Proceedings of the International Symposium on Evolving Intelligent Systems, Plamen Angelov, Dimitar Filev, and Nikola Kasabov (Eds.), at the AISB 2010 convention, 29 March – 1 April 2010, De Montfort University, Leicester, UK

Cascaded Multiresolution Spline-based Fuzzy Neural Network

Vitaliy Kolodyazhniy* and Yevgeniy Bodyanskiy, *Senior Member, IEEE*

***Abstract*—A novel cascaded multiresolution spline-based fuzzy neural network (CMS-FNN) with a very fast constructive training algorithm is introduced. Structurally the CMS-FNN resembles the known cascade-correlation architecture but differs from it in the type of neurons as well as in the training algorithm. The neurons in the CMS-FNN are a generalization of the so-called neo-fuzzy neurons (NFNs) via substitution of B-splines for triangular membership function (MF) while the training algorithm relies on direct linear least squares estimation of the synaptic weights due to the linearity of the NFNs in their parameters. Training of the CMS-FNN requires only as many epochs as the number of neurons created in the network in the process of network construction. The number of MFs of the NFNs in the CMS-FNN model is increased from layer to layer to form a multi-resolution model. The efficiency of the proposed model is confirmed by long-range iterated predictions of three chaotic time series. An adaptive modification of the proposed approach is discussed and is quite straightforward.**

I. Introduction

Artificial neural networks [3] and hybrid fuzzy neural networks (FNNs) [4] have been attracting increasingly more and more attention of researchers since the early 90s. Indeed, the computational intelligence approach to nonlinear system modeling, prediction, and classification proved to be very efficient and found numerous applications. Dozens of architectures and training algorithms were proposed, including the evolving models that can operate with little prior information about the modeled system and can determine their own structure in real time and adapt their parameters to track changes [1].

At the same time, the problem of slow convergence of training algorithms relying on error backpropagation and the curse of dimensionality for systems with a large number of input variables still remain important. To tackle these problems, we previously proposed the neuro-fuzzy Kolmogorov's network (NFKN) architecture [5]-[8] based on the so called neo-fuzzy neurons (NFNs) [12] which have many remarkable properties, one of them being the linearity of the neuron output in its synaptic weights. These properties helped us accelerate the training of the NFKN compared to conventional architectures maintaining high precision, and also develop deterministic initialization procedures [5]-[8] which make unnecessary the usual repetitive training of the conventional neural networks [1] using random initial weights. However, the training of the NFKN still requires from one to several hundreds of training epochs depending on the problem [5], [6], [8].

In this paper we make an attempt to develop a novel neuro-fuzzy architecture that does not require backpropagation for its training, can be trained in several epochs, provides high precision of nonlinear system modeling, and is scalable for problems with a large number of inputs.

To construct the new architecture, we also use the NFNs as in the NFKN model [5]-[7] but in a different cascaded architecture. This enables us to derive a very fast derivative-free constructive training algorithm based on linear least squares. To improve the approximation accuracy of the cascaded neuro-fuzzy network, we generalize the NFNs by using B-spline functions as in our previous work [8].

II. Architecture

The proposed cascaded multiresolution spline-based fuzzy neural network (CMS-FNN) is shown below in Fig. 1.

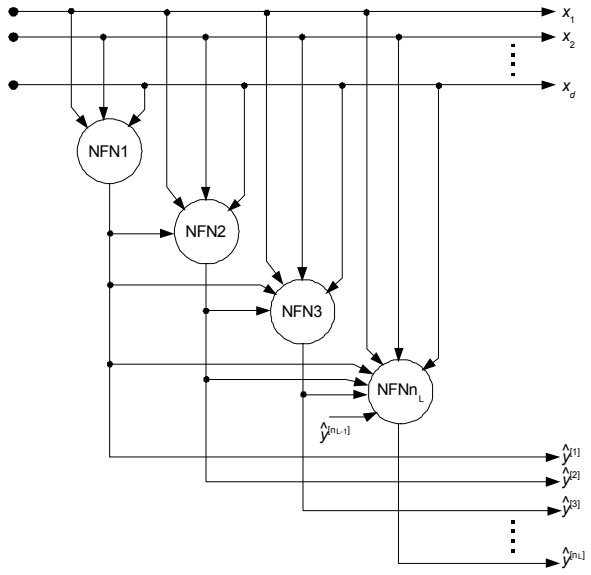

Fig. 1. Cascaded FNN

Structurally, the architecture of the CMS-FNN resembles that of a cascade-correlation neural network [2]. However,

*This research was in part supported by the 6th Framework Project EUCLOCK (No. 018741).

Vitaliy Kolodyazhniy is with the Institute for Psychology, University of Basel, Missionsstrasse 60/62, 4055 Basel, Switzerland. Tel.: +41 61 267 03 83, fax: +41 61 267 06 48, e-mail: v.kolodyazhniy@unibas.ch.

Yevgeniy Bodyanskiy is head of the Control Systems Research Laboratory at Kharkiv National University of Radioelectronics, 14, Lenin av., 61166, Kharkiv, Ukraine. Tel: +38 057 702 18 90, e-mail: bodya@kture.kharkov.ua.

we employ NFNs [12] instead of conventional artificial neurons with a sigmoid activation function as in [2].

The number of neurons in a layer is equal to the number of output variables in the modeled system, e.g. there is one neuron in each layer if there is only one output variable. Without loss of generality, we will further consider modeling of nonlinear systems with one output variable. Generalization to multiple output variables is trivial.

The use of NFNs, as will be shown below, enabled us to develop a new very fast constructive training algorithm. The structure of an NFN and a conventional neuron is shown in Fig. 2.

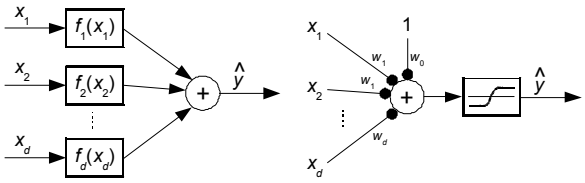

Fig. 2. Neo-fuzzy neuron and a conventional artificial neuron

The output of an NFN is computed according to the following equations [12]:

$$\hat{y} = \sum_{i=1}^{d} f_i(x_i), \quad f_i(x_i) = \sum_{h=1}^{m} \mu_{i,h}(x_i) w_{i,h}, \quad i = 1,\ldots,d, \tag{1}$$

where $\hat{y}$ is the neuron output, $f_i(x_i)$ is the function of the i-th nonlinear synapse, x_i is the i-th input, d is the number of inputs, $\mu_{i,h}$ is the h-th membership function (MF) on the input i, $w_{i,h}$ is the weight of the MF $\mu_{i,h}$, and m is the number of MFs per input. Alternatively, equations (1) can be represented in form of fuzzy rules [5]-[8]:

$$\text{IF } x_i \text{ IS } X_{i,h} \text{ THEN } \hat{y} = w_{i,h} d, \quad i = 1,\ldots,d, \quad h = 1,\ldots,m, \tag{2}$$

where $X_{i,h}$ is the linguistic term defined by $\mu_{i,h}$.

The CMS-FNN architecture consists of n_L layers (see Fig. 1). The output of layer l is computed according to equation

$$\hat{y}^{[l]} = \begin{cases} \sum_{i=1}^{d} \sum_{h=1}^{m_1} \mu_{i,h}^{[1]}(x_i) w_{i,h}^{[1]}, & \text{if } l = 1, \\ \sum_{i=1}^{d} \sum_{h=1}^{m_l} \mu_{i,h}^{[l]}(x_i) w_{i,h}^{[l]} \\ \quad + \sum_{j=1}^{l-1} \sum_{h=1}^{m_l} \mu_{j+d,h}^{[l]}(\hat{y}^{[j]}) w_{j+d,h}^{[l]}, & \text{if } l = 2,\ldots,n_L, \end{cases} \tag{3}$$

where m_l is the number of membership functions per synapse in the NFN of the layer l. We choose these numbers as $m_l = m_{l-1} + 1$ for $l = 2,\ldots,n_L$. In such a way, multi-resolution fuzzy partitions are induced where only the number of MFs in the first layer given by m_1 needs to be specified.

As is suggested in works [13] and [8], it is advantageous to define the MFs in a fuzzy system as B-splines ("basic splines"). Given a sequence of r ordered knots $\{c_1,\ldots,c_r\}$, the p-th B-spline function of order q is defined as

$$N_{p,q}(x) = \begin{cases} \begin{cases} 1, & \text{for } c_p \le x < c_{p+1}, \\ 0, & \text{otherwise}, \end{cases} & \text{if } q = 1, \\ \dfrac{x - c_p}{c_{p+q-1} - c_p} N_{p,q-1}(x) \\ \quad + \dfrac{c_{p+q} - x}{c_{p+q} - c_{p+1}} N_{p+1,q-1}(x), & \text{if } q > 1, \end{cases} \tag{4}$$

$$p = 1,\ldots,r-q.$$

Remarkably, one obtains triangular MFs letting $q = 2$ in (4). Below we will consider B-splines for $q = 2$ and $q = 4$, the latter case being cubic functions. For practical use, also quadratic B-splines for $q = 3$ can be of interest. For a spline-based neo-fuzzy neuron (1), the MFs are defined as $\mu_{i,h}(x_i) = N_{h,q}(x_i)$. As in the NFKN model [5]-[8], we assume that the MFs are fixed and equidistantly spaced over the range of each NFN input. The parameters of the MFs (spline knots) are not tuned.

To maintain the partition of unity throughout the universe of discourse of x, marginal functions are required at both ends of the universe of discourse of x in (4) for $q > 2$ (see Fig. 3) [8], [13]. The minimum number of B-functions per input is two for $q = 2$ and four for $q = 4$ including the marginal functions.

To further guarantee the partition of unity also for inputs outside the universe of discourse of x, we set the leftmost and rightmost knots to very large negative and positive values, respectively [8].

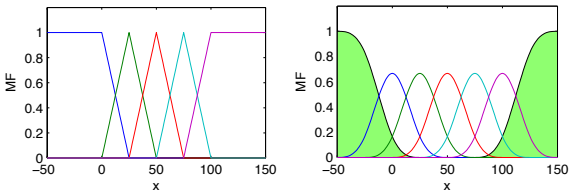

Fig. 3. B-spline membership functions of order 2 (*left*) and 4 (*right*) defined for variable $x \in [0,100]$ such that 5 membership function are defined over the universe of discourse of x. Shaded areas (*right*) correspond to marginal B-functions

It should be noted that for B-spline MFs their normality is guaranteed only for $q \le 2$, i.e. for higher orders the maximum of an MF is generally below one. This can be seen from comparison of MFs for $q = 2$ and $q = 4$ in Fig. 3. For $q \le 2$, the maximum of an MF depends on the location of its knots. Setting the leftmost and rightmost knots for the marginal MFs for $q > 2$ to very large negative and positive values, respectively, results in maxima of the marginal functions close to one.

III. Training Algorithm

The weights of the CMS-FNN are determined by means of a batch-training algorithm briefly outlined below.

Let us assume that the training of the network starts from the first layer, i.e. from one single NFN receiving $x_1,\ldots,x_d$ as its inputs. At this point no further layers exist. Since the nonlinear synapses in (1) are linear in their parameters, the synaptic weights of the first layer can be estimated with direct linear least squares (LS) optimization.

After determing the synaptic weights for the first layer, we "freeze" them and compute $\hat{y}^{[1]}$ according to (3). Then we can add a second layer and again use the LS procedure for determining the weights of the second layer.

The same procedure can be repeated to construct n_L layers. A stopping criterion can be based, e.g., on accuracy or a maximum number of weights in the network.

For each layer $l=1,\ldots,n_L$ the minimized error function is

$$E^{[l]}=\frac{1}{2}\sum_{k=1}^{K}\left[y(k)-\hat{y}^{[l]}(k)\right]^2=\frac{1}{2}\left[Y-\hat{Y}^{[l]}\right]^T\left[Y-\hat{Y}^{[l]}\right], \tag{5}$$

where $Y=\left[y(1),\ldots,y(K)\right]^T$ is the vector of target values, and $\hat{Y}^{[l]}=\left[\hat{y}^{[l]}(1),\ldots,\hat{y}^{[l]}(K)\right]^T$ is the vector of network outputs, and K is the number of data points in the training data set. To obtain the LS solution for the l-th layer, rewrite (3) as

$$y^{[l]}=\begin{cases}W^{[1]^T}\varphi^{[1]}(x_1,\ldots,x_d), & \text{if } l=1,\\ W^{[l]^T}\varphi^{[l]}(x_1,\ldots,x_d,\hat{y}^{[1]},\ldots,\hat{y}^{[l-1]}), & \text{if } l=2,\ldots,n_L.\end{cases} \tag{6}$$

We find a regularized LS solution as

$$W^{[l]}=\left(\Phi^{[l]^T}\Phi^{[l]}+\eta_l I\right)^{-1}\Phi^{[l]^T}Y,\quad l=1,\ldots,n_L, \tag{7}$$

where for the first layer $W^{[1]}=\left[w_{1,1}^{[1]},w_{1,2}^{[1]},\ldots,w_{d,m_1}^{[1]}\right]^T$,

$\Phi^{[1]}=\left[\varphi^{[1]}(x_1(1),\ldots,x_d(1)),\ldots,\varphi^{[1]}(x_1(K),\ldots,x_d(K))\right]^T$,

$\varphi^{[1]}(x_1,\ldots,x_d)=\left[\mu_{1,1}^{[1]}(x_1),\mu_{1,2}^{[1]}(x_1),\ldots,\mu_{d,m_2}^{[1]}(x_d)\right]^T$, and for the other layers for $l=2,\ldots,n_L$:

$W^{[l]}=\left[w_{1,1}^{[l]},w_{1,2}^{[l]},\ldots,w_{d,m_l}^{[l]},w_{d+1,1}^{[l]},w_{d+1,2}^{[l]},\ldots,w_{d+l-1,m_l}^{[l]}\right]^T$,

$\Phi^{[l]}=[\varphi^{[l]}(x_1(1),\ldots,x_d(1),\hat{y}^{[1]},\ldots,\hat{y}^{[l-1]}),\ldots$

$\ldots,\varphi^{[l]}(x_1(K),\ldots,x_d(K),\hat{y}^{[1]},\ldots,\hat{y}^{[l-1]})]^T$,

$\varphi^{[l]}(x_1,\ldots,x_d,\hat{y}^{[1]},\ldots,\hat{y}^{[l-1]})=$

$[\mu_{1,1}^{[l]}(x_1),\mu_{1,2}^{[l]}(x_1),\ldots,\mu_{d,m_l}^{[l]}(x_d),\mu_{d+1,1}^{[l]}(\hat{y}^{[1]}),\mu_{d+1,2}^{[l]}(\hat{y}^{[1]}),\ldots$

$\ldots,\mu_{d+l-1,m_l}^{[l]}(\hat{y}^{[l-1]})]^T$.

In the CMS-FNN training algorithm the number of training epochs is equal to the number of layers n_L that are created in the network. It is expected that for most practical applications no more than ten layers will be required.

The parameter η_l is an adjustable regularization term used to prevent rank deficiency because in the general case the matrix $\Phi^{[l]^T}\Phi^{[l]}$ cannot be inverted directly. As in [8], we compute the parameter η_0 as follows:

$$\eta_l=\eta_0\max(eig(\Phi^{[l]^T}\Phi^{[l]})),\quad l=1,\ldots,n_L, \tag{8}$$

i.e. η_l is proportional to the largest eigenvalue of the matrix $\Phi^{[l]^T}\Phi^{[l]}$ and η_0 is a small constant usually chosen in the range $10^{-9}\ldots10^{-3}$ which controls the smoothness of the obtained solution. This is also important for noisy data.

IV. Experiments

In our experiments we tested the performance of models with membership functions of order $q=2$ (triangular MFs) and $q=4$ (cubic B-spline MFs) in iterated predictions of the Mackey-Glass (MG) time series [9] generated for $\tau=17$ and $\tau=30$, and the laser time series from Santa-Fe time series competition [11].

For the MG time series, we trained networks with 17 and 30 delayed inputs using 3000 data points from 118 to 3117 and 131 to 3130 to predict the time series one step ahead. After training, the networks were used for iterated predictions of the next 1000 points without 'knowing' the actual data of the time series for these 1000 points.

For the laser time series, networks with 50 delayed inputs were trained using 950 data points from 51 to 1000, and the iterative predictions were performed for the next 100 points from 1001 to 1100.

For all the three time series, the best performance was achieved with $q=4$ (cubic B-spline MFs). In all of the tested networks with both types of MFs we set $m_1=4$. For the MG data with $\tau=17$ and the laser data the best prediction was achieved with seven layers, and with six layers for the MG data with $\tau=30$. The best results are shown in Fig. 4-6.

For both of the MG time series the real process and its prediction remain nearly indistinguishable for 1000 time steps and the difference can be seen only in the error curves with a finer scale.

For the laser data, the iterated prediction is somewhat worse than that in [8] and [11]. However, the CMS-FNN model was able to correctly predict the abrupt change in the amplitude of intensity oscillations after the 1060th time step that is a very difficult prediction task per se. For the MG time series prediction our results rank among the very best achieved so far.

The CMS-FNN model and its training were implemented in MATLAB 6.5. In all cases the training time did not exceed 6 seconds on a PC with an Intel Core2 Duo CPU running at 2 GHz with 2 GB of RAM.

V. Conclusion

A new cascaded neuro-fuzzy architecture with a very fast derivative-free constructive training algorithm was proposed. Although the CMS-FNN model is structurally similar to a cascade correlation neural network [2], the training of CMS-FNN is much faster as it is based on direct linear least squares estimation and requires only as many epochs as the number of layers.

The constructive training of the CMS-FNN model is also similar to that of the GMDH neural nets [10] but is much computationally simpler and is expected to be better scalable for problems with a large number of input variables.

An adaptive modification of the CMF-FNN training algorithm for online updating of weights and possibly structure is straightforward and will be implemented by the authors in the nearest future. Such an implementation would include a recursive least squares procedure and an online multi-step algorithm for updating the synaptic weights from layer to layer. New layers can be added or existing layers discarded without affecting the synaptic weights of the underlying layers due to the nature of the CMF-FNN model.

References

[1] P. Angelov, *Evolving Rule-Based Models*, Heidelberg-New York: Physica-Verlag, 2002.

[2] S.E. Fahlman, C. Lebiere, "The cascade-correlation learning architecture," in *Advances in Neural Information Processing Systems*, D.S. Touretzky, Ed. San Mateo, CA: Morgan Kaufman, 1990, pp. 524-532.

[3] S. Haykin, *Neural networks: A comprehensive foundation*. Upper Saddle River: Prentice Hall, 1999.

[4] J.-S. R. Jang, C.-T. Sun, and E. Mizutani, *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*, Upper Saddle River: Prentice Hall, 1997.

[5] V. Kolodyazhniy and Ye. Bodyanskiy, "Fuzzy Kolmogorov's Network," in *Lecture Notes in Computer Science* vol. 3214, M.G. Negoita et al., Ed. Berlin: Springer-Verlag, 2004, pp. 764-771.

[6] V. Kolodyazhniy, Ye. Bodyanskiy, and P. Otto, "Universal Approximator Employing Neo-Fuzzy Neurons," in *Computational Intelligence, Theory and Applications*. *Advances in Soft Computing*, vol. 33, B. Reusch, Ed. Berlin: Springer-Verlag, 2005, pp. 631-640.

[7] V. Kolodyazhniy, F. Klawonn, and K. Tschumitschew, "A Neuro-Fuzzy Model for Dimensionality Reduction and Its Application," *International Journal of Uncertainty, Fuzziness, and Knowledge-Based Systems*, vol. 15, 2007, pp. 571-593.

[8] V. Kolodyazhniy, "Spline-based neuro-fuzzy Kolmogorov's network for time series prediction," in *Proc. 17th European Symposium on Artificial Neural Networks*, Bruges, Belgium, 2009, pp. 153-158.

[9] M. C. Mackey and L. Glass, "Oscillation and chaos in physiological control systems," *Science*, vol. 197, 1977, pp. 287-289.

[10] D. T. Pham and X. Liu, *Neural Networks for Identification, Prediction and Control*, London: Springer-Verlag, 1995.

[11] A. S. Weigend and N. A. Gershenfeld, "Results of the time series prediction competition at the Santa Fe Institute," in *Proc. IEEE Int. Conf. on Neural Networks*, San Francisco, CA, 1993, vol. 3, pp. 1786-1793.

[12] T. Yamakawa, E. Uchino, T. Miki, and H. Kusanagi, "A neo fuzzy neuron and its applications to system identification and prediction of the system behavior," in *Proc. 2nd Int. Conf. on Fuzzy Logic and Neural Networks (IIZUKA-92)*, Iizuka, Japan, 1992, pp. 477-483.

[13] J. Zhang and A. Knoll, "Constructing Fuzzy Controllers with B-Spline Models— Principles and Applications," *Int. J. of Intelligent Systems*, vol. 13, 1998, pp. 257-285.

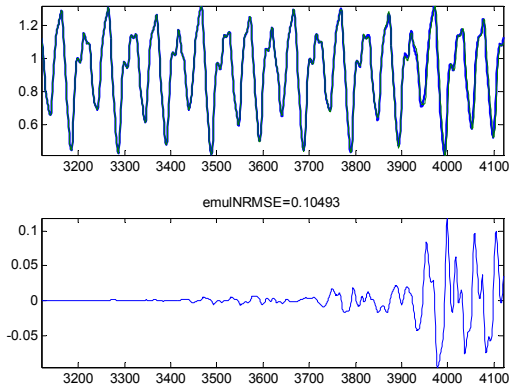

Fig. 4. MG time series for $\tau = 17$ (thick line in the upper plot), iterated prediction (thin line in the upper plot), and prediction error (lower plot)

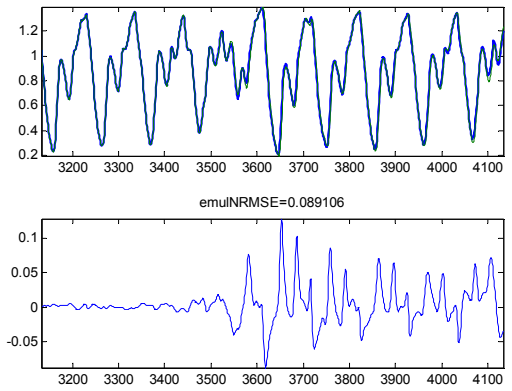

Fig. 5 MG time series for $\tau = 30$ (thick line in the upper plot), iterated prediction (thin line in the upper plot), and prediction error (lower plot)

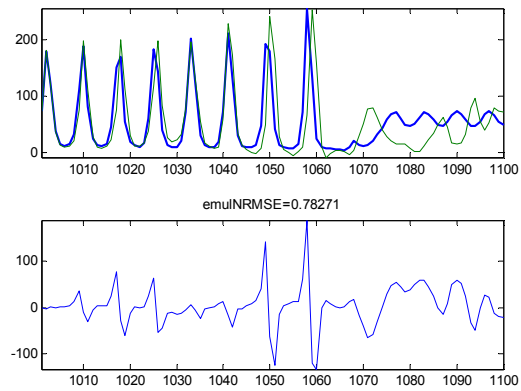

Fig. 6 Laser time series (thick line in the upper plot), iterated prediction (thin line in the upper plot), and prediction error (lower plot)