УДК 004.032.26

Ye. Bodyanskiy, Ye. Viktorov, I. Pliss (Kharkiv National University of Radio Electronics)

THE CASCADE NEO-FUZZY NEURAL NETWORK AND ITS LEARNING ALGORITHM

New hybrid system of computational intelligence called the Cascade Neo-Fuzzy Neural Network (CNFNN) is introduced. This architecture has the similar structure with the Cascade-Correlation Learning Architecture (CasCorLA) proposed by S.E. Fahlman and C. Lebiere, but differs from it in type of artificial neurons. The CNFNN consists of neo-fuzzy neurons, which can be adjusted using high-speed linear learning procedures. Proposed architecture is characterized by high learning rate, low size of the learning sample and its operations can be described by the fuzzy linguistic "if-then" rules providing transparency of received results, as compared with the conventional neural networks.

У статті запропоновано нову гібридну систему обчислювального інтелекту — каскадну нео-фаззі нейронну мережу (CNFNN), що має однакову структуру з каскадно-кореляційною нейронною мережею Фальмана–Леб'єра (CasCorLA), однак у якості вузлів вона використовує нео-фаззі нейрони, що навчаються за допомогою швидкодіючих ліninих процедур навчання. Порівняно з існуючими традиційними апроксимуючими нейронними мережами CNFNN характеризується високою швидкістю навчання, потребує навчальну виборку невеликого об'єму, а її функціювання може бути описано нечіткими лінгвістичними "якщо-то"правилами, забезпечуючи прозорість результатів, що отримуються.

Introduction. Nowadays artificial neural networks (ANNs) are widely applied for solving a large class of problems related with the processing information given as time-series or numerical "object-property" tables generated by the non-stationary, chaotic or stochastic systems. The most attractive properties of the ANNs are their approximating possibilities and learning capabilities.

Traditionally by the learning we understand the process of the neural network's synaptic weights adjustment accordingly to selected optimization procedure of the accepted learning criterion [1, 2]. Quality of the received results can be improved not only by adjusting weight coefficients but also by adjusting architecture of the neural network (number of nodes). There are two basic approaches of the neural network architecture adjustment: 1) "constructive approach" [3–5] — starts with simple architecture and gradually adds new nodes during learning; 2) "destructive approach" [6–8] — starts with initially redundant network and simplifies it throughout learning process.

Obviously, constructive approach needs less computational resources and within the bounds of this technique the cascade neural networks (CNNs) [9–11] can be marked out. The most efficient representative of the CNNs is the Cascade-Correlation Learning Architecture (CasCorLA) [9]. This network begins with the simplest architecture which consists of a single neuron. Throughout a learning procedure new neurons are added to the network, producing a multilayer structure. It is important that during each learning epoch only one neuron of the last cascade is adjusted. All pre-existing neurons process information with "frozen" weights. The CasCorLA authors, S.E. Fahlman and C. Lebiere, point out high speed of the learning procedure and good approximation properties of this network. But it should be observed that elementary Rosenblatt perceptrons with hyperbolic tangent activation functions are used in this architecture as nodes. Thus an output signal of each neuron is non-linearly depended from its weight coefficients. Therefore it is necessary to use gradient learning methods such as delta-rule or its modifications, and speed of an operation optimization becomes impossible. In connection with the above it seems to be reasonable to synthesize the cascade architecture based on the elementary nodes with linear dependence of an output signal from the synaptic weights. It allows to increase a speed of synaptic weights adjustment and to reduce minimally required size of training set.

In [12] the ortho-neurons were proposed as such nodes. Also it was shown how simply and effectively an approximation of sufficiently complex function can be performed using this technique. In [13–19] the orthogonal and the cascade orthogonal networks were introduced. These architectures have shown quite good results during simulation modeling, significantly exceeding the conventional cascade neural networks in training speed.

It is well known the main ANN's disadvantage is a non-interpretability of received results, i.e. trained neural network is a "black box", and often their usage is restrained because of this reason. An interpretability and transparency together with the learning capabilities are the main properties of the neuro-fuzzy systems [20], which can be trained using backpropagation and in consequence the time required for weights tuning and the size of a training set are significantly increase.

At this paper an attempt of the new computational intelligence hybrid system synthesis is taken. This system has cascade architecture and uses nodes which perform fuzzy inference and can be trained utilizing some optimization procedure. From our point of view using the neo-fuzzy neurons [21–23] as such elements is the most reasonable. They possess a high approximation possibilities and their computation realization is a quite simple.

1. The Neo-Fuzzy Neuron.

The neo-fuzzy neuron is a nonlinear multi-input single-output system shown on Fig. 1. It realizes the following mapping:

$$\hat{y} = \sum_{i=1}^{n} f_i(x_i), \qquad (1)$$

where x_i is the i-th input ($i = 1, 2, \ldots, n$), $\hat{y}$ is a system output. The structural blocks of the neo-fuzzy neuron are nonlinear synapses NS_i which perform transformation of i-th input signal in the form

$$f_i(x_i) = \sum_{j=1}^{h} w_{ji}\mu_{ji}(x_i)$$

and realizes the fuzzy inference

$$\text{IF } x_i \text{ IS } x_{ji} \text{ THEN THE OUTPUT IS } w_{ji},$$

where x_{ji} is a fuzzy set which membership function is μ_{ji}, w_{ji} is a singleton (synaptic weight) in consequent [22]. As it can be readily seen the nonlinear synapse in fact realizes Takagi-Sugeno fuzzy inference of zero order.

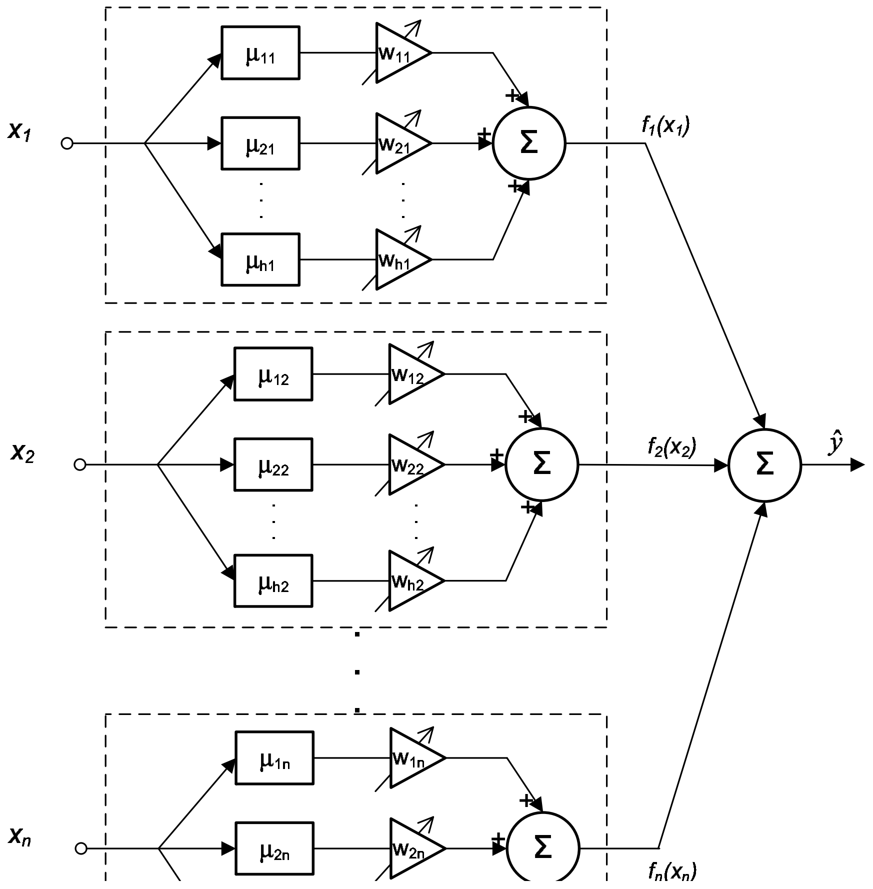


Figure 1. Neo-Fuzzy Neuron

Conventionally the membership functions $\mu_{ji}(x_i)$ in the antecedent are complementary triangular functions as shown on Fig. 2.

For the preliminary normalized input variables x_i (usually $0 \le x_i \le 1$), the membership functions can be expressed in the form:

$$\mu_{ji}(x_i) = \begin{cases} \dfrac{x_i - c_{j-1,i}}{c_{ji} - c_{j-1,i}}, \ x \in [c_{j-1,i}, \ c_{ji}], \\ \dfrac{c_{j+1,i} - x_i}{c_{j+1,i} - c_{ji}}, \ x \in [c_{ji}, \ c_{j+1,i}], \\ 0, \ otherwise, \end{cases}$$

where c_{ji} are arbitrarily selected centers of the corresponding membership functions. Usually they are evenly spaced on the interval $[0, 1]$. This contributes to simplify the fuzzy inference process. That is, an input signal x_i activates only two neighboring membership functions simultaneously and the sum of the grades of these two membership functions equals to unity (Ruspini partitioning), i.e.

$$\mu_{ji}(x_i) + \mu_{j+1,i}(x_i) = 1.$$

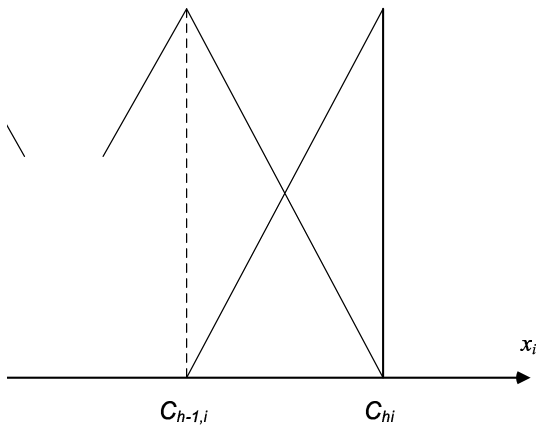


Figure 2. Triangular activation functions

Thus, the fuzzy inference result produced by the Center-of-Gravity defuzzification method can be given in a very simple form:

$$f_i(x_i) = w_{ji}\mu_{ji}(x_i) + w_{j+1,i}\mu_{j+1,i}(x_i).$$

By summing up $f_i(x_i)$, the output $\hat{y}$ of the equation (1) is produced.

When a vector signal $x(k) = (x_1(k), x_2(k), \dots, x_n(k))^T$ (here $k = 1, 2, \dots$ is a discrete time) is fed to the input of the neo-fuzzy neuron, the output of this neuron is determined by both the membership functions $\mu_{ji}(x_i(k))$ and the tunable synaptic weights $w_{ji}(k-1)$, which have been obtained at the previous training epoch:

$$\hat{y}(k) = \sum_{i=1}^{n} f_i(x_i(k)) = \sum_{i=1}^{n}\sum_{j=1}^{h} w_{ji}(k-1)\mu_{ji}(x_i(k)),$$

and thereby the neo-fuzzy neuron contains $h \cdot n$ synaptic weights which should be determined.

The authors of the NFN note [21–23] among its most important advantages, the high rate of learning, computational simplicity, the possibility of finding the global minimum of the learning criterion in real time, and also that it is characterized by the fuzzy linguistic "if-then" rules.

The learning criterion (goal function) is the standard local quadratic error function:

$$E(k) = \frac{1}{2}\left(y(k) - \hat{y}(k)\right)^2 = \frac{1}{2}e(k)^2 = \frac{1}{2}\left(y(k) - \sum_{i=1}^{n}\sum_{j=1}^{h} w_{ji}\mu_{ji}(x_i(k))\right)^2,$$

minimized via the conventional gradient stepwise algorithm, resulting in the following weights update procedure:

$$w_{ji}(k+1) = w_{ji}(k) + \eta e(k+1)\mu_{ji}(x_i(k+1)) =$$

$$= w_{ji}(k) + \eta \left(y(k+1) - \sum_{i=1}^{n} \sum_{j=1}^{h} w_{ji}(k)\mu_{ji}(x_i(k+1)) \right) \mu_{ji}(x_i(k+1)),$$

where $y(k)$ is a target value of the output, η is a scalar learning rate parameter which determines the speed of convergence and chosen empirically.

For the purpose of increasing training speed [24, 25] Kaczmarz-Widrow-Hoff optimal one-step algorithm [26–28] was used in the following form:

$$w(k+1) = w(k) + \frac{y(k+1) - w^T(k)\mu(x(k+1))}{\|\mu(x(k+1))\|^2}\mu(x(k+1)), \tag{2}$$

where

$$\mu(x(k{+}1)) = (\mu_{11}(x_1(k{+}1)), \ldots, \mu_{h1}(x_1(k{+}1)), \ldots, \mu_{h2}(x_2(k{+}1)), \ldots$$
$$\ldots, \mu_{ji}(x_i(k{+}1)), \ldots, \mu_{hn}(x_n(k+1)))^T,$$

$w(k) = (w_{11}(k), \ldots, w_{h1}(k), \ldots, w_{h2}(k), \ldots, w_{ji}(k), \ldots, w_{hn}(k))^T$ — $(h{\cdot}n)\times 1$-vectors, generated by the corresponding variables, and its exponentially weighted modification

$$\begin{cases} w(k+1) = w(k) + r^{-1}(k+1)(y(k+1) - w^T(k)\mu(x(k+1)))\mu(x(k+1)), \\ r(k+1) = \alpha r(k) + \|\mu(x(k+1))\|^2,\ 0 \le \alpha \le 1, \end{cases} \tag{3}$$

which possesses both smoothing and filtering properties.

If we have *a priori* defined data set the training procedure can be performed in a batch mode using the conventional least squares method.

On basis of the neo-fuzzy neurons in [29–34] the two-layer feedforward neuro-fuzzy network was synthesized. It possesses improved approximation capabilities in comparison with the conventional multilayer perceptron. Given ANN utilized a backpropagation for weights adaptation and obviously it results in decreasing rate of a convergence in the hidden layer. This circumstance also was a reason for developing the cascade neo-fuzzy neural network described below.

2. The Cascade Neo-Fuzzy Neural Network Architecture. The CNFNN architecture shown on Fig. 3 and the mapping which it realized have the following form:

– the neo-fuzzy neuron of the first cascade

$$\hat{y}^{[1]} = \sum_{i=1}^{n} \sum_{j=1}^{h} w_{ji}^{[1]} \mu_{ji}(x_i),$$

– the neo-fuzzy neuron of the second cascade

$$\hat{y}^{[2]} = \sum_{i=1}^{n} \sum_{j=1}^{h} w_{ji}^{[2]} \mu_{ji}(x_i) + \sum_{j=1}^{h} w_{j,n+1}^{[2]} \mu_{j,n+1}(\hat{y}^{[1]}),$$

– the neo-fuzzy neuron of the third cascade

$$\hat{y}^{[3]} = \sum_{i=1}^{n} \sum_{j=1}^{h} w_{ji}^{[3]} \mu_{ji}(x_i) + \sum_{j=1}^{h} w_{j,n+1}^{[3]} \mu_{j,n+1}(\hat{y}^{[1]}) + \sum_{j=1}^{h} w_{j,n+2}^{[3]} \mu_{j,n+2}(\hat{y}^{[2]}),$$

– the neo-fuzzy neuron of the m-th cascade

$$\hat{y}^{[m]} = \sum_{i=1}^{n}\sum_{j=1}^{h} w_{ji}^{[m]}\mu_{ji}(x_i) + \sum_{l=n+1}^{n+m-1}\sum_{j=1}^{h} w_{jl}^{[m]}\mu_{jl}(\hat{y}^{[l-n]}). \tag{4}$$

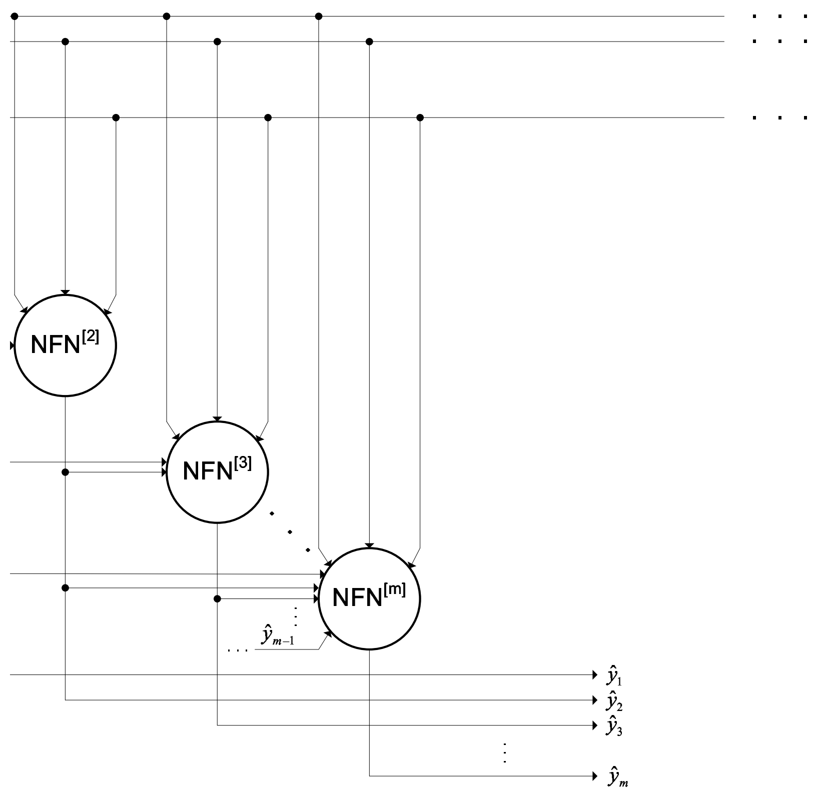


Figure 3. The Cascade Neo-Fuzzy Neural Network Architecture

Thus the cascade neo-fuzzy neural network contains $h \cdot \left(n + \sum_{l=1}^{m-1} l\right)$ adjustable parameters and it is important that all of them are linearly included in the description (4).

Let us define $h(n + m - 1) \times 1$ the membership functions vector of the m-th neo-fuzzy neuron

$$\mu^{[m]} = (\mu_{11}(x_1), \ldots, \mu_{h1}(x_1), \mu_{12}(x_2), \ldots, \mu_{h2}(x_2), \ldots$$

$$\ldots, \mu_{ji}(x_i), \ldots, \mu_{hn}(x_n), \mu_{1,n+1}(\hat{y}^{[1]}), \ldots, \mu_{h,n+1}(\hat{y}^{[1]}), \ldots, \mu_{h,n+m-1}(\hat{y}^{m-1}))^T$$

and the corresponding vector of synaptic weights

$$w^{[m]} = (w_{11}^{[m]}, w_{21}^{[m]}, \ldots, w_{h1}^{[m]}, w_{12}^{[m]}, \ldots, w_{h2}^{[m]}, \ldots$$

$$\ldots, w_{ji}^{[m]}, \ldots, w_{hn}^{[m]}, w_{1,n+1}^{[m]}, \ldots, w_{h,n+1}^{[m]}, \ldots, w_{h,n+m-1}^{[m]})^T,$$

which has the same dimensionality. Then we can represent expression (4) in the vector notation:

$$\hat{y}^{[m]} = w^{[m]T}\mu^{[m]}.$$

3. The Cascade Neo-Fuzzy Neural Network Learning. The cascade neo-fuzzy neural network learning can be performed in both the batch mode and the mode of sequential information processing (an adaptive weights tuning).

Firstly, let us consider situation when the training data set is defined *a priori*, i.e. we have a set of points $x(1), y(1); x(2), y(2); \ldots ; x(k), y(k); \ldots ; x(N), y(N)$. For the neo-fuzzy neuron of the first cascade $NFN^{[1]}$ a set of membership level values $\mu^{[1]}(1), \mu^{[1]}(2), \ldots, \mu^{[1]}(k), \ldots, \mu^{[1]}(N)$ ($hn \times 1$ vectors) is evaluated, where

$$\mu^{[1]}(k) = (\mu_{11}(x_1(k)), \ldots, \mu_{h1}(x_1(k)), \mu_{12}(x_2(k)), \ldots, \mu_{h2}(x_2(k)), \ldots$$
$$\ldots, \mu_{ji}(x_i(k)), \ldots, \mu_{hn}(x_n(k)))^T.$$

Then using direct minimization of the learning criterion

$$E_N^{[1]} = \frac{1}{2}\sum_{k=1}^{N}(e^{[1]}(k))^2 = \frac{1}{2}\sum_{k=1}^{N}(y(k) - \hat{y}^{[1]}(k))^2$$

a vector of the synaptic weights can be evaluated

$$w^{[1]}(N) = \left(\sum_{k=1}^{N}\mu^{[1]}(k)\mu^{[1]T}(k)\right)^{+}\sum_{k=1}^{N}\mu^{[1]}(k)y(k) = P^{[1]}(N)\sum_{k=1}^{N}\mu^{[1]}(k)y(k), \qquad (5)$$

where $(\bullet)^{+}$ denotes the Moore-Penrose pseudoinversion.

In case a data proceeds sequentially more suitable to use a recurrent form of the least squares method instead of (8)

$$\begin{cases} w^{[1]}(k+1) = w^{[1]}(k) + \dfrac{P^{[1]}(k)(y(k+1) - w^{[1]T}(k)\mu^{[1]}(k+1))}{1 + \mu^{[1]T}(k+1)P^{[1]}(k)\mu^{[1]}(k+1)}\mu^{[1]}(k+1), \\ P^{[1]}(k+1) = P^{[1]}(k) - \dfrac{P^{[1]}(k)\mu^{[1]}(k+1)\mu^{[1]T}(k+1)P^{[1]}(k)}{1 + \mu^{[1]T}(k+1)P^{[1]}(k)\mu^{[1]}(k+1)},\ P^{[1]}(0) = \beta I, \end{cases} \qquad (6)$$

where β is a large positive number, I is a unity matrix with corresponding dimensionality.

Using of adaptive algorithms (2) or (3) is also possible and leads to reducing of computational complexity of learning process. In any case utilization of the procedures (2), (3), (8), (6) essentially reduces a learning time in comparison with the gradient algorithms underlying delta-rule and backpropagation.

After the first cascade learning completion, the synaptic weights of the neo-fuzzy neuron $NFN^{[1]}$ become "frozen", all values $\hat{y}^{[1]}(1), \hat{y}^{[1]}(2), \ldots, \hat{y}^{[1]}(k), \ldots, \hat{y}^{[1]}(N)$ are evaluated and the second cascade of the network which consists of a single neo-fuzzy neuron $NFN^{[2]}$ is generated. It has one additional input for the output signal of the first cascade. Then the procedure (8) is again applied for adjusting a vector of the weight coefficients $w^{[2]}$, which has dimensionality $h(n+1) \times 1$.

In on-line mode the neurons are trained sequentially, i.e. on basis of the input signal $x(k)$ the synaptic weights $w^{[1]}(k)$ are estimated and the vector of outputs $\hat{y}^{[1]}(k)$ is obtained, then using vector of the second cascade inputs $(x^T(k), \hat{y}^{[1]}(k))$ the weights $w^{[2]}(k)$ and the outputs $\hat{y}^{[2]}(k)$ are calculated. For this purpose algorithms (2), (3) and (6) can be used equally well.

The neural network growing process (an increasing quantity of cascades) continues until we obtain required precision of the solved problem's solution, and for the adjusting weight coefficients of the last m-th cascade the following expressions are used:

$$w^{[m]}(N) = \left(\sum_{k=1}^{N} \mu^{[m]}(k)\mu^{[m]T}(k)\right)^{+} \sum_{k=1}^{N} \mu^{[m]}(k)y(k) = P^{[m]}(N)\sum_{k=1}^{N} \mu^{[m]}(k)y(k),$$

in a batch mode or

$$\begin{cases} w^{[m]}(k+1) = w^{[m]}(k) + \dfrac{P^{[m]}(k)(y(k+1) - w^{[m]T}(k)\mu^{[m]}(k+1))}{1+\mu^{[m]T}(k+1)P^{[m]}(k)\mu^{[m]}(k+1)}\mu^{[m]}(k+1), \\ P^{[m]}(k+1) = P^{[m]}(k) - \dfrac{P^{[m]}(k)\mu^{[m]}(k+1)\mu^{[m]T}(k+1)P^{[m]}(k)}{1+\mu^{[m]T}(k+1)P^{[m]}(k)\mu^{[m]}(k+1)}, \ P^{[m]}(0) = \beta I, \end{cases}$$

or

$$\begin{cases} w^{[m]}(k+1) = w^{[m]}(k) + (r^{[m]}(k+1))^{-1}(y(k+1) - w^{[m]T}(k)\mu^{[m]}(k+1))\mu^{[m]}(k+1), \\ r^{[m]}(k+1) = \alpha r^{[m]}(k) + \|\mu^{[m]}(k+1)\|^2, \ 0 \le \alpha \le 1, \end{cases}$$

in a sequential mode.

Thus, the proposed CNFNN significantly excels the Cascade-Correlation Architecture in learning speed and can be trained in both the batch mode and the sequential (adaptive) mode. A linguistic interpretation of received results considerably extends functional facilities of the cascade neo-fuzzy neural network.

4. Simulation Results. In order to confirm the performance of the proposed architecture the prediction of time-series is examined. We applied the Cascade Neo-Fuzzy Neural Network for the forecasting of a chaotic process defined by the Mackey-Glass equation [35]:

$$y'(t) = \frac{0.2(t-\tau)}{1+y^{10}(t-\tau)} - 0.1y(t). \tag{7}$$

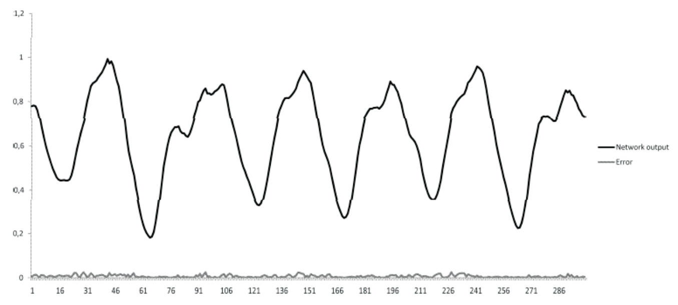
The signal defined by (7) was quantized with step 0.1. We took a fragment containing 500 points for the training set. Our goal was to predict the time-series value on six steps forward using its values at the time steps k, $(k-6)$, $(k-12)$, and $(k-18)$. The testing set contained 9500 points, i.e. the time-series values at the time steps from 501 to 1000.

For estimation of received result we used normalized mean square error:

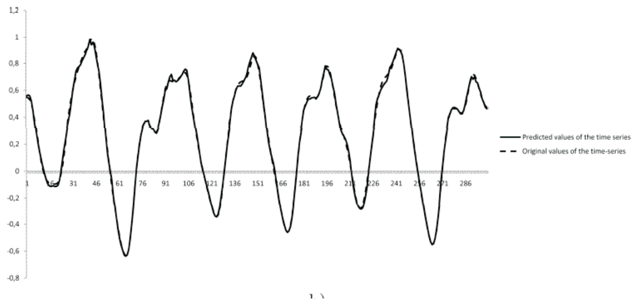
$$\text{NRMSE}\ (k, N) = \frac{\sum_{q=1}^{N} e^2(k+q)}{N\sigma},$$

where σ is a mean square deviation of the predicted process on the training set.

During simulation modelling the CNFNN with 5 cascades was used. First cascade contained 4 non-linear synapses for each input value of the time-series and each non-linear synapse contained 10 activation functions (i.e. membership functions). The input signal was recoded on interval $[0,1]$. Obtained experimental results of the the Mackey-Glass time-series prediction and its error are shown on Fig. 4 a) and b). After the training of the CNFNN was complete an error on the testing set was $4{\cdot}10^{-4}$.



a)



b)

Figure 4. The CNFNN simulation results: a) network output and error; b) predicted and original time-series

Conclusion. The Cascade Orthogonal Neural Network is proposed. It differs from the known cascade networks in increased speed of operation, real-time processing possibility and transparency due to linguistic interpretability of received results. Theoretical justification and experiment results confirm the efficiency of developed approach to cascade neo-fuzzy systems synthesis.

1. *Cichocki A., Unbehanen R.* Neural Networks for Optimization and Signal Processing. – Stuttgart: Teubner, 1993.
2. *Haykin S.* Neural Networks. A Comprehensive Foundation. – Upper Saddle River, N.J.: Prentice Hall, 1999. – 864 p.
3. *Platt J.* A resource allocating network for function interpolation // Neural Computation. – 1991. – 3. – P. 213–225.
4. *Nag A., Ghosh J.* Flexible resource allocating network for noisy data // Proc. SPIE Conf. on Applications and Science of Computational Intelligence. – 1998. – P. 551–559.
5. *Yingwei L., Sundararajan N., Saratchandran P.* Performance evaluation of a sequential minimal radial basis function (RBF) neural network learning algorithm // IEEE Trans. on Neural Networks. – 1998. – 9. – P. 308–318.
6. *Cun Y. L., Denker J. S., Solla S. A.* Optimal Brain Damage // Advances in Neural Information Processing Systems. – 1990. – 2. – P. 598–605.
7. *Hassibi B. Stork D. G.* Second-order derivatives for network pruning: Optimal brain surgeon // Advances in Neural Information Processing Systems. Ed. Hanson et al. – San Mateo, CA: Morgan Kaufman. – 1993. – P. 164–171.
8. *Prechelt L.* Connection pruning with static and adaptive pruning schedules // Neurocomputing. – 1997. – 16. – P. 49–61.
9. *Fahlman S. E., Lebiere C.* The cascade-correlation learning architecture // Advances in Neural Information Processing Systems. Ed. D. S. Touretzky. – San Mateo, CA: Morgan Kaufman. – 1990. – P. 524–532.
10. *Schalkoff R. J.* Artificial Neural Networks. – N.Y.: The McGraw-Hill Comp., 1997.
11. *Аведьян Э. Д., Баркан Г. В., Левин И. К.* Каскадные нейронные сети // Автоматика и

телемеханика. – 1999. – №3. – С. 38–55.

12. *Бодянский Е. В., Викторов Е. А., Слипченко А. Н.* Ортосинапс, ортонейроны и нейропредиктор на их основе // Системи обробки інформації. – 2007. – Випуск 4(62). – С. 139–143.
13. *Bodyanskiy Ye., Kolodyazhniy V., Slipchenko O.* Artificial neural network with orthogonal activation functions for dynamic system identification // Synergies between Information Processing and Automation. Ed. O. Sawodny and P. Scharff. –Aachen: Shaker Verlag. – 2004. – P. 24–30.
14. *Bodyanskiy Ye., Kolodyazhniy V., Slipchenko O.* Structural and synaptic adaptation in the artificial neural networks with orthogonal activation functions // Sci. Proc. of Riga Technical University. Comp. Sci., Inf. Technology and Management Sci. – 2004. – 20. – P. 69–76.
15. *Bodyanskiy Ye., Pliss I., Slipchenko O.* Growing neural networks based on orthogonal activation functions // Proc. XII-th Int. Conf. "Knowledge–Dialog–Solution". Varna. – 2006. – P. 84–89.
16. *Bodyanskiy Ye., Slipchenko O.* Ontogenic neural networks using orthogonal activation functions // Prace naukowe Akademii Ekonomicznej we Wroclawiu. – 2006. – 21. – P. 13-20.
17. *Bodyanskiy Ye., Pliss I., Slipchenko O.* Growing neural network using nonconventional activation functions // Int. J. Information Theories & Applications. – 2007. – 14. – P. 275–281.
18. *Bodyanskiy Ye., Dolotov A., Pliss I., Viktorov Ye.* The cascade orthogonal neural network // Advanced Research in Artificial Intelligence. Bulgaria, Sofia: Institute of Informational Theories and Applications FOI ITHEA. – 2008. – 2. – P. 13–20.
19. *Viktorov Ye., Bodyanskiy Ye., Dolotov A.* Solving approximation and forecasting problems using double ortho-neuron // Комп'ютерні науки та інформаційні технології. – 2008. – №598. – С. 70–77.
20. *Jang Jr. S. R., Sun C. T., Mizutani E.* Neuro-Fuzzy and Soft Computing. A Computational Approach to Learning and Machine Intelligence. – N.J.: Prentice Hall, 1997. – 614 p.
21. *Yamakawa T., Uchino E., Miki T., Kusanagi H.* A neo fuzzy neuron and its applications to system identification and prediction of the system behavior // Proc. 2-nd Int.Conf. on Fuzzy Logic and Neural Networks "LIZUKA–92". Lizuka, Japan. – 1992. – P. 477–483.
22. *Uchino E., Yamakawa T.* Soft computing based signal prediction, restoration and filtering // Intelligent Hybrid Systems: Fuzzy Logic, Neural Networks and Genetic Algorithms. Ed. Da Ruan. – Boston: Kluwer Academic Publisher. – 1997. – P. 331–349.
23. *Miki T., Yamakawa T.* Analog implementation of neo-fuzzy neuron and its on-board learning // Computational Intelligence and Applications. Ed. N. E. Mastorakis. – Piraeus: WSES Press. – 1999. – P. 144–149.
24. *Bodyanskiy Ye., Kokshenev I., Kolodyazhniy V.* An adaptive learning algorithm for a neo-fuzzy neuron // Proc. 3rd Int. Conf. of European Union Soc. for Fuzzy Logic and Technology (EUSFLAT 2003). – Zittau, Germany. – 2003. – P. 375–379.
25. *Kolodyazhniy V., Bodyanskiy Ye., Otto P.* Universal approximator employing neo-fuzzy neurons // In "Computational Intelligence: Theory and Applications." Ed. by B. Reusch. – Berlin-Heidelberg: Springer. – 2005. – P. 631–640.
26. *Kaczmarz S.* Angenaeherte Ausloesung von Systemen Linearer Gleichungen // Bull. Int. Acad. Polon. Sci. – 1937. – Let. A. – S. 355–357.
27. *Kaczmarz S.* Approximate solution of systems of linear equations // Int. J. Control. – 1993. – 53. – P. 1269–1271.
28. *Widrow B., Hoff Jr. M. E.* Adaptive switching circuits // 1960 URE WESCON Convention Record. – N.Y.: IRE. – 1960. – Part 4. – P. 96–104.
29. *Kolodyazhniy V., Bodyanskiy Ye.* Fuzzy neural network with Kolmogorov's structure // Proc. East West Fuzzy Coll. – Zittau/Goerlity: HS. – 2004. – P. 139–146.
30. *Kolodyazhniy V., Bodyanskiy Ye.* Fuzzy Kolmogorov's network // In "Lecture Notes in Computer Science." –Heidlberg: Springer Verlag. – 2004. – V.3214. – P. 764–771.
31. *Bodyanskiy Ye., Gorshkov Ye., Kolodyazhniy V.* Neuro-fuzzy Kologorov's network with a hybid learning algorithm // Proc. XI-th Int. Conf. "Knowledge-Dialog-Solution." – Varna, Bulgaria, – 2005. – V.2. – P. 622–627.
32. *Bodyanskiy Ye., Kolodyazhniy V., Otto P.* Neuro-fuzzy Kolmogorov's network for time-series prediction and pattern classification // In "Lecture Notes in Artificial Intelligence." – Heidelberg: Springer Verlag. – 2005. – V. 3698. – P. 191–202.

33. *Bodyanskiy Ye., Gorshkov Ye., Kolodyazhniy V., Poedintseva V.* Neuro-fuzzy Kolmogorov's network // In "Lecture Notes in Computer Science." – Berlin-Heidelberg: Springer Verlag. – 2005. – V.3697. – P. 1–6.
34. *Kolodyazhniy V., Bodyanskiy Ye., Poedintseva V., Stephan A.* Neuro-fuzzy Kolmogorov's network with a modified perceptron learning rule for classification problems // In "Computational Intelligence: Theory and Applications." Ed. by B. Reusch. – Berlin-Heidelberg: Springer-Verlag. – 2006. – P. 41–49.
35. *Mackey M. C., Glass L.* Oscillation and chaos in physiological control systems // Science. – 1977. – 197. – P. 238–289.

Одержано 24.10.2008