Extreme Learning for Evolving Hybrid Neural Networks

Fernando Bordignon , Fernando Gomide

Departamento de Engenharia de Computação e Automação Industrial (DCA)
Faculdade de Engenharia Elétrica e de Computação (FEEC)
Universidade Estadual de Campinas (Unicamp)
Caixa Postal 6101, 13083-970 – Campinas, SP, Brasil

{bordi,gomide}@dca.fee.unicamp.br

Abstract – This paper addresses a structure and introduces an evolving learning approach to train uninorm-based hybrid neural networks using extreme learning concepts. Evolving systems are high level adaptive systems able to simultaneously modify their structures and parameters from a stream of data, on line. Learning from data streams is a contemporary and challenging issue due to the increasing rate of the size and temporal availability of data, turning traditional learning methods impracticable. Uninorms bring flexibility and generality to fuzzy neuron models as they can behave like triangular norms, triangular conorms, or in between by adjusting identity elements. This feature adds a form of plasticity in neural network modeling. An online clustering method is used to granulate the input space, and a scheme based on extreme learning is developed to train the neural network. Computational results show that the learning approach is competitive when compared with alternative evolving modeling methods.

Keywords – hybrid neural networks; unineurons; evolving systems; online learning; extreme learning.

1. Introduction

Machine learning methods are being reevaluated over the last years as the need of online capabilities is made evident by the massive growth in numbers and accessibility of computing devices [1] [10]. With this context in mind a new class of machine learning approach with adaptive abilities to simultaneously learn a system structure and its parameters emerged, namely, the evolving system approach. The term evolving means online, gradual systems development and adaptation. Evolving systems are an alternative and innovative way of adapt, learn and represent knowledge about changing environments [1].

In this paper we address a hybrid neural fuzzy network with evolving learning whose structure has two main parts: a fuzzy neural system and a neural network. The multilayer structure of the network has membership functions in the input layer neurons, uninorm-based neurons in the second layer, and classic neurons in the third layer. Uninorms provide flexibility at the cost of an extra parameter (identity element) to learn for each unineuron. Interesting, however, the identity element can be either chosen by the designer if he has prior knowledge or needs a particular structure, or leave it to be learnt using data. Due to its inherent highly nonlinear nature, learning complexity can be reduced using extreme learning [7]. Basically, extreme learning consists in randomly assigning weights for the hidden layer of feed-forward neural networks and analytically determine the output weights. Extreme learning has shown to provide fast training, accurate results and good generalization performance [8].

After this brief introduction, the paper proceeds as follows. Section 2. gives detailed description of the learning steps including the evolving fuzzy neural network system approach. Next, computational results of the tests conducted using well known benchmarks are reported in Section 3.. Section 4. concludes the paper summarizing its contributions and suggesting issues for future investigation.

2. Evolving Extreme Learning Uninetwork

In this section we present an evolving hybrid fuzzy neural network based on uninorms. Learning involves recursive clustering to granulate the input space and extreme learning algorithms to simultaneously adjust the network weights and parameters. For short, we call the evolving hybrid fuzzy neural network model eXUninet (evolving eXtreme learning Uninetwork) in the rest of the paper.

2.1. Uninorms and the Unineuron

A generalization of t-norms and t-conorms called uninorm was introduced by [13]. Formally, a Uninorm is a mapping $u : [0,1] \times [0,1] \rightarrow [0,1]$ that satisfies the following properties:

1. Commutativity: $a\ u\ b = b\ u\ a$
2. Associativity: $a\ u\ (b\ u\ c) = (a\ u\ b)\ u\ c$
3. Monotonicity: if $b \leq c$, then $a\ u\ b \leq a\ u\ c$
4. Identity element: $a\ u\ e = a,\ \forall\ a\ \in [0,1]$

While t-norms respect the first three conditions with the identity element $e = 1$, t-conorms satisfies the first three as well but with identity element $e = 0$. Uninorms extends triangular norms by allowing the identity element to be in the unity interval.

Fuzzy neurons were introduced as basic units of fuzzy neural networks in [12]. The two basic neuron models were called or and and, they are defined using t-norms and s-norms as follows:

$$or(A, W) = S_{i=1}^{n}(a_i \; t \; w_i) \tag{1}$$

$$and(A, W) = T_{i=1}^{n}(a_i \; s \; w_i) \tag{2}$$

Here T is a t-norm, S is a s-norm, $a_i \in [0, 1]$ are the inputs and $w_i \in [0, 1]$ the synaptic weights. Neural networks based on these neurons were successfully used in various applications such as thermal modeling of power transformers [5] and time series prediction [3]. Recent work suggested uninorm-based neurons (unineurons) [11] to explore uninorms in modeling distinct neuron operators, a desired characteristic to add flexibility in fuzzy neural networks. Further improvements of the unineuron model was reported in [6]. The unineuron used in the hybrid neural fuzzy network addressed in this paper is of the type OR_U, which consists of uninorms at synaptic processing level and a s-norm at global aggregation level, as shown in (3).

$$OR\_U = S_{i=1}^{n}(a_i \; u \; w_i) \tag{3}$$

The uninorm realization used in this paper adopts the following construct [13]

$$u(a, b) = \begin{cases} a \; s \; b, & \text{if } a, b \in [e, 1] \\ a \; t \; b, & \text{else} \end{cases} \tag{4}$$

with t a t-norm and s a s-norm.

2.2. Topology of the Network

The structure of the uninetwork adopted in this paper based on [4]. The network has two major parts, fuzzy inference systems and aggregation neural network, assembled into three layers. The fuzzy neural system is composed by the first two layers, the input and hidden layer, respectively. The input layer consists of neurons whose activation functions are membership functions of fuzzy sets that granulate the input space to form a fuzzy partition. Here we use Gaussian membership functions centered at c_l with dispersion (radius) σ. The membership degree of input x_i is computed using (5). For each dimension x_i of a n-dimensional input vector $\mathbf{x}$ there are L^t fuzzy sets $A_i^{l_i}$, $l_i = 1, \ldots, L^t$. L^t corresponds to the number of fuzzy rules of the system at step t.

$$a_{li} = e^{-\frac{(x_i - c_{li})^2}{2\sigma^2}} \tag{5}$$

Here $l = 1, \ldots, L^t$, $i = 1, \ldots, n$ and c_{li} is the ith coordinate of the lth cluster center. The radius σ is defined a priori and kept constant during the evolving process.

The second layer contains OR_U unineurons to aggregate the outputs of the input layer weighted by connections w_{li}:

$$z_l = S_{i=1}^{n}(a_{li} \; u \; w_{li}) \tag{6}$$

Here z_l , $l = 1, \ldots, L^t$ is the output of the lth unineuron, S implemented using the max s-norm; w_{li} are the synaptic weights. The third layer is a traditional neural network layer with sigmoidal activation functions $f(\cdot)$ as follows:

$$\hat{y}_j = f\left(\sum_{l=1}^{L^t} r_{jl} z_l\right) \tag{7}$$

with m being the dimension of the output space, $j = 1, \ldots, m$, and r_{jl} are the output weights connecting the jth output with the lth rule. Figure 1 shows the uninetwork structure.

2.3. Clustering Procedure

Most batch fuzzy neural networks relies on a clustering method to granulate input data. We present an alternative called eClustering+ [2] to perform clustering online. This approach estimates the density recursively at each data point by using (8).

$$D_t(o_t) = \left((t-1)\,(so_t + 1) + b_t - 2\sum_{j=1}^{n+m} o_{tj} h_{tj}\right)^{-1} \tag{8}$$

with $o_t = [\mathbf{x}^T, \mathbf{y}^T]^T$; $D_1(o_1) = 1$; $b_t = b_{t-1} + \sum_{j=1}^{n+m} o_{(t-1)j}^2$; $b_1 = 0$; $h_{tj} = h_{(t-1)j} + o_{(t-1)j}$; $h_{1j} = 0$; $j = 1, \ldots, n+m$; $so_t = \sum_{j=1}^{n+m} o_{tj}^2$.

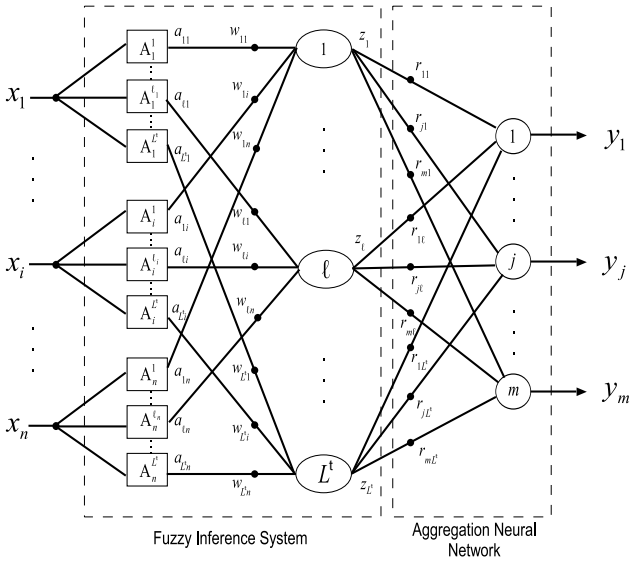

Figure 1. Uninetwork structure

Density is calculated recursively for each cluster center by (9):

$$D_t(o^{i^*}) = \frac{t-1}{t-1+(t-2)\left(\frac{1}{D_{t-1}(o^{i^*})}-1\right)+dist} \tag{9}$$

where $dist = \sum_{j=1}^{n+m}(o_{tj} - o_{(t-1)j})$ and $D_1(o^{i^*}) = 1$. Cluster centers candidates are selected if the following condition is true: Condition A: $D_t(o_t) > \max D_t(o^{i^*})$ *or* $D_t(o_t) < \min D_t(o^{i^*})$ where i^* are the indexes of cluster centers. To avoid overlapping and redundancy, the cluster is created only if it does not satisfy: Condition B: $\exists l, l \in [i, L^t] : a_{li} > e^{-1}, \forall i, i \in [1, n]$.

2.4. Extreme Learning

The extreme learning method was introduced as a way to train single hidden layer feed-forward neural networks (SLFNNs). Originally developed in [7], the method chooses the hidden layer weights randomly and estimates the weights of the output layer using the least squares algorithm. Results indicate good performance and greater ability to generalize [8]. Interestingly, the authors have shown that general non-linear activation functions can be used by the hidden layer neurons.

The eXUninet training is performed similarly assigning random values in $[0, 1]$ to weights and identity elements of the OR_U neurons, and next using the recursive least squares (RLS) algorithm with forgetting factor λ to update the output layer weights $\mathbf{r}^t = [r_{11}^t \dots r_{jl}^t \dots r_{mL^t}^t]$:

$$\mathbf{J}^t = \mathbf{Q}^{t-1}\mathbf{z}^t\{\lambda + \mathbf{z}^t\mathbf{Q}^{t-1}(\mathbf{z}^t)^T\}^{-1} \tag{10}$$

$$\mathbf{Q}^t = (\mathbf{I}_{L^t} - \mathbf{J}^t\mathbf{z}^t)\lambda^{-1}\mathbf{Q}^{t-1} \tag{11}$$

$$\mathbf{r}^t = \mathbf{r}^{t-1} + (\mathbf{J}^t)^T(f^{-1}(\mathbf{y}^t) - \mathbf{z}^t(\mathbf{r}^{t-1})^T) \tag{12}$$

With $f^{-1}(\mathbf{y}^t) = \log(\mathbf{y}^t) - \log(1 - \mathbf{y}^t)$. Initialization of $\mathbf{Q}$ is commonly $\mathbf{I}_{L^t}\omega$, $\omega = 1000$, where $\mathbf{I}_{L^t}$ is the identity matrix.

3. Computational Results

Simulations were performed setting the forgetting factor at 0.9. The root mean squared error (RMSE) is computed at step t:

$$RMSE = \sqrt{\frac{1}{t}\sum_{j=1}^{t}(y^j - \hat{y}^j)^2} \tag{13}$$

Data were normalized in the range $[0.1, 0.9]$, and the errors were computed for normalized data. All the results reported were obtained running each method 20 times. The best and average performances are displayed.

3.1. Mackey-Glass Time Series

The Mackey-Glass time series is a well known benchmark whose data is generated using:

$$\frac{dx}{dt} = \frac{Ax^{t-\tau}}{1+(x^{t-\tau})^C} - Bx^t \ , \ A, B, C > 0 \tag{14}$$

Semi-periodic or chaotic behaviour depends of the parameters values chosen. Several studies adopt: $A = 0.2$, $B = 0.1$, $C = 10$ and $\tau = 17$, with a time step of 0.1 for integration [1, 9], 3200 data samples were generated and used for training and testing simultaneously. The goal is to predict the value of x^t 85 steps ahead, that is:

$$x^{t+85} = p(x^t, x^{t-6}, x^{t-12}, x^{t-18}) \tag{15}$$

Results comparing the best and average RMSE values with alternative approaches suggested in the literature are shown in Table 1.

Clearly, for this case the eXUninet performance is the best. Figure 2 depicts the results of eXUninet for the first 500 data samples.

Table 1. Performances for Mackey-Glass

Model	Ref.	Rules	RMSE	avg. RMSE
eTS	[1]	24	0.0779	0.0779
DENFIS	[9]	25	0.0730	0.0730
FBeM	[10]	26	0.0968	0.0968
eXUninet	-	27	0.0501	0.0622

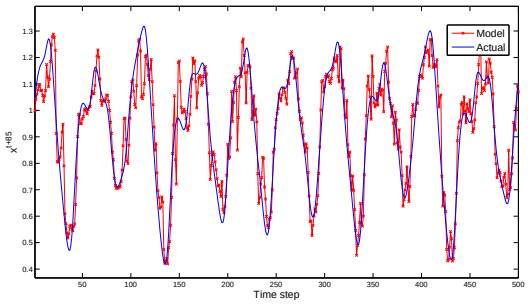

Figure 2. Mackey-Glass results

4. Conclusion

This paper has suggested a new evolving learning approach for hybrid fuzzy neural networks based on uninorms. Computational results show that the fuzzy neural network is competitive with alternative evolving methods. The learning approach is simple, fast and accurate, and hence suitable for online applications.

Future work shall explore examples with significant concept drift to further test the online learning capability of the approach. Detailed statistical analysis is also required to show that the evolving learning approach is statistically superior. One can also analyse the use of various types of uninorms realizations at the global and local levels of neurons in the fuzzy layer of the eXUninet, as well as extensions for nullnorms-based neural fuzzy networks and systems.

References

[1] P. Angelov and Xiaowei Zhou. Evolving fuzzy systems from data streams in real-time. In *Evolving Fuzzy Systems, 2006 International Symposium on*, pages 29 –35, sept. 2006.

[2] Plamen Angelov. Evolving takagi-sugeno fuzzy systems from streaming data (ets+). In Plamen Angelov, Dimitar Filev, and N. Kasabov, editors, *Evolving Intelligent Systems*, pages 21–50. John Wiley & Sons, Inc., 2010.

[3] R. Ballini, S. Soares, and F. Gomide. A recurrent neuro-fuzzy network structure and learning procedure. In *Fuzzy Systems, 2001. The 10th IEEE International Conference on*, volume 3, pages 1408 –1411, 2001.

[4] M. Hell, P. Costa, and F. Gomide. Hybrid neurofuzzy computing with nullneurons. In *Neural Networks, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, pages 3653 –3659, june 2008.

[5] M. Hell, P. Costa, and F. Gomide. Participatory learning in power transformers thermal modeling. *Power Delivery, IEEE Transactions on*, 23(4):2058 –2067, oct. 2008.

[6] M. Hell, F. Gomide, R. Ballini, and P. Costa. Uninetworks in time series forecasting. In *Fuzzy Information Processing Society, 2009. NAFIPS 2009. Annual Meeting of the North American*, pages 1 –6, june 2009.

[7] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. Extreme learning machine: a new learning scheme of feedforward neural networks. In *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*, volume 2, pages 985 – 990 vol.2, july 2004.

[8] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. Extreme learning machine: Theory and applications. *Neurocomputing*, 70(1-3):489 – 501, 2006. Neural Networks Selected Papers from the 7th Brazilian Symposium on Neural Networks.

[9] N.K. Kasabov and Qun Song. Denfis: dynamic evolving neural-fuzzy inference system and its application for time-series prediction. *Fuzzy Systems, IEEE Transactions on*, 10(2):144 –154, apr 2002.

[10] Daniel Leite, Fernando Gomide, Rosangela Ballini, and Pyramo Costa Jr. Fuzzy granular evolving modeling for time series prediction. In *FUZZ-IEEE*, pages 2794–2801. IEEE, 2011.

[11] W. Pedrycz. Logic-based fuzzy neurocomputing with unineurons. *Fuzzy Systems, IEEE Transactions on*, 14(6):860 –873, dec. 2006.

[12] W. Pedrycz and A.F. Rocha. Fuzzy-set based models of neurons and knowledge-based networks. *Fuzzy Systems, IEEE Transactions on*, 1(4):254 –266, nov 1993.

[13] Ronald R. Yager and Alexander Rybalov. Uninorm aggregation operators. *Fuzzy Sets Syst.*, 80(1):111–120, May 1996.