RESERVOIR FORECASTING NEURO-FUZZY NETWORK AND ITS LEARNING

Yevgeniy Bodyanskiy, Oleksii Tyshchenko, Iryna Pliss

***Abstract**: The idea of Reservoir Computing has become so popular recently due to its computational efficiency and the fact that just only a linear output layer must be taught. In this paper a reservoir is proposed to be built using neo-fuzzy neurons which means that Reservoir Computing and paradigms and fuzzy logic are combined.*

***Keywords**: reservoir computing, hybrid systems, neo-fuzzy neuron, online learning procedure, prediction.*

***ACM Classification Keywords**: I.2.6 Learning – Connectionism and neural nets.*

Introduction

Training a recurrent neural network is not an easy task to do, it has been offered lately to construct a custom recurrent topology to train a single linear output layer. Nowadays it can be easily performed with the help of Reservoir Computing [Jaeger, 2001a; Jaeger, 2001b]. Reservoirs are randomly created, and the exact weight distribution and sparsity have very limited influence on the reservoir's performance.

Reservoir Computing usually consists of Echo State Networks [Alexandre, 2009], Liquid State Machines and the Backpropagation Decorrelation learning rule. The most important part of tuning weights in traditional learning procedures of recurrent neural networks is the output layer weight tuning.

Reservoir neural networks were designed to have the same computational power as traditional recurrent neural networks except the fact that there's no need to train internal weights.

The reservoir is a recurrent neural network which has n input units, h internal units of the hidden layer and m output units. Input elements at a time point k form a vector $\mathrm{x}(\mathrm{k}) = \left(x_1(\mathrm{k})\ldots x_n(\mathrm{k})\right)^T$, internal units form a vector $\tilde{\mathrm{s}}(\mathrm{k}) = \left(\tilde{\mathrm{s}}_1(k)\ldots\tilde{\mathrm{s}}_h(k)\right)^T$ and finally output units form a vector $y(\mathrm{k}) = \left(y_1(\mathrm{k})\ldots y_m(\mathrm{k})\right)^T$.

Real-valued connection weights are collected in a $(h\times n)$ weight matrix W_{in} for the input weights, in an $(h\times h)$ matrix W_{res} for the internal connections, in an $m\times(h+n+m)$- matrix W_{out} for the connections to the output units, and in a $(n\times m)$-matrix W_{back} for the connections that feed back from the output to the internal units. The weight matrix W_{in} is created randomly. This matrix can be either full or sparsed. The weight matrix W_{res} is usually created according to Gaussian distribution. It can be easily explained by the fact that a network creates a reservoir full of different nonlinear current and preceding values (a so-called "echo"). Connections directly from the input to the output units and connections between output units are allowed. A reservoir should have a suitable perceptibility. The simplest way to make it is to tune a spectral radius (the biggest eigenvalue) of weight matrix W_{res}.

The activation of internal units is updated according to expression

$$\tilde{\mathrm{s}}(k+1) = \mathrm{f}\left(\mathrm{W}_{\mathrm{in}}x(\mathrm{k}+1)+\mathrm{W}_{res}\tilde{\mathrm{s}}(\mathrm{k})+\mathrm{W}_{\mathrm{back}}\mathrm{y}(\mathrm{k})\right), \quad (1)$$

where $\tilde{s}(k+1)$ is a vector of reservoir states at a time point k, where $f = (f_1 \dots f_P)$ – the internal unit's output functions (typically sigmoid functions).

The output is computed according to equation

$$y(k+1) = f_{out}\left(W_{out}\begin{bmatrix}\tilde{s}(k+1)\\ x(k+1)\end{bmatrix}\right), \quad (2)$$

where $f_{out} = (f_{out1} \dots f_{outT})$ – the output unit's output functions.

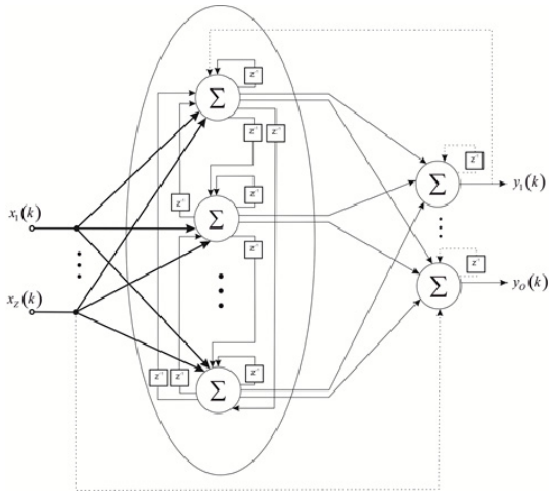

Figure 1. Representation of a reservoir network. Large circles are the neurons and small ones represent the input data. The hidden layer is the reservoir. Dashed connections are optional

In the Echo-State Networks literature, the reservoirs are rescaled using measures based on stability bounds. Several of these measures have been presented in [Jaeger, 2001b]. Jaeger has proposed the spectral radius to be slightly lower than one (because the reservoir is then guaranteed to have the echo state property).

The most interesting thing about reservoirs is that all weight matrices to the reservoir are initialized randomly, while all connections to the output are trained. When using the system after training with the "reservoir-output" connections, the computed output by is fed back into the reservoir.

Although many different neuron types have already been used in reservoirs but there is no yet clear understanding which node types are optimal for given applications. The necessary and sufficient conditions for a recurrent network to have echo states are based on the spectral radius and the largest singular value of the connection matrix. The echo state property means that the current state of the network is uniquely determined by the network input up to now. The network is thus state forgetting.

Network Architecture

In this paper a new architecture of a forecasting neural network is proposed which is built with the help of neo-fuzzy neurons (NFN) [Horio, 2001] and a layer of delay elements. The proposed network was called a reservoir forecasting neuro-fuzzy network in [Bodyanskiy, 2009; Bodyanskiy, 2010]. The scheme of the proposed network is on fig. 2.

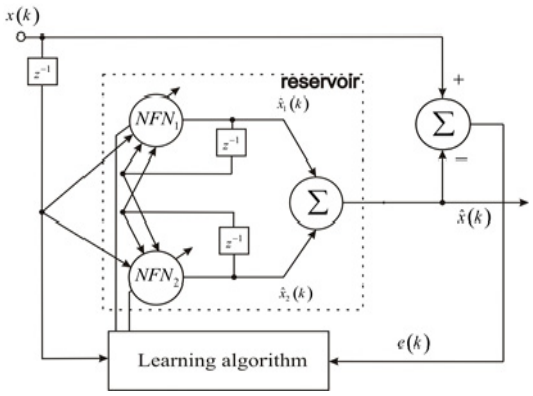

Figure 2. The architecture of the reservoir forecasting neuro-fuzzy network

$x(k)$ is an input value of the network, $\hat{x}(k)$ is a network output, $k = 1, 2, \ldots, N, \ldots$ - current discrete time.

Neo-fuzzy neurons were proposed by T. Yamakawa and co-authors [Yamakawa, 1992]. These constructions are neuronal models with nonlinear synapses. The output of the nonlinear synapse neuron is obtained by sum of the outputs of the synapses represented by nonlinear functions. They can approximate a nonlinear input-output relationship by one neuron, and there is no local minimum problem in learning [Uchino, 1997; Miki, 1999]. Fig. 3 shows a structure of the conventional NFN.

An input signal $x(k)$ is fed into the NFN layer. It should be mentioned that this construction has back connections which come through the layer of delay elements from NFN outputs back to their inputs [Tyshchenko, 2012].

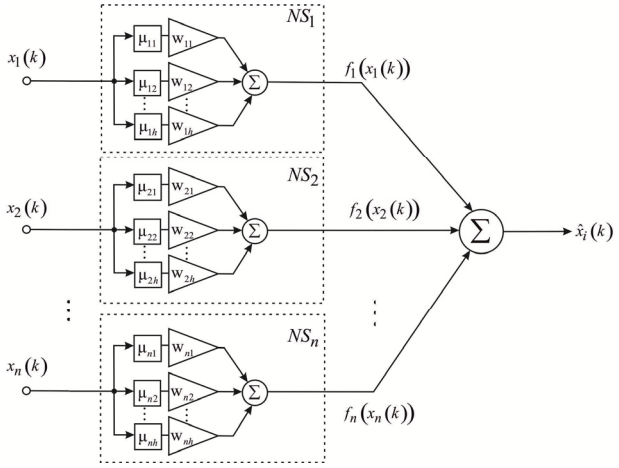

Figure 3. The structure of the conventional neo-fuzzy neuron

A network output is calculated in the form

$$\hat{x}(k) = \hat{x}_1(k) + \hat{x}_2(k) = \sum_{l=1}^{2} f^{[l]}(x(k-1)) + f^{[l]}(\hat{x}_1(k-2)) + f^{[l]}(\hat{x}_2(k-2)) =$$
$$= \sum_{l=1}^{2}\sum_{i=1}^{3}\sum_{j=1}^{h} \mu_{ij}^{[l]}\left(x(k-1)\right) w_{ij}^{[l]}(k) + \mu_{ij}^{[l]}\left(\hat{x}_1(k-2)\right) w_{ij}^{[l]}(k) + \mu_{ij}^{[l]}\left(\hat{x}_2(k-2)\right) w_{ij}^{[l]}(k), \quad (3)$$

where $\hat{x}_1, \hat{x}_2$ – outputs of NFN_1 and NFN_2 correspondingly;

$\mu_{ij}^{[l]}$ – membership functions of NFN_l;

$w_{ij}^{[l]}$ – synaptic weights of NFN_l;

h - a number of membership functions in a nonlinear synapse.

The architecture of the special NFN used in our case is presented on fig. 4. Here

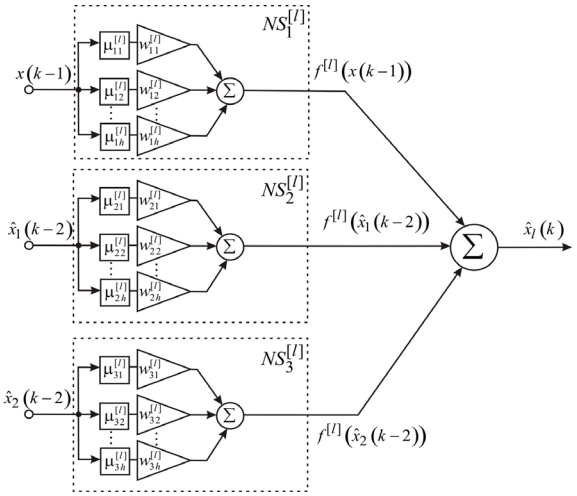

Figure 4. The special neo-fuzzy neuron's architecture for the reservoir forecasting network

$$\hat{x}_l(k) = f^{[l]}(x(k-1)) + f^{[l]}(\hat{x}_1(k-2)) + f^{[l]}(\hat{x}_2(k-2)), \tag{4}$$

$$f^{[l]}(x(k-1)) = \sum_{i=1}^{3}\sum_{j=1}^{h} \mu_{ij}^{[l]}\left(x(k-1)\right) w_{ij}^{[l]}\left(k\right), \tag{5}$$

$$f^{[l]}(\hat{x}_1(k-2)) = \sum_{i=1}^{3}\sum_{j=1}^{h} \mu_{ij}^{[l]}\left(\hat{x}_1(k-2)\right) w_{ij}^{[l]}\left(k\right), \tag{6}$$

$$f^{[l]}(\hat{x}_2(k-2)) = \sum_{i=1}^{3}\sum_{j=1}^{h} \mu_{ij}^{[l]}\left(\hat{x}_2(k-2)\right) w_{ij}^{[l]}\left(k\right). \tag{7}$$

Membership functions usually form a set of functions similar to the function array shown on fig. 5.

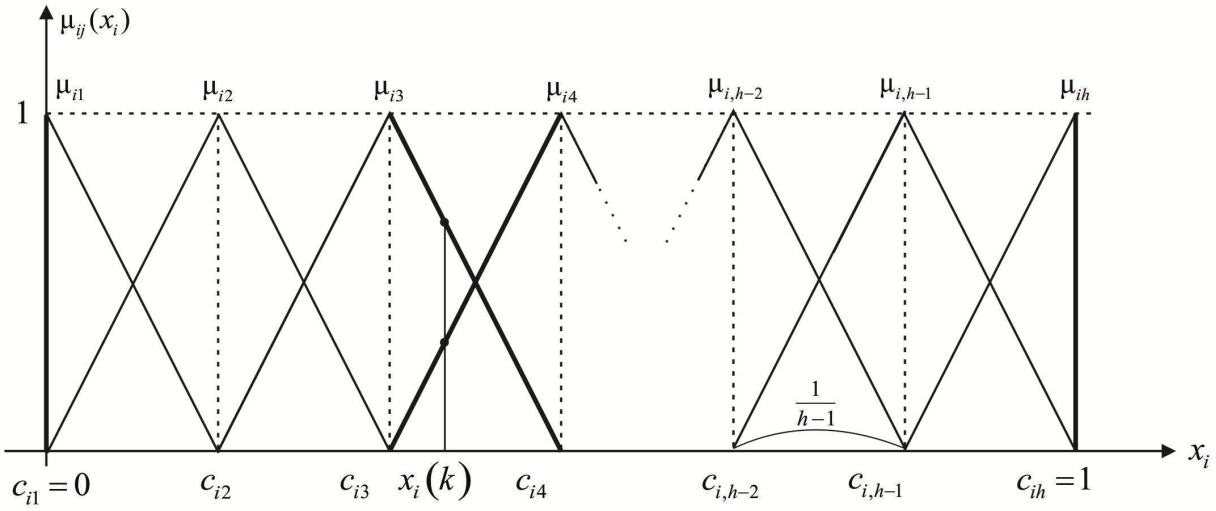

Figure 5. Triangular membership functions

Membership functions μ_{ij} of nonlinear synapses provide Ruspini (unity) partitioning which means that

$$\sum_{j=1}^{m} \mu_{ij}\left(x_i\right) = 1. \tag{8}$$

We will consider μ_{ij} to be a triangular membership function which is defined as

$$\mu_{ij}(x_i) = \begin{cases} \dfrac{c_{ij} - x_i}{c_{ij} - c_{i,j-1}}, & x_i \in (c_{i,j-1}; c_{ij}], \\ \dfrac{x_i - c_{ij}}{c_{i,j+1} - c_{ij}}, & x_i \in (c_{ij}; c_{i,j+1}], \\ 0, & otherwise \end{cases} \tag{9}$$

for $i = 1\ldots 3$, $j = 1, 2, \ldots, h$.

In our case membership functions are evenly distributed in the range $[0,1]$.

It should be noticed that triangular membership functions provide piecewise-linear approximation which can lead to the deterioration of results accuracy. To minimize this effect one may increase the amount of membership functions but this will increase the amount of synaptic weights and make a system architecture much more complicated as well as its learning algorithm. Cubic splines should be used as membership functions to get rid of the above-mentioned situations which can be written in the form:

$$\mu_{ij}\left(x_i\right) = \begin{cases} 0.25\left(2 + 3\dfrac{2x_i - c_{ij} - c_{i,j-1}}{c_{ij} - c_{i,j-1}} - \left(\dfrac{2x_i - c_{ij} - c_{i,j-1}}{c_{ij} - c_{i,j-1}}\right)^3\right), & x \in \left[c_{i,j-1}, c_{ij}\right], \\ 0.25\left(2 - 3\dfrac{2x_i - c_{i,j+1} - c_{ij}}{c_{i,j+1} - c_{ij}} + \left(\dfrac{2x_i - c_{i,j+1} - c_{ij}}{c_{i,j+1} - c_{ij}}\right)^3\right), & x \in \left(c_{ij}, c_{i,j+1}\right], \\ 0, \quad otherwise. \end{cases} \tag{10}$$

Cubic splines satisfy Ruspini partition too and improve approximation characteristics of the fuzzy inference process. On the other hand, the usage of cubic splines provides smooth polynomial approximation and allows to model nonstationary signals with high accuracy results (fig. 6).

Taking into consideration a $\left(\left(n+1\right)nhg\right)\times 1$-vector of membership functions values

$$\mu\left(k\right) = \left(\left(\mu_{11}^{[11]}\left(x_1\left(k-1\right)\right);\ldots;\mu_{21}^{[11]}\left(\hat{x}_1^1\left(k-2\right)\right);\ldots;\mu_{n+1,h}^{[11]}\left(\hat{x}_n^1\left(k-2\right)\right);\right.$$
$$\left.\left(\mu_{11}^{[gn]}\left(x_g\left(k-1\right)\right);\ldots;\mu_{21}^{[gn]}\left(\hat{x}_1^g\left(k-2\right)\right);\ldots;\mu_{n+1,h}^{[gn]}\left(\hat{x}_n^g\left(k-2\right)\right)\right)^T$$

and a synaptic weights vector of the same dimensionalities

$$w\left(k\right) = \left(\left(w_{11}^{[11]}\left(x_1\left(k-1\right)\right);\ldots;w_{21}^{[11]}\left(\hat{x}_1^1\left(k-2\right)\right);\ldots;w_{n+1,h}^{[11]}\left(\hat{x}_n^1\left(k-2\right)\right);\right.$$
$$\left.\left(w_{11}^{[gn]}\left(x_g\left(k-1\right)\right);\ldots;w_{21}^{[gn]}\left(\hat{x}_1^g\left(k-2\right)\right);\ldots;w_{n+1,h}^{[gn]}\left(\hat{x}_n^g\left(k-2\right)\right)\right)^T$$

the network output can be presented in a vector form

$$\hat{x} = w^T \mu(k). \tag{11}$$

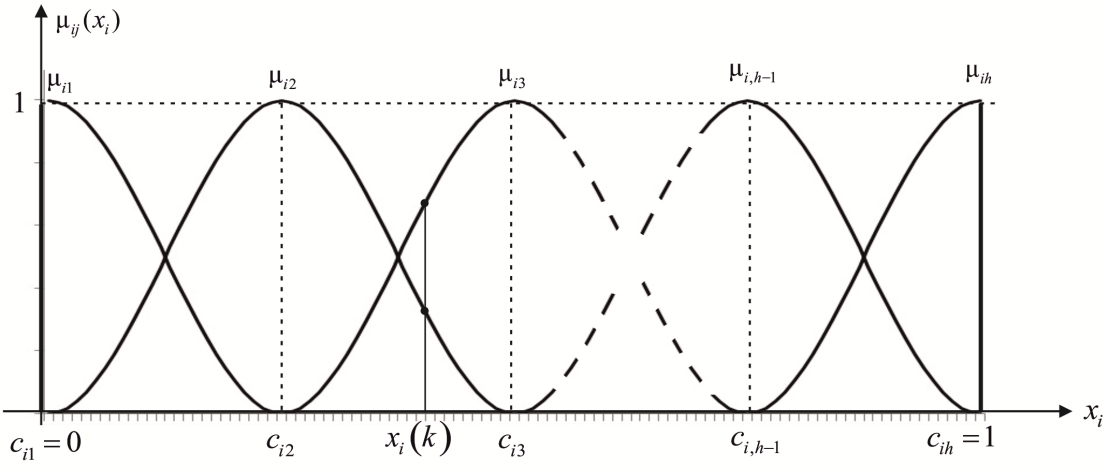

Figure 6. Cubic spline membership functions.

Learning Procedure

To find optimal values of synaptic weights the conventional learning criterion

$$E = \frac{1}{2}\sum_{k=1}^{N} \left\| y(k) - \hat{y}(k) \right\|^2 \alpha^{N-k} \tag{12}$$

and an exponentially weighted recurrent least squares optimization technique are used:

$$\begin{cases} w(k+1) = w(k) + \dfrac{P(k)(x(k+1) - w^T(k)\mu(k+1))}{1 + \mu^T(k+1)P(k)\mu(k+1)}\mu(k+1), \\ P(k+1) = \dfrac{1}{\alpha}\left(P(k) - \dfrac{P(k)\mu(k+1)\mu^T(k+1)P(k)}{\alpha + \mu^T(k+1)P(k)\mu(k+1)} \right), \ 0 < \alpha < 1. \end{cases} \tag{13}$$

Conclusion

The proposed architecture forms a "reservoir" which is built using neo-fuzzy neurons. This neuro-fuzzy network is designed to deal with time series of considerably nonstationary characteristics. The network has such advantages as numerical simplicity and high processing speed while comparing it to traditional forecasting neural networks and neuro-fuzzy systems.

Acknowledgements

The paper is published with financial support by the project ITHEA XXI of the Institute of Information Theories and Applications FOI ITHEA (www.ithea.org) and Association of Developers and Users of Intelligent Systems ADUIS Ukraine (www.aduis.com.ua).

Bibliography

[Alexandre, 2009] L.A. Alexandre, M.J. Embrechts. Reservoir size, spectral radius and connectivity in static classification problems. Ed. C. Alippi: ICANN 2009, Part I, LNCS 5768, Springer-Verlag, Berlin Heidelberg, 2009, P. 1015-1024.

[Bodyanskiy, 2009] Ye. Bodyanskiy, O. Tyshchenko Neo-fuzzy forecasting echo state network. Advanced Computer Systems and Networks: Design and Application: Proc. 4th Int. Conf. ACSN-2009, Lviv: NVF «Ukrainski Tehnologii», 2009, P. 95-96.

[Bodyanskiy, 2010] Ye. Bodyanskiy, O. Tyshchenko. The reservoir predictive neuro-fuzzy network. Proc. Int. Conf. on Intellectual Systems for Decision Making and Problems of Computational Intelligence, Kherson:KhNTU, Vol.1, 2010, P. 279-282 (in Russian).

[Horio, 2001] K. Horio, T. Yamakawa. Modified counterpropagation employing neo fuzzy neurons and its application to system modelling. Proc. Int. Conf. on Info-tech and Info-net (ICII 2001), IEEE Press, 2001, Vol.4, P. 50–55.

[Jaeger, 2001a] H. Jaeger. The echo-state approach to analysing and training recurrent neural networks. Technical report, German National Research Centre for Information Technology, 2001, 45p.

[Jaeger, 2001b] H. Jaeger. Short term memory in echo state networks. Technical report, German National Research Centre for Information Technology, 2001, 36p.

[Miki, 1999] T. Miki, T. Yamakawa. 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.

[Tyshchenko, 2012] O. Tyshchenko, I. Pliss. The forecasting neuro-neo-fuzzy network based on reservoir computing. Control Systems, Navigation and Connection Systems, No. 1(21), Vol. 2, Kyiv, 2012, P. 123-126 (in Russian).

[Uchino, 1997] E. Uchino, T. Yamakawa. 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.

[Yamakawa, 1992] T. Yamakawa, E. Uchino, T. Miki, H. Kusanagi. 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 "IIZUKA-92", Iizuka, Japan, 1992, P. 477-483.

Authors' Information

Yevgeniy Bodyanskiy *– Doctor of Technical Sciences, Professor of Artificial Intelligence Department and Scientific Head of the Control Systems Research Laboratory; Kharkiv National University of Radio Electronics, Lenina av. 14, Kharkiv, 61166, Ukraine. Tel: +380577021890,e-mail:* bodya@kture.kharkov.ua.

Major Fields of Scientific Research: Hybrid Systems of Computational Intelligence.

Iryna Pliss - *Ph.D., Leading researcher, Control Systems Research Laboratory, Kharkiv National University of Radio Electronics, Lenin Av., 14, Kharkiv, 61166, Ukraine, e-mail:* pliss@kture.kharkov.ua.

Major Fields of Scientific Research: Hybrid Neuro-Fuzzy Systems.

Oleksii Tyshchenko *– Junior Researcher, Control Systems Research Laboratory, Kharkiv National University of Radio Electronics, Lenina av. 14, Kharkiv, Ukraine, 61166. Tel: +380577021890, e-mail:lehatish@gmail.com.*

Major Fields of Scientific Research: Hybrid Neuro-Fuzzy Systems.