides aggregation of the membership levels, that are computed in the first layer. Outputs of the second layer h are values

$$\tilde{x}_l(k) = \mu_{li}(x_i(k))\mu_{lj}(x_j(k)). \quad (2)$$

The third layer is one of synaptic weights that are adjusted during learning process. The number of the membership functions h on each input defines the number of weights. The outputs of the third layer are values

$$w_l^{ij}\mu_{li}(x_i(k))\mu_{lj}(x_j(k)) = w_l^{ij}\tilde{x}_l(k). \quad (3)$$

The fourth layer contains two summation units. In this layer the sums of the output signals of the second and the third layers are computed

$$\sum_{l=1}^{h}\mu_{li}(x_i(k))\mu_{lj}(x_j(k)) = \sum_{l=1}^{h}\tilde{x}_l(k), \quad \sum_{l=1}^{h}w_l^{ij}\mu_{li}(x_i(k))\mu_{lj}(x_j(k)) = \sum_{l=1}^{h}w_l^{ij}\tilde{x}_l(k). \quad (4)$$

Finally, in the fifth (output) layer normalization is realized and the output signal of node is computed

$$\hat{y}(k) = \frac{\sum_{l=1}^{h} w_l^{ij}\mu_{li}(x_i(k))\mu_{lj}(x_j(k))}{\sum_{p=1}^{h}\mu_{pi}(x_i(k))\mu_{pj}(x_j(k))} = \frac{\sum_{l=1}^{h} w_l^{ij}\tilde{x}_l(k)}{\sum_{p=1}^{h}\tilde{x}_p(k)} = \sum_{l=1}^{h} w_l^{ij}\frac{\tilde{x}_l(k)}{\sum_{p=1}^{h}\tilde{x}_p(k)} = \quad (5)$$

$$= \sum_{l=1}^{h} w_l^{ij}\phi_l^{ij}(\mathbf{x}(k)) = (w^{ij})^{\mathrm{T}}\phi^{ij}(\mathbf{x}(k)),$$

where $\varphi^{\mathbf{ij}}(\mathbf{x}(k)) = (\varphi_1^{ij}(\mathbf{x}(k)), \ldots, \varphi_p^{ij}(\mathbf{x}(k)))^{\mathrm{T}}$, $\mathbf{w}^{\mathbf{ij}} = (w_1^{ij}, \ldots, w_h^{ij})^{\mathrm{T}}$,

$$\varphi_l^{ij}(\mathbf{x}(k)) = \mu_{li}(x_i(k))\mu_{lj}(x_j(k))\left(\sum_{p=1}^{h}\mu_{pi}(x_i(k))\mu_{pj}(x_j(k))\right)^{-1}.$$

The adjustment procedures for all parameters of the system

Considering that the reference signal $\hat{y}_s^{[1]}(k)$ of every node of the system depends linearly on the adjusted synaptic weights w_l^{ij}, for their adjustment we can use either the conventional least squares method, or its recurrent form. If training data is non-stationary, for weights adjustment it is reasonable to use the exponentially weighted recurrent least-squares method in the form

$$\begin{cases} \mathbf{w}^{\mathbf{ij}}(k)=\mathbf{w}^{\mathbf{ij}}(k-1)+\dfrac{\mathbf{P}^{\mathbf{ij}}(k-1)\left(y(k)-\left(\mathbf{w}^{\mathbf{ij}}(k-1)\right)^{\mathrm{T}}\varphi^{\mathbf{ij}}(\mathbf{x}(k))\right)\varphi^{\mathbf{ij}}(\mathbf{x}(k))}{\alpha+\left(\varphi^{\mathbf{ij}}(\mathbf{x}(k))\right)^{\mathrm{T}}\mathbf{P}^{\mathbf{ij}}(k-1)\varphi^{\mathbf{ij}}(\mathbf{x}(k))}, \\ \mathbf{P}^{\mathbf{ij}}(k)=\dfrac{1}{2}\left(\mathbf{P}^{\mathbf{ij}}(k-1)-\dfrac{\mathbf{P}^{\mathbf{ij}}(k-1)\varphi^{\mathbf{ij}}(\mathbf{x}(k))\left(\varphi^{\mathbf{ij}}(\mathbf{x}(k))\right)^{\mathrm{T}}\mathbf{P}^{\mathbf{ij}}(k-1)}{\alpha+\left(\varphi^{\mathbf{ij}}(\mathbf{x}(k))\right)^{\mathrm{T}}\mathbf{P}^{\mathbf{ij}}(k-1)\varphi^{\mathbf{ij}}(\mathbf{x}(k))}\right) \end{cases} \quad (6)$$

(here $0<\alpha\le 1$ is forgetting factor, $y(k)$ is reference signal) or the exponentially weighted gradient learning algorithm for neuro-fuzzy systems [22]

$$\begin{cases} \mathbf{w}^{\mathbf{ij}}(k)=\mathbf{w}^{\mathbf{ij}}(k-1)+\left(\beta^{ij}(k)\right)^{-1}\left(y(k)-\left(\mathbf{w}^{\mathbf{ij}}(k-1)\right)^{\mathrm{T}}\varphi^{\mathbf{ij}}(\mathbf{x}(k))\right)\varphi^{\mathbf{ij}}(\mathbf{x}(k)), \\ \beta^{ij}(k)=\alpha\beta^{ij}(k-1)+\left\|\varphi^{\mathbf{ij}}(\mathbf{x}(k))\right\|^{2}, 0\le\alpha\le 1. \end{cases} \quad (7)$$

The centers and weights parameters adjustment can be realized using the gradient procedures of the learning criterion minimization

$$E_s^{[1]}(k)=\frac{1}{2}\left(y(k)-\hat{y}_s^{[1]}(k)\right)^2=\frac{1}{2}\left(y(k)-\left(\mathbf{w}^{\mathbf{ij}}(k)\right)^{\mathrm{T}}\varphi^{\mathbf{ij}}(\mathbf{x}(k))\right)^2 \quad (8)$$

in the form

$$\begin{cases} c_{ri}(k)=c_{ri}(k-1)-\eta_c\dfrac{\partial E_s^{[1]}(k)}{\partial c_{ri}}, \\ \tilde{\sigma}_{ri}^2(k)=\tilde{\sigma}_{ri}^2(k-1)-\eta_\sigma\dfrac{\partial E_s^{[1]}(k)}{\partial\tilde{\sigma}_{ri}^2}, \end{cases} \quad (9)$$

where $r=1,2,...,h$; η_c, η_σ are learning rates for the center and the width parameter correspondingly, $\tilde{\sigma}_{ri}^2(k)=-0{,}5\sigma_{ri}^{-2}(k)$. It easy to see that

$$\begin{cases} \dfrac{\partial E_s^{[1]}(k)}{\partial c_{ri}}=\left(\left(\mathbf{w}^{\mathbf{ij}}(k)\right)^{\mathrm{T}}\varphi^{\mathbf{ij}}(\mathbf{x}(k))-y(k)\right)\sum\limits_{l=1}^{h}w_l^{ij}(k)\dfrac{\partial\varphi_l^{ij}(\mathbf{x}(k))}{\partial c_{ri}}, \\ \dfrac{\partial E_s^{[1]}(k)}{\partial\tilde{\sigma}_{ri}^2}=\left(\left(\mathbf{w}^{\mathbf{ij}}(k)\right)^{\mathrm{T}}\varphi^{\mathbf{ij}}(\mathbf{x}(k))-y(k)\right)\sum\limits_{l=1}^{h}w_l^{ij}(k)\dfrac{\partial\varphi_l^{ij}(\mathbf{x}(k))}{\partial\tilde{\sigma}_{ri}^2}. \end{cases} \quad (10)$$

The derivatives $\dfrac{\partial\varphi_l^{ij}(\mathbf{x}(k))}{\partial c_{ri}}$ and $\dfrac{\partial\varphi_l^{ij}(\mathbf{x}(k))}{\partial\tilde{\sigma}_{ri}^2}$ in Eq. (10) can be written in the form

$$\begin{cases} \dfrac{\partial \varphi_l(\mathbf{x}(k))}{\partial c_{ri}} = \dfrac{\delta_{lr} \sum\limits_{p=1}^{h} \mu_{pi}(x_i(k))\mu_{pj}(x_j(k)) - \mu_{li}(x_i(k))\mu_{lj}(x_j(k))}{\left(\sum\limits_{p=1}^{h} \mu_{pi}(x_i(k))\mu_{pj}(x_j(k)) \right)^2} \mu_{rj}(x_j(k)) \dfrac{\partial \mu_{ri}(x_i(k))}{\partial c_{ri}}, \\ \dfrac{\partial \varphi_l(\mathbf{x}(k))}{\partial \tilde{\sigma}_{ri}^2} = \dfrac{\delta_{lr} \sum\limits_{p=1}^{h} \mu_{pi}(x_i(k))\mu_{pj}(x_j(k)) - \mu_{li}(x_i(k))\mu_{lj}(x_j(k))}{\left(\sum\limits_{p=1}^{h} \mu_{pi}(x_i(k))\mu_{pj}(x_j(k)) \right)^2} \mu_{rj}(x_j(k)) \dfrac{\partial \mu_{ri}(x_i(k))}{\partial \tilde{\sigma}_{ri}^2}, \end{cases} \qquad (11)$$

where δ_{lr} is the Kronecker delta.

The derivatives $\frac{\partial \mu_{ri}(x_i(k))}{\partial c_{ri}}$ and $\frac{\partial \mu_{ri}(x_i(k))}{\partial \tilde{\sigma}_{ri}^2}$, defined on the base of Eq. (1), can be written in the form

$$\begin{cases} \dfrac{\partial \mu_{ri}(x_i(k))}{\partial c_{ri}} = \dfrac{x_i(k) - c_{ri}(k)}{\sigma_{ri}^2(k)} \exp\left(-\dfrac{(x_i(k) - c_{ri}(k))^2}{2\sigma_{ri}^2(k)} \right), \\ \dfrac{\partial \mu_{ri}(x_i(k))}{\partial \tilde{\sigma}_{ri}^2} = (x_i(k) - c_{ri}(k))^2 \exp\left(-\dfrac{(x_i(k) - c_{ri}(k))^2}{2\sigma_{ri}^2(k)} \right). \end{cases} \qquad (12)$$

In such a way we can adjust all synaptic weights, centers and width parameters of the membership functions of the first hidden layer nodes of the system. The nodes of next layers are adjusted similarly to the nodes of the first hidden layer but the node inputs of the sth layer are pairwise combinations of the signals $\hat{y}_l^{[s-1]*}, \hat{y}_p^{[s-1]*}$, formed by the selection block $SB^{[s-1]}$. The reference signal $y(k)$ is one for all elements of the evolving multilayer system.

Computational experiments

The efficiency of the proposed approach was demonstrated by solving the problem of the emulation of the dynamic object [32] that is described by the equation

$$y_p(k+1) = f(y_p(k), y_p(k-1), y_p(k-2), u(k), u(k-1)). \qquad (13)$$

The emulation results are presented in Figure 3 (actual values are marked with dotted line, emulation results are marked with solid line).

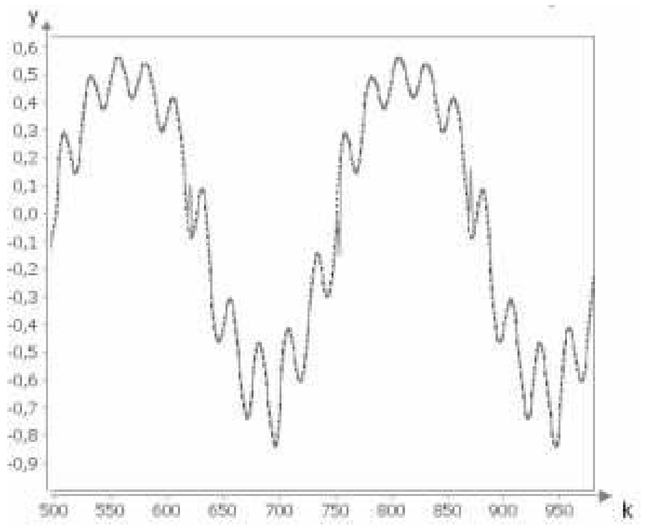


Fig. 3 – A fragment of the emulation results

The training set for the experiment was obtained using Eq. (12) with control signals $f(x_1, x_2, x_3, x_4, x_5) = \frac{x_1 x_2 x_3 x_5 (x_3 - 1) + x_4}{1 + x_3^2 + x_2^2}$ and $u(k) = 0.8\sin(2\pi k/250) + 0.2\sin(2\pi k/25)$. As the prediction quality criterion the mean squared error (MSE) was used. This experiment was carried out for several values of h. For $h = 5$ the mean squared error is equal to 0,00175, for $h = 10$ MSE=0,00120, for $h = 20$ MSE=0,00115.

Conclusion

In the paper the neuro-fuzzy network with all its parameters adjustment is proposed as a node of the evolving multilayer system. The system architecture can evolve in online mode as the synaptic weights, centers and widths parameters of the proposed neuro-fuzzy nodes are adjusted.

REFERENCES

1. Angelov P. Evolving Rule-based Models: A Tool for Design of Flexible Adaptive Systems. – Heidelberg-New York: Springer-Verlag, 2002. – 213 p.
2. Kasabov N. Evolving Connectionist Systems. – London: Springer-Verlag, 2003. – 307 p.
3. Lughofer E. Evolving Fuzzy Systems – Methodologies, Advanced Concepts and Applications. – Berlin-Heidelberg: Springer-Verlag, 2011. – 454 p.
4. Takagi T., Sugeno M. Fuzzy identification of systems and its applications to modeling and control // IEEE Trans. on Systems, Man, and Cybernetics. – 1985. – 15. – P. 116-132.
5. Jang R. J.-S. ANFIS: Adaptive-network-based fuzzy inference systems // IEEE Trans. on Systems, Man, and Cybernetics. – 1993. – 23. – P. 665-685.

6. Sugeno M., Kang G. T. Structure identification of fuzzy model // Fuzzy Sets and Systems. – 1998. – 28. – P. 15-33.
7. Jang R. J.-S., Sun C.-T., Mizutani E. Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence. – Upper Saddle River: Prentice Hall, 1997. – 640 p.
8. Osowski S. Sieci neuronowe do przetwarzania informacji. – Warszawa: Oficyna Wydawnicza Politechniki Warszawskiej, 2006. – 422 s.
9. Ivakhnenko A. G. Heuristic self-organization in problems of engineering cybernetics // Automatica. – 1970. – 6. – №2. – P. 207-219.
10. Ivakhnenko A. G. Polynomial theory of complex systems // IEEE Trans. on Systems, Man, and Cybernetics. – 1971. – 1. – №4. – P. 364-378.
11. Ивахненко А. Г. Самообучающиеся системы распознавания и автоматического управления. – Киев: Техніка, 1969. – 392 с.
12. Ивахненко А. Г. Долгосрочное прогнозирование и управление сложными системами. – Киев: Техніка, 1975. – 311 с.
13. Ивахненко А. Г., Степашко В. С. Помехоустойчивость моделирования. – Киев: Наукова думка, 1985. – 216 с.
14. Ivakhnenko A. G., Ivakhnenko G. A., Mueller J. A. Self-organization of the neural networks with active neurons // Pattern Recognition and Image Analysis. – 1994. – 4. – №2. – P. 177-188.
15. Ivakhnenko G. A. Self-organization of neuronet with active neurons for effects of nuclear test explosions forecasting // System Analysis Modeling Simulation. – 1995. – 20. – P. 107-116.
16. Ivakhnenko A. G., Wuensch D., Ivakhnenko G. A. Inductive sorting-out GMDH algorithms with polynomial complexity for active neurons of neural networks // Neural Networks. – 1999. – 2. – P. 1169-1173.
17. Pham D. T., Liu X. Neural Networks for Identification, Prediction and Control. – London: Springer-Verlag, 1995. – 238 p.
18. Kondo T. Identification of radial basis function networks by using revised GMDH-type neural networks with a feedback loop // Proc. of the SICE Annual Conference. – Tokyo, Japan, 2002. – P. 2882-2887.
19. Ohtani T. Automatic variable selection in RBF network and its application to neurofuzzy GMDH // Proc. Fourth Int. Conf. on Knowledge-Based Intelligent Engineering Systems and Allied Technologies. – 2000. – V.2. – P. 840-843.
20. Bodyanskiy Ye., Teslenko N., Grimm P. Hybrid evolving neural network using kernel activation functions // Proc. 17th Zittau East-West Fuzzy Colloquium. – Zittau/Goerlitz: HS, 2010. – P. 39-46.

21.Bodyanskiy Ye., Vynokurova O., Pliss I. Hybrid GMDH-neural network of computational intelligence // Proc. 3rd Int. Workshop on Inductive Modeling. – Krynica, Poland, 2009. – P. 100-107.

22.Zaychenko Yu. The fuzzy Group Method of Data Handling and its application for economical processes forecasting // Scientific Inquiry. – 2006. – 7. – №1. – P. 83-96.

23.Зайченко Ю. П. Нечеткие модели и методы в интеллектуальных системах. – К.: Издательский дом «Слово», 2008. – 344 с.

24.Ohtani T., Ichihashi H., Miyoshi T., Nagasaka K., Kanaumi Y. Structural learning of neurofuzzy GMDH with Minkowski norm // Proc. 1998 Second Int. Conf. on Knowledge-Based Intelligent Electronic Systems. – 1998. – V.2. – P. 100-107.

25.Bodyanskiy Ye., Zaychenko Yu., Pavlikovskaya E., Samarina M., Viktorov Ye. The neo-fuzzy neural network structure optimization using the GMDH for the solving forecasting and classification problems // Proc. Int. Workshop on Inductive Modeling. – Krynica, Poland, 2009. – P. 77-89.

26.Бодянський Є.В., Винокурова О.А. Адаптивний вейвлон як вузол штучних МГУА-нейронних мереж // Моделювання та керування станом еколого-економічних систем регіонів. – Київ: MHHLS ITC, 2008. – 4. – С. 19-29.

27.Bodyanskiy Ye., Vynokurova O., Teslenko N. Cascade GMDH-wavelet-neuro-fuzzy network // Proc. 4th Int. Workshop on Inductive Modeling «IWIM 2011». – Kyiv, Ukraine, 2011. – P. 22-30.

28.Bodyanskiy Ye., Vynokurova O., Dolotov A., Kharchenko O. Wavelet-neuro-fuzzy network structure optimization using GMDH for the solving forecasting tasks // Proc. 4th Int. Conf. on Inductive Modelling ICIM 2013. – Kyiv, 2013. – P. 61-67.

29.Bodyanskiy Ye. V., Vynokurova O. A., Dolotov A. I. Self-learning cascade spiking neural network for fuzzy clustering based on Group Method of Data Handling // J. of Automation and Information Sciences. – 2013. – 45. – №3. – P. 23-33.

30.Wang L.-X., Mendel J. M. Fuzzy basis functions, universal approximation, and orthogonal least-squares learning // IEEE Trans. on Neural Networks. – 1992. – 3. – №5. – P. 807-814.

31.Wang L.-X. Adaptive Fuzzy Systems and Control. Design and Statistical Analysis. – Upper Saddle River: Prentice Hall, 1994. – 256 p.

32.Narendra K. S., Parthasarathy K. Identification and control of dynamical systems using neural networks // IEEE Trans. on Neural Networks. – 1990. – 1. – P. 4-26.