Hocine TILIOUINE
GDAŃSK UNIVERSITY OF TECHNOLOGY, FACULTY OF ELECTRICAL AND CONTROL ENGINEERING

Comparative study of learning methods for artificial neural network

Ph.D. Hocine TILIOUINE

He was born in Dellys, Algeria. He received the electrical engineering degree in 1977 from Ecole Nationale Polytechnique d'Alger and the PhD degree in 1986 from Gdańsk University of Technology. He is working at the Faculty of Theoretical Electrotechnics since 1995. His current research interests include power system modeling, fuzzy control, neural control and evolutionary algorithms.

e-mail: h.tiliouine@ely.pg.gda.pl

Abstract

The paper presents a comparative study of various learning methods for artificial neural network. The methods are: the backpropagation BP, the recursive least squares RLS, the Zangwill's method ZGW and the method based on evolutionary algorithm EA. The study consists of evaluating the learning effectiveness of these methods and selecting the most efficient one to be used in the designing of an adaptive neural voltage controller for a synchronous generator.

Keywords: learning methods, artificial neural network (ANN), neural voltage controller.

Badania porównawcze metod uczenia sieci neuronowej

Streszczenie

W artykule przedstawiono wyniki badań porównawczych metod uczenia sieci neuronowych takich jak: metoda propagacji wstecznej błędów, rekurencyjna metoda najmniejszych kwadratów, metoda Zangwill'a, metoda algorytmów ewolucyjnych. Celem tych badań jest dobieranie najefektywniejszej metody uczenia do projektowania adaptacyjnego neuronowego regulatora napięcia generatora synchronicznego.

Słowa kluczowe: metody uczenia, sieć neuronowa, neuronowy regulator napięcia.

1. Introduction

The intensive development of neural network applications began after introducing so called backpropagation learning method [18, 19, 20], in the middle of the 80s. The method has so far been generally used in multilayer neural network learning, even though it has several drawbacks. Among the biggest con of that method is its low effectiveness-slow and local convergence. That disadvantage is especially significant with regard to the use of neural networks in the nonlinear control plants, where most frequently the calculations connected with neural network learning are made in the real time. Therefore intensive studies are conducted aiming at:

- the improvement of the backpropagation method;
- developing and testing new methods;
- using simultaneously various methods, so called hybrid methods.

The problem of the feedforward neural network learning with a supervisor comes down to optimization approach, in which independent variables are the weights of particular neurons, and the performance function is the learning error. Regarding the big multi-dimension of such a task and the complex dependence of the learning error on the weights of the neurons (numerous local minimum), solving it demands a high computing time.

The paper presents the results of comparative study in order to select the more efficient learning method to be used in the designing of a neural voltage controller for synchronous generator. In these studies there has been analyzed the effectiveness of the following learning methods:

- backpropagation (BP);
- recursive least square (RLS);
- evolutionary algorithm (EA);
- Zangwill's method as a non-gradient optimization method (ZGW).

All the above mentioned methods were tested with reference to the feedforward artificial neural network (ANN) shown in fig. 1.

It should be noticed, that gaining objective comparative results is very difficult. Each of the methods is characterized by certain parameters influencing its effectiveness. However, the selection of these parameters in the conducted investigations has been made only on the basis of a small number of tests.

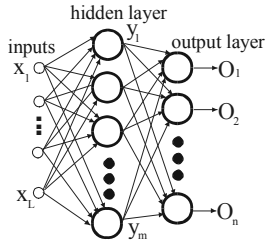

Fig. 1. Block diagram of two-layer feedforward ANN
Rys. 1. Schemat blokowy dwuwarstwowej sieci neuronowej

2. Learning methods

The objective of the learning process is to find an optimal set of weights to minimize the error between the teaching signal (desired output) and the actual response of the neural network. The learning problem can usually be represented as a function minimization problem.

2.1. Backpropagation method

This method is most frequently used in the studies concerning multilayer feed-forward neural networks learning with a supervisor [18, 19].

The minimized function is the following sum of square errors

$$E = \frac{1}{2}\sum_{k=1}^{p}\sum_{j=1}^{n}[d_j(k) - o_j(k)]^2 \tag{1}$$

where

$$o_j = \phi\left(\sum_{k=1}^{m} w_{jk}^{(2)} z_k - w_{j0}^{(2)}\right) \tag{2}$$

and: p- current learning pair number; n- number of neurons in the output layer; d_j- desired output of j-th neuron in the output layer; o_j- current output of j-th neuron in the output layer; w_{j0} - learnable bias weight of j-th neuron in the output layer; Φ- neuron activation function.

In the backpropagation method the minimization of the expression (1) towards the neural network weights is made on the

basis of the descent gradient method. The correction of the neuron weights is computed as [18, 19]:

$$\Delta w_{ij} = \eta \frac{\partial E}{\partial w_{ij}}, \tag{3}$$

where η is the learning rate, an important parameter of the method.

2.2. Recursive least squares method

The RLS method was derived from the signals processing theory [2, 19]. According to the data in the literature, the method is more effective than the classical backpropagation algorithm. In that method, the current learning error is defined as [19]:

$$E^{(p)} = \sum_{k=1}^{p} \lambda^{p-k} \sum_{j=1}^{n} [d_j(k) - \phi(y^T(k) w_j(p))]^2, \tag{4}$$

where: λ - forgetting factor selected from the range [0, 1]. Its role consists of reducing the influence of previous errors on the value of the expression (4); w_j- weight vector of the j-th neuron of the output layer; p- current learning pair number.

In this method the weights are updated according to the following algorithm:

1. After running the neural network the errors of the output signals are calculated:
- for the output layer

$$\varepsilon_i^{(2)}(p) = d_i(p) - o_i(p) \quad \text{for i=1,2,...,n.} \tag{5}$$

- for the hidden layer

$$\varepsilon_i^{(1)}(p) = \sum_{j=1}^{n} \phi'(s_j^{(2)}(p)) w_{ij}^{(2)}(p) \varepsilon_j^{(2)}(p) \ , \ \text{i=1,2,...,m.} \tag{6}$$

where the linear output of the last layer neurons are:

$$s_j^{(2)}(p) = \sum_{i=0}^{m} w_{ji}^{(2)} o_i(p) \ , \ \text{i=1,2,...,n.} \tag{7}$$

2. calculation of matrix $g^{(1)}[(L+1)\times m]$ and matrix $g^{(2)}[(m+1)\times m]$:

$$g_i^{(2)}(p) = \frac{\phi'(s_i^{(2)}(p)) P_i^{(2)}(p-1) o(p)}{\lambda + \phi'^2(s_i^{(2)}(p)) o^T(p) P_i^{(2)}(p-1) o(p)} \ , \ \text{i=1,2,...,n.} \tag{8}$$

$$g_i^{(1)}(p) = \frac{\phi'(s_i^{(1)}(p)) P_i^{(1)}(p-1) o(p)}{\lambda + \phi'^2(s_i^{(1)}(p)) x^T(p) P_i^{(1)}(p-1) x(p)} \ , \ \text{i=1,2,...,m.} \tag{9}$$

where linear outputs of the hidden layer neuron are:

$$s_i^{(1)} = \sum_{j=0}^{m} w_{ij}^{(2)} y_j(p) \ , \ \text{i=1,2,...,n.} \tag{10}$$

3. The change of each weight is given by the formula:

$$w_i^{(1)}(p) = w_i^{(1)}(p-1) + g_i^{(1)}(p) \varepsilon_i^{(1)}(p) \ , \ \text{i=1,2,...,m.} \tag{11}$$

$$w_i^{(2)}(p) = w_i^{(2)}(p-1) + g_i^{(2)}(p) \varepsilon_i^{(2)}(p) \ , \ \text{i=1,2,...,n.} \tag{12}$$

where:

$$w_i^{(1)} = [w_{i0}^{(1)}, w_{i1}^{(1)}, ..., w_{iL}^{(1)}]^T$$

$$w_i^{(2)} = [w_{i0}^{(2)}, w_{i1}^{(2)}, ..., w_{im}^{(2)}]^T$$

Matrices

$$P_i^{(1)}[(L+1)\times(L+1)] \quad \text{for i=1,2,...,m}$$

and

$$P_i^{(2)}[(m+1)\times(m+1)] \quad \text{for i=1,2,...,n}$$

are updated as follows:

$$P_i^{(1)}(p) = \lambda^{-1}[I^{(1)} - \phi'(s_i^{(1)}(p).g_i^{(1)}(p).x^T(p)] P_i^{(1)}(p-1) \tag{13}$$

$$P_i^{(2)}(p) = \lambda^{-1}[I^{(2)} - \phi'(s_i^{(2)}(p).g_i^{(2)}(p).y^T(p)] P_i^{(2)}(p-1) \tag{14}$$

Initial matrices are selected from identity matrices $I^{(1)}$ and $I^{(2)}$:

$$P_i^{(1)}(0) = \delta^{(1)}.I^{(1)} \tag{15}$$

$$P_i^{(2)}(0) = \delta^{(2)}.I^{(2)} \tag{16}$$

where $\delta^{(1)} >> 0$ and $\delta^{(2)} >> 0$.

Initial values for all weights are set zeros or are generated from given interval.

2.3. Learning with the use of the evolutionary algorithm

A simplified diagram of the used evolutionary algorithm (EA) is given in fig. 2. At the beginning it should be noticed that the classical genetic algorithm with a binary coding, with a big number of weights and required precision, would demand too long chromosomes. Therefore, in the presented algorithm the coding in real numbers, the deterministic tournament selection, the elitist strategy and the following genetic operators were used:

- Averaging crossover [2] - for the following parent's chromosomes

$$\mathbf{x}_1 = [x_{11}, x_{12}, ..., x_{1n}], \tag{17}$$

$$\mathbf{x}_2 = [x_{21}, x_{22}, ..., x_{2n}], \tag{18}$$

the genes descendant chromosomes x'_1 and x'_2 are calculated according to the following relations:

$$x_{1i}' = x_{1i} + \delta x_i, \tag{19}$$

$$x_{2i}' = x_{2i} - \delta x_i, \tag{20}$$

for i=1,2,...,n and where $\delta x_i = \xi_u(x_{2i} - x_{1i})$, ξ_u – pseudo random number from range [0, 1] with unified distribution.

The selection of individuals for crossing phase is made according with tournament method. However, the mutation of particular genes is realized with the probability p_m according to the equation:

$$x_{ij}' = x_{ij} + z\xi_{N(0,1)}, \tag{21}$$

where: x_{ij} – value of the gene before mutation, x_{ij}' – value of the gene after mutation, $\xi_{N(0,1)}$ – pseudo random number with Gaussian distribution, z – coefficient dependent on number of iteration k, calculated from the formula:

$$z = \frac{1}{\log(k+10)} \tag{22}$$

In the application of the evolutionary algorithm the task of the neural network learning is defined as an optimization problem, where the optimization criterion is the sum of square errors in the whole learning cycle. The error is calculated with constant weights

and the genes values in particular chromosome result from the network weights.

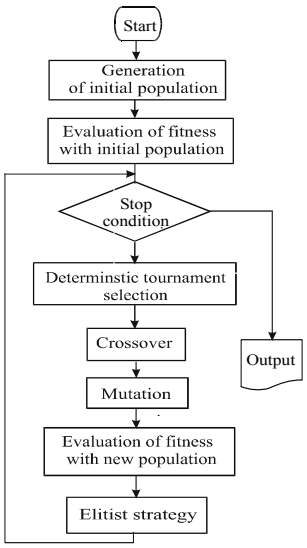

Fig. 2. System flow of EA operation
Rys. 2. Schemat blokowy algorytmu ewolucyjnego

2.4. Zangwill's method

It has been based on methods developed by Zangwill [9] and it is one of the most effective determined, iterative non-gradient methods. It is One of the methods for minimizing a function of several variables based only on the values of the function to be minimized. The method is used when the function is not differentiable or if a calculation of the derivatives involves a large amount of computation. Below the use of the coordinate-wise descent method for minimizing a function $F(x)$ on a set [10, 11]

$$X = \{x = (x^1,...,x^n) : \alpha_i \le x^i \le b_i, i = 1,...,n\},$$

where α_i and b_i are given numbers, $a_i \langle b_i$, is described and the cases where all or some of the $\alpha_i = -\infty$ or $b_i = +\infty$ are not excluded. Let $e_i = (0,...,0,1,0,...,0)$ be the coordinate vector in which the i-th coordinate is equal to 1 and the other coordinates are equal to zero. One specifies an initial approximation $x_0 \in X$, $\alpha_0 > 0$. Assume that the k-th approximation $x_k \in X$ is known and that $\alpha_k > 0$, for some $k \ge 0$. Take $P_k = e_{ik}$, where $i_k = k - n[k/n] + 1$ and $[\alpha]$ is the integer part of α. Then

$$p_0 = e_1,..., p_{n-1} = e_n,$$

$$p_0 = e_1,..., p_{2n-1} = e_n,$$

$$p_{2n} = e_1,...,$$

i.e. one performs a cyclic selection of the coordinate vectors $e_1,...,e_n$. First one checks if the conditions

$$x_k - \alpha_k p_k \in X, F(x_k + \alpha_k p_k) < F(x_k) \qquad (23)$$

are fulfilled. If (23) is fulfilled, one sets $x_{k+1} = x_k + \alpha_k p_k$, $\alpha_{k+1} = \alpha_k$. If on the other hand (23) is not fulfilled, one checks the condition

$$x_k - \alpha_k p_k \in X, F(x_k - \alpha_k p_k) < F(x_k) \qquad (24)$$

If (24) is fulfilled, one sets, $x_{k+1} = x_k + \alpha_k p_k$, $\alpha_{k+1} = \alpha_k$. If conditions (23) and (24) are both not fulfilled, one sets $x_{k+1} = x_k$,

$$\alpha_{k+1} = \begin{cases} \lambda\alpha_k & for \quad i_k = n, \quad x_k = x_{k-n+1}, \\ \alpha_k & for \quad i_k \ne n, \quad or \quad x_k \ne x_{k-n+1} \end{cases} \quad 0 \le k \le n-1 \qquad (25)$$

where λ is the parameter of the method, 0<λ<1. Condition (25) means that if at least one of the conditions (23) and (24) is fulfilled in a single cycle of n iterations involving a selection of all coordinate vectors $e_1,...,e_n$ with step α_k, then the length of the step α_k is not reduced and is retained at least during the following cycle of n iterations; if on the other hand neither (23) nor (24) is ever fulfilled in the subsequent n iterations, the step α_k is reduced.

Let $F(x)$ be convex and continuously differentiable on X, let the set

$$\{x \in X : F(x) \le F(x_0)\}$$

be bounded and let α_0 be a positive number. Then the methods (23)-(25) converge, i.e.

$$\lim_{k \to \infty} F(x) = \inf_{x \in X} F(x),$$

and the sequence $\{x_k\}$ converges to the set of minima for $F(x)$ in X. If $F(x)$ is not differentiable on X, the method need not converge [10, 11].

Applying to the neural network learning, like with the evolutionary algorithm, we choose as optimization criterion (performance function) the total square learning error in the whole learning cycle (calculated with constant weights). However, the independent variables are all the weights present in the network.

3. Simulation results

3.1. Problem formulation

As mentioned in the introduction, the main aim of the discussed studies was comparing the effectiveness of neural network learning algorithms in order to choose the appropriate method to design a neural controller for synchronous generator. The problem concerned the learning process of a neural controller, with an optimized conventional controller as a superviser. The simplified diagram scheme of the studied system is given in fig. 3.

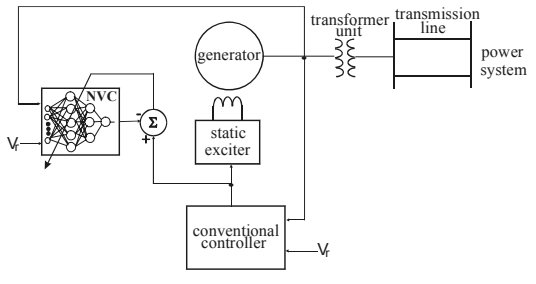

Fig. 3. Schematic diagram of the investigated Configuration
Rys. 3. Schemat blokowy badanego układu

The neural controller works simultaneously with conventional controller, however its output signal is not applied on the plant but its value is compared with the output value of the conventional

controller. The difference between the values of these signals is used to update weights of the neural controller.

The synchronous generator and the conventional controller were modeled as dynamic systems [13]. They are subject to disturbances, namely infinite bus voltage change, as it is shown in fig. 4. The infinite bus voltage is suddenly changed at time t_0, at $T_1+ t_0$ and at $2T_1+ t_0$.

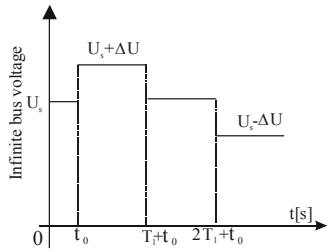

Fig. 4. Bus voltage disturbances in simulation cycle
Rys. 4. Zmiany napięcia systemu elektroenergetycznego podczas jednego cyklu symulacji

As a minimized performance function in the learning process we used the average square error between the output of the neural controller value and the output of the conventional controller value in the simulation cycle.

3.2. Results presentation

Three basic aspects were considered in comparative study of the learning methods:

- recognizing the character of the learning process convergence of the methods and selection of optimal parameters for these methods;
- comparing the effectiveness of learning measured by the number of simulation cycles made;
- comparing the calculating effectiveness defined by the time of the learning process.

Simulation calculations were performed with the following main data:

T = 15s - total simulation time (Fig. 4),
T1 =5s – period of disturbances change,
Δu = 0.05[pu] – value of disturbances change,
h_t = 0.0005s – sampling period for analog controller;
l_t =0.001s – sampling period for neural controller,
l = 5 – number of neural controller inputs,
m = 25 – number of neurons in the hidden layer of the neural network,
n =1 – number of neurons in the output layer.

Main performance characteristics of the methods BP, RLS and EA, in a form of the convergence graphs, are presented in the Fig.5-7 respectively.

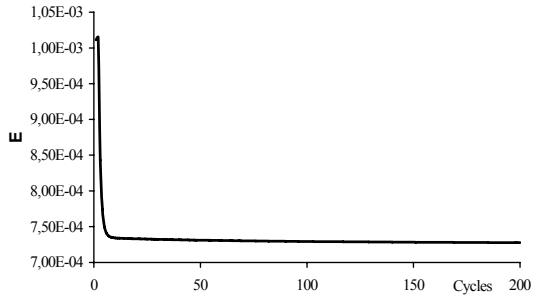

Fig. 5. Convergence graph for RLS method
Rys. 5. Zbieżność metody RLS

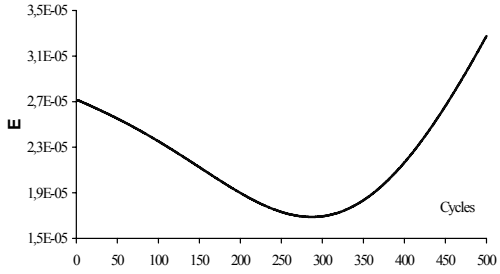

Fig. 6. Convergence graph for BP method
Rys. 6. Zbieżność metody BP

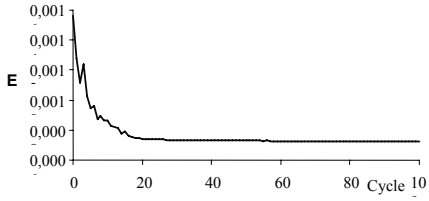

Fig. 7. Convergence graph
Rys. 7. Zbieżność metody EA

In ZGW method it was not possible to obtain intermediate results. The convergence graph for BP method has a minimum at 300 cycles, and then learning error increases. The RLS method gives fast convergence at the beginning of the learning process and subsequently very low convergence. Convergence graph for EA method has similar shape.

Comparative results referring to time of calculation are presented in Table 1. The total learning time in the table resulted from obtaining in the method the testing error of approximately Et=0.001. Testing process realized one simulation cycle without changing the values of the weights. For the BP method testing error (in testing process) was far bigger than error in last learning cycle. The cycle time significantly depends on the learning method.

It is especially big for RLS method. Collective results of the learning processes for all methods comparing to the conventional controller output U_c are presented in Fig. 8.

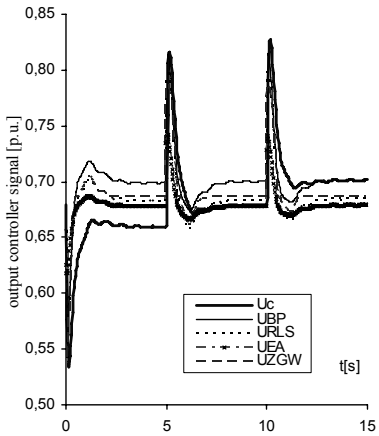

Fig. 8. Comparison results; Uc- output signal of conventional controller; UBP, URLS, UEA, UZGW- output signals of neural controller after BP, RLS, EA, and ZGW training respectively
Rys. 8. Porównanie wyników; Uc- wyściowy signał klasycznego regulatora; UBP, URLS, UEA, UZGW- wyściowe signały neuronowego regulatora po kolejno BP, RLS, EA, i ZGW uczenie

4. Summary and conclusions

In the case of applications of neural networks in control, an essential issue is the learning method's effectiveness. In the studied literature there are no synonymous guidelines concerning the selection of the best learning method for a particular approach [5, 8].

The paper presents the results of studies concerning the comparison of four neural network learning methods based on the example of a neural network voltage controller for a synchronous generator.

Tab. 1. Time of calculation
Tab. 1. Czas obliczeń

Time [s]	Method			
	RLS	BP	EA	ZGW
Time of one learning cycle	1.2	0.02	0.00875	0.0167
Total learning time	2.4	6	6.3	8.354

The responses of the tested systems achieved by using various learning methods can be treated as similar. This results from the fact, that not all possible parameters were studied, which influence the learning effectiveness. Moreover, it was not always possible to find the best values for the studied parameters. It should also be emphasized, that there was a number of representative learning methods. According the available literature, the improvements of RLS method [2] and variable metric optimization methods [6] are promising. Therefore, they should be taken into account in further studies.

To sum up, all studied methods appeared to be effective as they allow for obtaining similar results. However, the most effective is RLS, though it is characterized by high sensitivity to the values of the forgetting factor λ. The effectiveness of the other 3 methods is on the whole comparable.

5. Literatura

[1] Arabas. J.: Lectures on evolutionary algorithms (in Polish). WNT, Warsaw, 2001.

[2] Bilski J. : New algorithms for learning of the feedforward neural networks. Second Conference "Neural Network and their Applications", 1996, Szczyrk-Poland, pp. 39-45.

[3] Goldberg D. E.: Genetic algorithms in search, optimization and machine learning. - Addison-Wesley publishing Compagny,inc., 1989.

[4] Michalewicz Z.: Genetic algorithms + data structure = Evolution programs. - Springer-Verlag, 1992.

[5] Omatu S. , Khalid M. , Yusof R. : Neuro-Control and its Applications. – London: Springer-Verlag, 1995.

[6] Osowski S., Bojarczak P., Stodolski M.: Fast Second Order Learning Algorithm for Feedforward Multilayer Neural Networks and its Applications. - Neural Networks, Vol. 9, No. 9, 1996, pp. 1583-1596.

[7] Guang-Bin Huang; Qin-Yu Zhu; Chee-Kheong Siew: Real-time learning capability of neural networks, IEEE Transactions On Neural Network, vol.17, N0 4, July 2006, pp.863- 878.

[8] Prechelt L.: A Quantitative Study of Experimental Evaluations of Neural Network Learning Algorithms: Current Research Practice. - Neural Networks, Vol. 9, No. 3, 1996, pp. 457-462.

[9] Zangwill W.I.: Minimizing a function without calculating derivatives. – The Computer Journal, Vol. 10, 1967.

[10] Vasil'ev F.P.: Numeriocal methods for solving extremum problems, Moscow 1980.

[11] Karmanov V.G.: Mathematical programming, Moscow 1975.

[12] Yong Xu; Kwok-Wo Wong; Chi-Sing Leung: Generalized RLS approach to the training of neural networks, IEEE Transactions On Neural Network, vol.17, N01,january 2006, pp.19- 34.

[13] Lubośny z., Tiliouine H.: An integrated neural-network based adaptive controller of generator and turbine unit, Międzynarodowa Konferencja SYS'95, Brno, 1995.

[14] Lubośny z., Tiliouine H.: Neuronowy regulator napięcia generatora synchronicznego, Międzynarodowa Konferencja APE'05, Gdańsk-Jurata, Czerwiec 1995.

[15] Farag W.A., Quintana V. H., L-Torres G.: Applications of artificial intelligence techniques in synchronous machine control, International Journal of Engineering Intelligent Systems for Electrical Engineering an Communications, Vol. 6, No 2, June 1998.

[16] Tanomaru J., Omatu S.: Process control by on-line trained neural controllers IEEE Transactions on Industrial Electronics, Vol.39, No 6, December 1992.

[17] Noriega Jose R., Wang Hong: A direct neural-network control for unknown nonlinear systems and its application, IEEE Transactions on Neural Networks, Vol. 9, No 1, January 1998.

[18] Tadeusiewicz Ryszard: Sieci neuronowe, Akademicka Oficyjna wydawnicza RM, Warszawa 1993.

[19] Rutkowski Leszek: Metody i techniki sztucznej inteligencji, PWN, Warszawa 2005.

[20] Principe Jose C., Euliano Neil R., Lefebvre W. Curt: Neural and adaptive systems, Fundamentals through simulations, John Wiley & Sons, INC, 2000.

Artykuł recenzowany