

# Universal Approximation with Uninorm-Based Fuzzy Neural Networks

Andre Lemos\*, Vladik Kreinovich<sup>†</sup>, Walmir Caminhas\*, Fernando Gomide<sup>‡</sup>

\* Department of Electrical Engineering

Federal University of Minas Gerais - Belo Horizonte-MG, Brazil - 31270-901

Email: {andrepl,caminhas}@cpdee.ufmg.br

<sup>†</sup> Department of Computer Sciences

University of Texas - El Paso-TX, USA - 79968

Email: vladik@cs.utep.edu

<sup>‡</sup> Department of Computer Engineering and Automation

University of Campinas - Campinas-SP, Brazil - 19083-970

Email: gomide@dca.fee.unicamp.br

**Abstract**—Fuzzy neural networks are hybrid models capable to approximate functions with high precision and to generate transparent models, enabling the extraction of valuable information from the resulting topology. In this paper we will show that the recently proposed fuzzy neural network based on weighted uninorms aggregations uniformly approximates any real functions on any compact set. We will describe the network topology and inference mechanism and show that the universal approximation property of this network is valid for a given choice of operators.

## I. INTRODUCTION

Fuzzy neural networks are a synergy between fuzzy set theory, as a mechanism for information compactation and knowledge representation, and neural networks. They provide a network like topology and enables the utilization of a large variety of data based learning procedures. The main feature of these networks is its transparency, enabling the utilization of a priori information to define initial network topology and the extraction of valuable information from the resulting topology after training in the form of a set of fuzzy rules [1]. Fuzzy neural networks based on logic neurons, called *and* and *or* [2], have been used to solve a large variety of problems, such as pattern classification [1], time series prediction [3] and dynamic system modelling [4], [5]. These type of logic based neurons are nonlinear mappings of the form  $[0, 1]^n \rightarrow [0, 1]$  whose standard implementation of fuzzy connectives involves triangular norms. More specifically *and* neurons are implemented through t-norms while *or* neurons are implemented using t-conorms (s-norms).

Some of most recent trend in the neurofuzzy systems area involves the development of new operators (logic connectives) [4], [6]–[12]. Apart from traditional operators several modifications and extensions have been addressed in literature. The most recent conceptual developments in this line involve the concept of *uninorms* [13] and *nullnorms* [14].

The concept of uninorm inspired the development of new operators, most of them seen as an extension of the *and* and *or* neurons, called unineurons [6], [10]. The processing of these

neurons occurs in two levels. At first level, the input signals are individually combined with the weights. At second level, a overall aggregation operation is made over the results of all combinations of the first level. Traditional logic neurons uses t-norms and s-norms to perform the operations described in these two levels. For instance, [6] proposes uninorm based neurons where the triangular norms used in the first or the second level are replaced by uninorms. More recently, [10] extended [6] using uninorms to perform operations at each of the two levels.

In [12] an alternative approach to construct uninorm based neurons was proposed. These neurons perform weighted aggregation of their inputs and weighted uninorm aggregation, similarly as discussed in [13]. The proposed unineuron generalizes the *and* and *or* neurons. Neural fuzzy network constructed with unineurons has a transparent topology and represents a set of linguistic fuzzy rules. Also, the inference mechanism used for this network processing is similar with the uninorm based fuzzy inference proposed in [15], which has been proved to provide universal function approximation [16].

In this paper, we will prove that for a specific choice of operators, weighted uninorm based fuzzy neural networks [12] are capable of uniformly approximating any real continuous function on a compact set to arbitrary accuracy. We will also show that this property does not hold for any choice of operators.

The paper is organized as follows. Section II we review the necessary basic concepts about fuzzy set theory and describe the recently proposed uninorm based neuron. In section III we detail the neural network addressed in [12] using this new type of unineuron. Next, section IV discusses the universal approximation properties of this network. Finally, the conclusions and further developments are summarized in section V.

## II. UNINORM BASED LOGIC NEURONS

### A. Uninorms

Uninorms are generalizations of t-norms and t-conorms (s-norms) [13]. These norms allow a smooth connection between "and" and "or" type of aggregation depending on a numeric parameter called *identity element*.

A uninorm is a mapping  $U : [0, 1]^2 \rightarrow [0, 1]$  with the following properties:

- 1) *Commutativity*:  $U(x, y) = U(y, x)$ .
- 2) *Monotonicity*:  $U(x, y) \geq U(u, v)$  for  $x \geq u, y \geq v$ .
- 3) *Associativity*:  $U(x, U(y, z)) = U(U(x, y), z)$ .
- 4) *Identity*: There exists an identity element  $g \in [0, 1]$ , such that  $U(x, g) = x$ .

Uninorms extend t-norms and s-norm allowing the values of the identity element  $g$  vary between 0 and 1. Therefore the uninorm can switch smoothly between a s-norm (if  $g = 0$ ) and a t-norm (if  $g = 1$ ). Evidently, when  $g = 0$  the uninorm is the "or" type of aggregation, and when  $g = 1$  the uninorm is the "and" type of aggregation.

A large variety of uninorms realizations can be found in the literature [13], [15]. One of the most widely used adopts the following construct [6]:

$$U(x, y) = \begin{cases} g \cdot T\left(\frac{x}{g}, \frac{y}{g}\right) & \text{if } x, y \in [0, g] \\ g + (1-g) \cdot S\left(\frac{x-g}{1-g}, \frac{y-g}{1-g}\right) & \text{if } x, y \in (g, 1] \\ \max(x, y) \text{ or } \min(x, y) & \text{otherwise} \end{cases} \quad (1)$$

where  $T$  represents a t-norm and  $S$  a s-norm. This construction does not require any duality relationship between the specific t-norm and s-norm. Figure 1 illustrates two fundamental variants of this construct based on the max and min operators.

Analysing (1) one can note that is possible to deliver some flexibility to the specific realization of the uninorm by the choice of minimum or maximum operators for the intermediate regions of the uninorm (denoted by  $\Omega$  in Figure 1). As discussed in [7], the uninorm  $U$  is *and-dominated* if one chooses the minimum operator (the largest triangular norm) for the intermediate regions, and *or-dominated* if one chooses the maximum operator (the smallest triangular norm) for these regions.

### B. Fuzzy Logic Neurons

Fuzzy logic neurons are functional units that performs multivariate nonlinear operations in unit hypercubes  $[0, 1]^n \rightarrow [0, 1]$  [17]. The name "logic" is associated with disjunction *or* and conjunction *and* operations performed by these neurons viewed as t-norms and s-norms.

The *or* and *and* neurons aggregate input signals  $\mathbf{a} = [a_1, a_2, \dots, a_n]$  by first combining them individually with the weights (or connections)  $\mathbf{w} = [w_1, w_2, \dots, w_n]$ ,  $\mathbf{a}, \mathbf{w} \in [0, 1]^n$  and combining afterward these results globally as follows:

$$\begin{aligned} y &= \text{or}(\mathbf{a}, \mathbf{w}) = S_{i=1}^n a_i t w_i \\ y &= \text{and}(\mathbf{a}, \mathbf{w}) = T_{i=1}^n a_i s w_i \end{aligned} \quad (2)$$

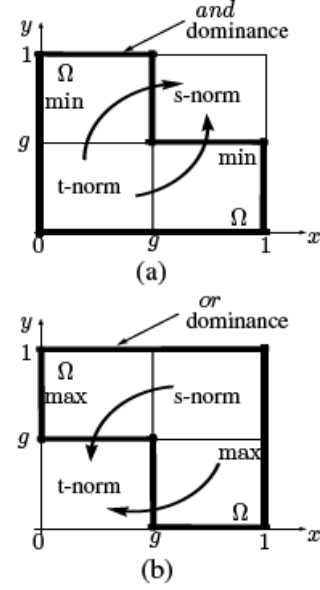


Fig. 1. Realizations of a uninorm: (a) *and-dominated*, (b) *or-dominated*

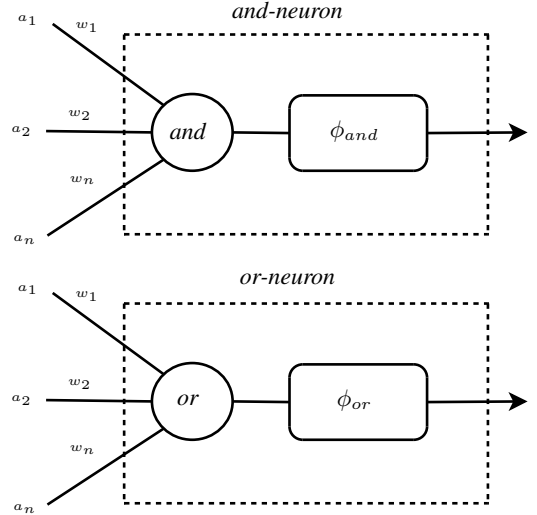


Fig. 2. Fuzzy neurons

where  $T$  and  $t$  are t-norms and  $S$  and  $s$  are s-norms, respectively.

The *or* neuron is interpreted as a logic expression that performs a *and*-type local aggregation of the inputs and the weights using a t-norm, followed by a *or*-type global aggregation of the results using a s-norm. The *and* neuron is interpreted similarly, with a *or*-type local aggregation and a *and*-type global aggregation. Figure 2 illustrates the neurons.

The activation functions  $\phi_{and}$  and  $\phi_{or}$  can, in general, be nonlinear mappings, but here we assume these two functions to be the identity function, i.e.,  $\phi_{and}(\xi) = \phi_{or}(\xi) = \xi$ .

According to [18], the local aggregations performed by the neurons can be interpreted as weighting operations of the inputs, since the role of the weights is to differentiate between particular levels of impact that the individual inputs might have on the result of the global aggregation. For the *or* neuron, lower values of  $w_i$  discount the impact of the corresponding input, while higher values do not affect the original value of the inputs. In the limit, if all weights are set to 1, the neuron output is a plain *or* combination of the inputs. For the *and* neuron, the interpretation of the values of the weights is inverse, i.e., higher values of  $w_i$  reduce the impact of the corresponding input. For this neuron, in the limit, if all weights are set to 0, the output is a plain *and* combination of the inputs.

### C. Unineuron

As discussed above, the processing of logic neurons occurs in two levels. At first level, the input signals  $\mathbf{a} = [a_1, a_2, \dots, a_n]$  are individually combined with the weights  $\mathbf{w} = [w_1, w_2, \dots, w_n]$ , i.e.,  $L_1 : (a_i, w_i) \rightarrow [0, 1]$ . At second level, an overall logic aggregation is computed using the results of the first level, that is

$$z = L_2[L_1(a_1, w_1), L_1(a_2, w_2), \dots, L_1(a_n, w_n)] \quad (3)$$

For the *and* neuron,  $L_1$  is performed by any realization of the *or* operator (s-norm),  $L_1 = or$ , while  $L_2$  concerns the realization of an *and* operator (t-norm),  $L_2 = and$ . The structure of the *and* neuron is dual to that of *and* neuron, it is, for *or* neurons,  $L_1 = and$  and  $L_2 = or$ .

Pedrycz [6] and Hell [10] propose uninorm based logic neurons that use uninorms to generalize the classical logic neurons, replacing the  $L_1$  and/or  $L_2$  composition operations by uninorms. Depending on where the uninorm is used in aggregation, three possible realizations can be addressed: (a) uninorm based processing realized at local level  $L_1$ , (b) uninorm based aggregation realized at global level  $L_2$  and (c) uninorm based at both, local processing and global level. [6] addresses the cases (a) and (b), while [10] addresses case (c). The unineuron proposed in [10] is as follows:

$$y = \mathbf{UNI}(\mathbf{a}, \mathbf{w}; g) = \tilde{U}_{i=1}^n [\tilde{U}(a_i, w_i; 1-g); g] \quad (4)$$

where  $n$  is the number of inputs,  $w_i$  is the weight associated with the input  $a_i$ , and  $\tilde{U}$  is the modified uninorm:

$$\tilde{U}(x, y) = \begin{cases} g \cdot T(\frac{x}{g}, \frac{y}{g}) & , \text{if } x, y \in [0, g] \\ g + (1-g) \cdot S(\frac{x-g}{1-g}, \frac{y-g}{1-g}) & , \text{if } x, y \in (g, 1] \\ \varphi(x, y) & , \text{otherwise} \end{cases} \quad (5)$$

where  $\varphi(x, y)$  is

$$\varphi(x, y) = \begin{cases} \max(x, y) & , \text{if } g \in [0, 0.5] \\ \min(x, y) & , \text{if } g \in (0.5, 1] \end{cases} \quad (6)$$

Lemos [12] proposed an alternative approach to construct uninorm based logic neurons. Logical neurons are interpreted as weighted aggregation of the inputs, and the unineuron viewed as weighted uninorm aggregation.

According to [13], given a collection of  $n$  pairs  $(a_i, w_i)$ , with  $a_i \in [0, 1]$  is a input and  $w_i \in [0, 1]$ , the corresponding weight, the following steps must be followed to perform weighted uninorm aggregation:

- transform each pair  $(a_i, w_i)$  into a single value  $b_i = h(a_i, w_i)$ ;
- compute the uninorm aggregation of the transformed values  $U(b_1, b_2, \dots, b_n)$ .

The function  $h$ , responsible for transforming the inputs and corresponding weights into single transformed values, is named *relevancy transformation* and must satisfy the following conditions:

- *monotonicity in value*: for  $a > a'$  it is required that  $h(w, a) \geq h(w, a')$ , in other words, if the input value increases the transformed value must also increase;
- *zero importance elements should have no effect*:  $h(0, a) = g$ , where  $g$  is the identity element of the uninorm aggregation;
- *normality of importance of one*:  $h(1, a) = a$ , i.e., the effect of  $h$  on  $a$  when  $w = 1$  is simply return the value of  $a$ ;
- *consistency of effect of  $w_i$* : as  $w$  goes from 0 to 1,  $h(w, a)$  monotonically moves from  $g$  to  $a$ .

Yager [15] suggests two possible formulations for the relevancy transformation  $h$  that satisfies the required conditions:

$$h(w, a) = (w \wedge a) \vee (\bar{w} \wedge g) \vee (a \wedge g) \quad (7)$$

$$h(w, a) = w \cdot a + \bar{w} \cdot g \quad (8)$$

The unineuron proposed in [12] is as follows:

$$y = \mathbf{WUNI}(w, x; g) = \tilde{U}_{i=1}^n h(w_i, x_i) \quad (9)$$

where  $\tilde{U}$  is the modified uninorm (5).

The dominance of the modified uninorm (5) is controlled by the value of the identity element  $g$ . When  $g \in [0, 0.5]$  the global aggregation becomes *and*-dominated and when  $g \in (0.5, 1]$  the global aggregation becomes *or*-dominated.

Using the first relevancy transformation (7), when  $g = 0$ , the global aggregation becomes a s-norm and the relevancy transformation becomes  $h(w, a) = w \wedge a$ . Thus, when  $g = 0$ , the unineuron becomes an *or*-neuron. When  $g = 1$ , the global aggregation becomes a t-norm, the relevancy aggregation becomes  $h(w, a) = \bar{w} \vee a$ , and the unineuron becomes an *and*-neuron replacing  $w$  by its complement. The use of the weight complement instead of the original value in the *and*-neuron unifies the interpretation of the weights for the two logic neurons, since low values of  $w$  reduce the impact of the corresponding input and high values increase the impact. Thus, using the first relevancy transformation (7), the unineuron

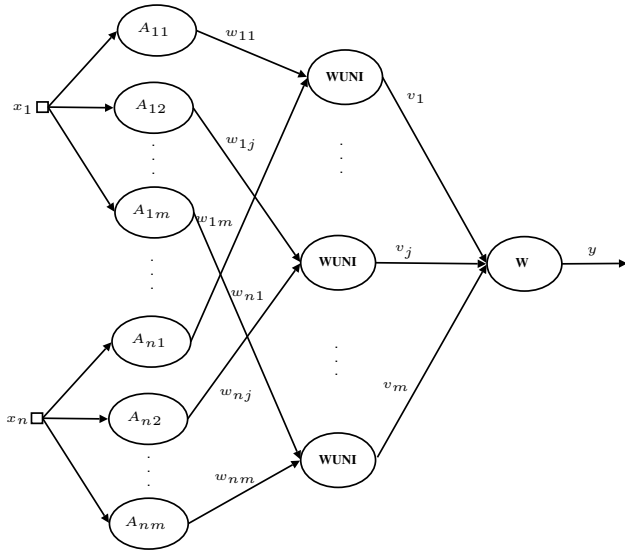


Fig. 3. Feedforward uninorm-based fuzzy neural network

proposed can be viewed as a generalization of the classic logic neurons.

### III. UNINORM BASED FUZZY NEURAL NETWORKS

The unineuron described on the previous section was used in [12] to construct neural fuzzy networks. Figure 3 illustrates the feedforward topology of the network.

The first layer is composed by neurons whose activation functions are membership functions of fuzzy sets defined by the input space granulation. For each input variable  $x_i$ ,  $m$  fuzzy sets are defined  $A_{ij}, j = 1, \dots, m$  whose membership functions are the activation functions of the corresponding neurons. Thus, the outputs of the first layer are the membership degrees associated with the input values, i.e.,  $a_{ij} = \mu_{A_{ij}}$  for  $i = 1, \dots, n$  and  $j = 1, \dots, m$ , where  $n$  is the number of inputs and  $m$  is the number of fuzzy sets for each input (note that all inputs are partitioned using the same number of fuzzy sets).

Input space granulation is done using a clustering algorithm. All fuzzy sets generated by the input space granulation have Gaussian membership functions with the modal values computed as the projections of the clusters centers in the input space. The spreads of the Gaussian membership functions are free parameters to be optimized.

The second layer is composed by unineurons (9). Each unineuron performs a weighted aggregation of the outputs of the first layer. The aggregation is done using the weights  $w_{ij}$  (for  $i = 1, \dots, n$  and  $j = 1, \dots, m$ ) which also are parameters to be optimized. Given  $n$  inputs  $x_1, \dots, x_n$ , each neuron  $j$  ( $1 \leq j \leq m$ ) produces a value

$$y_j = \tilde{U}_{i=1}^n h(w_{ij}, A_{ij}(x_i)), \quad (10)$$

The number of unineurons is equal to the number of clusters found by the clustering step. Each unineuron performs

a weighted aggregation of the fuzzy sets created by the projection of the corresponding cluster. All unineurons in this layer uses the product as t-norm, the probabilistic sum as s-norm and the relevancy transformation  $h$  given in (8). The uninorm and the relevancy transformation can, in principle, correspond to different values of the identity  $g$ .

The network structure provides a set of  $m$  *if-then* rules of the form:

$$R_j \quad \begin{array}{l} \text{If } x_1 \text{ is } A_{1j} \text{ with relevancy } w_{1j} \dots \\ \text{and/or } x_k \text{ is } A_{kj} \text{ with relevancy } w_{kj} \dots \\ \text{and/or } x_n \text{ is } A_{nj} \text{ with relevancy } w_{nj} \\ \text{Then } y_j \text{ is } v_j \end{array} \quad (11)$$

where  $v_j$  for  $i = 1, \dots, m$  is a fuzzy singleton.

The last layer (output layer) uses a classic neuron, denoted by  $W$ , to compute the network output. Overall, the network architecture process information according to a fuzzy inference process similar to the one proposed by [15]. Neuron  $W$  computes its output as follows:

$$y = \min \left( 1, \sum_{j=1}^m y_j \cdot v_j \right) \quad (12)$$

where  $y_j$  for  $j = 1, \dots, m$  is the output of each unineuron of the second layer (10).

According to [15], given a set of *if-then* fuzzy rules, a fuzzy inference process is composed by an aggregation step, in which contributions of all rules of the model are combined, and a second step characterized by the determination of the contribution of each component of the model based on its relevancy to the current input. Yager [15] suggests an uninorm operator to implement the aggregation step and a *relevancy transformation operator* (RET) for the determination of the rule contribution step. The neuron of the output layer uses an aggregation operator of the form  $A(a, b) = \min(1, a + b)$  and the RET operator  $w_i x_i$ . The output layer of the fuzzy neural network performs fuzzy inference process as described by [15] because the output layer unineuron can be viewed as a uninorm global aggregation of weighted inputs, with the weight operation done by a RET operator.

The training procedure suggested in [12] for this network uses genetic algorithm for parameter optimization. The network has already been used for nonlinear systems identification. The results obtained suggest that the network is a promising alternative to build accurate and transparent models.

### IV. UNIVERSAL APPROXIMATION

In this section we analyse whether the proposed uninorm-based network has a universal approximation property, i.e., whether an arbitrary continuous function  $f : X \rightarrow [0, 1]$  defined on a bounded closed set  $X \subseteq \mathbb{R}^n$  – e.g., on a box

$$[\underline{x}_1, \bar{x}_1] \times \dots \times [\underline{x}_n, \bar{x}_n],$$

can be, for an arbitrary  $\varepsilon > 0$ , approximated, with this accuracy, by an appropriate function of type (10), (12). In

other words, we need to check whether for every such function  $f(x_1, \dots, x_n)$  and for every  $\varepsilon > 0$ , there exists a uninorm, a relevancy transformation, and the weights  $w_{ij}$  and  $v_j$  for which, for every input  $x = (x_1, \dots, x_n)$ , the value (12) differs from  $f(x)$  by no more than  $\varepsilon$ .

**Result.** Our result is that the proposed network does have a universal approximation property.

**Proof.** To prove the universal approximation property, consider as the relevancy transformation  $h(w, u) = w \cdot u$ ; this relevancy transformation corresponds to (8) with  $g = 0$ . We will take  $w_{ij} = 1$ , so  $h(w, u) = u$ . We will also take  $U(a, b) = a \cdot b$ . This uninorm corresponds to a different identity  $g = 1$ , but this is acceptable because we have explicitly allowed different values of the identity for the uninorm and for the relevancy transformation.

In our construction, we will use two small positive real numbers  $h > 0$  and  $\delta > 0$ . We will prove that for every  $\varepsilon > 0$ , once  $h > 0$  and  $\delta > 0$  are small enough, the network output approximates  $f$  with the desired accuracy  $\varepsilon$ . In the set  $X$ , we select a rectangular grid with steps  $h$ . Let  $N$  denote the total number of such points. Points from this grid will be denoted by  $x^{(j)} = (x_1^{(j)}, \dots, x_n^{(j)})$ . To each of these  $N$  points, we put into correspondence a neuron of type (10), with the Gaussian membership function  $A_{ij}(x) = \exp\left(-\frac{(x_i - x_i^{(j)})^2}{\delta^2}\right)$ . Then, due to our choice of the relevancy transformation  $h$  and the weights  $w_{ij}$ , we get  $h(w_{ij}, A_{ij}(x_i)) = A_{ij}(x_i)$  and therefore, due to our choice of the uninorm,

$$y_j = \prod_{i=1}^n \exp\left(-\frac{(x_i - x_i^{(j)})^2}{\delta^2}\right) = A_0\left(\frac{x - x^{(j)}}{\delta}\right),$$

where we denoted  $A_0(z) \stackrel{\text{def}}{=} \exp(-\|z\|^2)$  and  $\|z\|^2 \stackrel{\text{def}}{=} z_1^2 + \dots + z_n^2$ .

The signal  $y = \sum_{j=1}^N v_j \cdot y_j$  then takes the form

$$y = \sum_{j=1}^N v_j \cdot y_j = \sum_{j=1}^N v_j \cdot A_0\left(\frac{x - x^{(j)}}{\delta}\right). \quad (13)$$

Let us denote  $\int A_0(z) dz$  by  $I_0$ . Then, for every  $\delta > 0$ , we have

$$\int A_0\left(\frac{z}{\delta}\right) dz = \int A_0(z') d(\delta \cdot z') = \delta^n \cdot \int A_0(z') dz' = \delta^n \cdot I_0.$$

To select the appropriate weights  $v_j$ , we take into account that for every continuous function  $f(x)$  on a bounded set  $X$ , when  $\delta \rightarrow 0$ , the convolution

$$\begin{aligned} f_\delta(x) &\stackrel{\text{def}}{=} \frac{\int f(z) \cdot A_0\left(\frac{x - z}{\delta}\right) dz}{\int A_0\left(\frac{z}{\delta}\right) dz} \\ &= \int f(z) \cdot \frac{1}{\delta^n \cdot I_0} \cdot A_0\left(\frac{x - z}{\delta}\right) dz \end{aligned} \quad (14)$$

uniformly converges to the original function  $f(x)$ . Thus, for every  $\varepsilon > 0$ , for sufficiently small  $\delta$ , we have  $|f_\delta(x) - f(x)| \leq$

$\frac{\varepsilon}{2}$ . The integral (14), in turn, can be approximated, with arbitrary accuracy by an appropriate integral sum, i.e., by an expression

$$e(x) = f(x^{(j)}) \cdot \frac{1}{\delta^n \cdot I_0} \cdot h^n \cdot A_0\left(\frac{x - x^{(j)}}{\delta}\right). \quad (15)$$

In other words, for every  $\varepsilon > 0$ , for sufficiently small  $h$ , we have  $|e(x) - f_\delta(x)| \leq \frac{\varepsilon}{2}$  and thus,

$$|e(x) - f(x)| \leq |e(x) - f_\delta(x)| + |f_\delta(x) - f(x)| \leq \frac{\varepsilon}{2} + \frac{\varepsilon}{2} = \varepsilon.$$

The expression (15) is exactly the result (13) of our neural network, with the weights  $v_j = f(x^{(j)}) \cdot \frac{1}{\delta^n \cdot I_0} \cdot h^n$ . Thus, for sufficiently small  $\delta$  and  $h$ , if we select these weights we see the output of the neural network is  $\varepsilon$ -close to the original function  $f(x)$ . The universal approximation property is proven.

**Natural question.** In the above text, we have shown that for every continuous function  $f(x_1, \dots, x_n)$  and for every  $\varepsilon > 0$ , we can select a uninorm, a relevancy transformation, and the values of the weights for which the corresponding neural network approximates the given function with a given accuracy. In [16], for a similar uninorm approximation problem, a stronger result is proven: that even when we pre-select an arbitrary uninorm and an arbitrary relevancy transformation, we still get a universal approximation property. A natural question is whether such a strengthening is possible in our case.

**Our answer: formulation.** Our answer is negative: we will give an example of a uninorm and a relevancy transformation for which even the function  $f(x_1, x_2) = x_1 \cdot x_2$  cannot be approximated for  $x_1, x_2 \in [0, 1]$ .

Thus, to get a universal approximation property, it is not enough to find appropriate weights: we also need to select an appropriate uninorm and/or an appropriate relevancy transformation.

**Our answer: proof.** For our example, we will take the same product relevancy transformation  $h(a, b) = a \cdot b$  as before. As an uninorm, instead of the product, let us take the maximum (corresponding to  $g = 0$ ). Let us prove, by contradiction, that this neural network cannot approximate the product.

Let us assume that for every  $\varepsilon > 0$ , we have a neural network that  $\varepsilon$ -approximates the product. In this case, the signal generated by each neuron has the form  $y_j^{(\varepsilon)} = \max_i w_{ij}^{(\varepsilon)} \cdot A_{ij}^{(\varepsilon)}(x_i)$ . This expression can be simplified if we denote  $B_{ij}^{(\varepsilon)}(x_i) \stackrel{\text{def}}{=} w_{ij}^{(\varepsilon)} \cdot A_{ij}^{(\varepsilon)}(x_i)$ . In terms of these new notation, we have  $y_j^{(\varepsilon)} = \max_i B_{ij}^{(\varepsilon)}(x_i)$ . In particular, for our case  $n = 2$ , we have

$$y_j^{(\varepsilon)}(x_1, x_2) = \max(B_{1j}^{(\varepsilon)}(x_1), B_{2j}^{(\varepsilon)}(x_2)). \quad (16)$$

Let us prove that for all possible values of  $x_1, x'_1, x_2$ , and  $x'_2$ , this function satisfies the property

$$y_j^{(\varepsilon)}(x'_1, x_2) + y_j^{(\varepsilon)}(x_1, x'_2) \geq y_j^{(\varepsilon)}(x_1, x_2). \quad (17)$$

This can be proven by considering the cases  $B_{1j}^{(\varepsilon)}(x_1) \leq B_{2j}^{(\varepsilon)}(x_2)$  and  $B_{1j}^{(\varepsilon)}(x_1) \geq B_{2j}^{(\varepsilon)}(x_2)$ . In the first we have

$$y_j^{(\varepsilon)}(x_1, x_2) = \max(B_{1j}^{(\varepsilon)}(x_1), B_{2j}^{(\varepsilon)}(x_2)) = B_{2j}^{(\varepsilon)}(x_2).$$

By (16), we have

$$y_j^{(\varepsilon)}(x'_1, x_2) = \max(B_{1j}^{(\varepsilon)}(x'_1), B_{2j}^{(\varepsilon)}(x_2)) \geq B_{2j}^{(\varepsilon)}(x_2),$$

hence  $y_j^{(\varepsilon)}(x'_1, x_2) \geq y_j^{(\varepsilon)}(x_1, x_2)$  and thus, (17) it is true (since the value of  $y_j^{(\varepsilon)}$  is always non-negative).

Similarly, in the second case, we have

$$y_j^{(\varepsilon)}(x_1, x_2) = \max(B_{1j}^{(\varepsilon)}(x_1), B_{2j}^{(\varepsilon)}(x_2)) = B_{1j}^{(\varepsilon)}(x_1).$$

By (16), we have

$$y_j^{(\varepsilon)}(x_1, x'_2) = \max(B_{1j}^{(\varepsilon)}(x_1), B_{2j}^{(\varepsilon)}(x'_2)) \geq B_{1j}^{(\varepsilon)}(x_1),$$

hence  $y_j^{(\varepsilon)}(x_1, x'_2) \geq y_j^{(\varepsilon)}(x_1, x_2)$  and thus, (17) is also true. The statement is proven.

The overall signal has the form  $y^{(\varepsilon)}(x_1, x_2) = \min\left(\sum_{j=1}^N v_j^{(\varepsilon)} \cdot y_j^{(\varepsilon)}\right)$ . For values of  $x = (x_1, x_2)$  for which  $f(x_1, x_2) < 1 - \varepsilon$ , to approximate the value  $f(x_1, x_2)$  with accuracy  $\leq \varepsilon$ , we must take  $y < 1$ ; thus, for such values, we have  $y^{(\varepsilon)}(x_1, x_2) = \sum_{j=1}^N v_j \cdot y_j^{(\varepsilon)}$ .

Multiplying both sides of (17) by  $v_j^{(\varepsilon)}$  and adding up the resulting inequalities, we can now conclude that

$$y^{(\varepsilon)}(x'_1, x_2) + y^{(\varepsilon)}(x_1, x'_2) \geq y^{(\varepsilon)}(x_1, x_2). \quad (18)$$

We assumed that the corresponding neural networks approximate the function  $f(x_1, x_2)$  with an arbitrary accuracy  $\varepsilon$ . For every  $\varepsilon$ , we have the inequality (18). In the limit  $\varepsilon \rightarrow 0$ , we have  $y^{(\varepsilon)}(x_1, x_2) \rightarrow f(x_1, x_2)$  and therefore, (18) turns into

$$f(x'_1, x_2) + f(x_1, x'_2) \geq f(x_1, x_2). \quad (19)$$

In particular, for  $x_1 = x_2 = 0.5$  and  $x'_1 = x'_2 = 0$ , we get  $0 + 0 = 0 \geq 0.25$  – a contradiction. This contradiction proves that for this selection of a uninorm and a relevancy transformation, we do not have the universal approximation property.

## V. CONCLUSION

In this paper we have addressed the approximation capability of a new class of unineuron based fuzzy neural networks. The topology of the analysed neural fuzzy network encodes a set of linguistic fuzzy rules, and the processing mechanism of the network models is similar to the uninorm-based fuzzy inference mechanism.

We have proved that, for selected uninorm and relevancy transformation operators, the analysed neural network is capable of uniformly approximating any real continuous function on a compact set to arbitrary accuracy. We have also proved that this property does not hold for any choice of operators.

Future work shall address how to characterize the classes of weighted uninorm operators for which universal approximation holds, the development of new types of unineuron networks, and comparison of these networks with alternative approaches for nonlinear system modeling and time series forecasting.

## ACKNOWLEDGMENT

The authors thank the Brazilian National Research Council, CNPq, for grants 141323/2009-4, 309666/2007-4 and 304596/2009-4. Waldir Caminhas also acknowledges the support of FAPEMIG, the Research Foundation of the State of Minas Gerais, for grant PPM-00252-09.

## REFERENCES

- [1] W. Caminhas, H. Tavares, F. Gomide, and W. Pedrycz, "Fuzzy sets based neural networks: Structure, learning and applications," *Journal of Advanced Computational Intelligence*, vol. 3, no. 3, pp. 151–157, 1999.
- [2] F. Gomide and W. Pedrycz, *Fuzzy Systems Engineering: Toward Human-Centric Computing*. NJ, USA: Wiley Interscience, 2007.
- [3] R. Ballini and F. Gomide, "Learning in recurrent, hybrid neurofuzzy networks," in *IEEE International Conference on Fuzzy Systems*, 2002, pp. 785–791.
- [4] M. Hell, P. Costa, and F. Gomide, "Participatory learning in power transformers thermal modeling," *IEEE Transactions on Power Delivery*, vol. 23, no. 4, pp. 2058–2067, Oct. 2008.
- [5] A. E. Gobi and W. Pedrycz, "Logic minimization as an efficient means of fuzzy structure discovery," *IEEE Transactions on Fuzzy Systems*, vol. 16, no. 3, pp. 553–566, JUN 2008.
- [6] W. Pedrycz, "Logic-based fuzzy neurocomputing with unineurons," *IEEE Transactions on Fuzzy Systems*, vol. 14, no. 6, pp. 860–873, DEC 2006.
- [7] W. Pedrycz and K. Hirota, "Uninorm-based logic neurons as adaptive and interpretable processing constructs," *Soft Computing*, vol. 11, no. 1, pp. 41–52, JAN 2007.
- [8] M. Hell, P. Costa, and F. Gomide, "Nullneurons-based hybrid neurofuzzy network," in *Fuzzy Information Processing Society, 2007. NAFIPS '07. Annual Meeting of the North American*, 2007, pp. 331–336.
- [9] W. Pedrycz and R. A. Aliev, "Logic-oriented neural networks for fuzzy neurocomputing," *Neurocomputing*, vol. 73, no. 1-3, pp. 10–23, 2009.
- [10] M. Hell, F. Gomide, R. Ballini, and P. Costa, "Uninetworks in time series forecasting," in *NAFIPS 2009. Annual Meeting of the North American Fuzzy Information Processing Society, 2009.*, june 2009, pp. 1–6.
- [11] X. Liang and W. Pedrycz, "Logic-based fuzzy networks: A study in system modeling with triangular norms and uninorms," *Fuzzy Sets and Systems*, vol. 160, no. 24, pp. 3475–3502, 2009.
- [12] A. Lemos, W. Caminhas, and F. Gomide, "New uninorm-based neuron model and fuzzy neural networks," in *Fuzzy Information Processing Society (NAFIPS), 2010 Annual Meeting of the North American*, 2010, pp. 1–6.
- [13] R. Yager and A. Rybalov, "Uninorm aggregation operators," *Fuzzy Sets and Systems*, vol. 80, no. 1, pp. 111–120, MAY 27 1996.
- [14] T. Calvo, B. D. Baets, and J. Fodor, "The functional equations of frank and alsina for uninorms and nullnorms," *Fuzzy Sets and Systems*, vol. 120, no. 3, pp. 385–394, 2001.
- [15] R. Yager, "Uninorms in fuzzy systems modeling," *Fuzzy Sets and Systems*, vol. 122, no. 1, pp. 167–175, AUG 16 2001.
- [16] R. R. Yager and V. Kreinovich, "Universal approximation theorem for uninorm-based fuzzy systems modeling," *Fuzzy Sets and Systems*, vol. 140, no. 2, pp. 331–339, 2003.
- [17] W. Pedrycz, "Fuzzy Neural Networks and Neurocomputations," *Fuzzy Sets and Systems*, vol. 56, no. 1, pp. 1–28, MAY 25 1993.
- [18] —, "Heterogeneous fuzzy logic networks: Fundamentals and development studies," *IEEE Transactions on Neural Networks*, vol. 15, no. 6, pp. 1466–1481, NOV 2004.