Uninorm Based Fuzzy Network for Tree Data Structures

Angelo Ciaramella[1], Witold Pedrycz[2], and Alfredo Petrosino[1]

[1] Dept. of Applied Sciences University of Naples "Parthenope",
Isola C4, Centro Direzionale, I-80143, Napoli, Italy
{ciaramella,petrosino}@uniparthenope.it
[2] Dept. of Electrical and Computer Engineering
University of Alberta, Edmonton AB, Canada
pedrycz@ee.ualberta.ca

Abstract. The aim of this study is to introduce a fuzzy model to process structured data. A structured organization of information is typically required by symbolic processing. Most connectionist models assume that data are organized in a form of relatively simple structures such as vectors or sequences. In this work, we propose a connectionist model that can directly process labeled trees. The model is based on a new category of logic connectives and logic neurons that use the concept of uninorms. Uninorms are a generalization of t-norms and t-conorms used for aggregating fuzzy sets. Using a back-propagation algorithm we optimize the parameters of the model (relations and membership functions). The learning issues are presented and some experimental results obtained for synthetic realistic data, are reported.

Keywords: Graphical Models, Trees, Fuzzy Logic Connectives, t-norms (t-conorms), Uninorms, Structured Data.

1 Introduction

Recently, structured data is becoming more and more important in data mining and data analysis. Structured domains are characterized by complex patterns which are usually represented as lists, trees and graphs of variable sizes and complexity. Data can be naturally represented by tree or graph structures in several application areas, including proteomics and molecular biology, image analysis, scene description, software engineering, natural language processing, XML document retrieval and others.

While neural networks are able to classify static information or temporal sequences, the current state of the art does not allow the efficient classification of structures of different size. The standard way to approach the classification of structured data by using a neural network is to encode the tree or graph in a fixed-size vector. In detail, it is encoded in a vector which is fed to a feedforward neural network for classification. The encoding process is usually defined *a priori* and does not depend on the classification task. The *a priori* definition of the encoding process has two main drawbacks

1. the relevance of different features of the graphs may change dramatically for different learning tasks.
2. each graph must get a different representation; this may result in vectorial representaions which are very difficult to classify.

To overcome the above difficulties, in [13] the authors propose to adapt the encoding process through an additional neural network which is trained to learn the best way to encode the graphs for the given classification task. To achieve this aim, they introduce a generalization of a recursive neuron. Moreover in [7] and [10] a Recursive Neural Network (RNN) for acyclic graphs was proposed. The RNN model can only process Directed Positional Acyclic Graphs (DPAGs). More recently, an extended model was proposed that can cope with non-positional acyclic graphs with labeled edges (i.e. DAGs) [2].

To solve the classification of labeled trees and handling encertainty, we propose a model based on fuzzy information. A fuzzy connectionist structure is developed from the input tree topology, by extending the capabilities of the RNN model. It is already demostrated [8] that it is possible to encode a nondeterministic fuzzy tree automata into a RNN. This encoding has been studied from a theoretical point of view, proving its stability. This theoretical study led to the model design reported in the present paper. Specifically, In the structure we propose the nodes are connected by using a relation and the composition of information is obtained by using an uninorm-based connective.

The paper is organized as follows. In Section 2 the notation about the graph adopted in the paper is introduced. In Section 3, we describe the uninorm-based generalization of norms and in Section 4 we introduce the Fuzzy Recursive Neural Model. In Section 5, we present several experimental results.

2 Graph and Tree Notation

A labeled graph (or graph) **G** is a quadruple (**N**, **E**, λ, ϵ), where **N** is the set of nodes (or vertices), and **E** is the set of edges between nodes, i.e. $\mathbf{E} \subseteq \{(u,v)|u,v \in \mathbf{N}\}$. Nodes and edges constitute the *skeleton* of the graph. The last two items, that associate vectors of real numbers of dimension respectively $\Re^{l_N}$ and $\Re^{l_E}$ to each node and edge, are respectively a node-labeling function $\lambda : \mathbf{N} \rightarrow \Re^{l_N}$ and an *edge-labeling* function $\epsilon : \mathbf{E} \rightarrow \Re^{l_E}$. Node labels are represented by $\mathbf{l}_n$ and edge labels by $\mathbf{l}_{uv}$. If the graph is directed, an edge (u,v) is an ordered pair of nodes, where u is the father and v its child. If the graph is undirected, the ordering between u and v in (u,v) is not defined, i.e. $(u,v) = (v,u)$. A graph is called acyclic if there is no path, i.e. a sequence of connected edges, that starts and ends at the same node. Combining some properties it is possible to specify various graph categories: Directed Positional Acyclic Graphs (DPAGs), Directed Acyclic Graphs (DAGs) and other graph-oriented structures. A tree **T** could be defined as an acyclic connected graph, represented by a quadruple (**N**, **E**, λ, ϵ), where each node has a set of zero or more children nodes and at most one parent node.

3 Uninorms

Triangular norms (t-norms) and the corresponding t-conorms play a fundamental role in several branches of mathematics [11], e.g., in probabilistic metric spaces, the theory of generalized measures, game theory, and fuzzy logic. The semantics of logic operators (logic connectives) in fuzzy sets is enormously rich. Some of the most recent conceptual

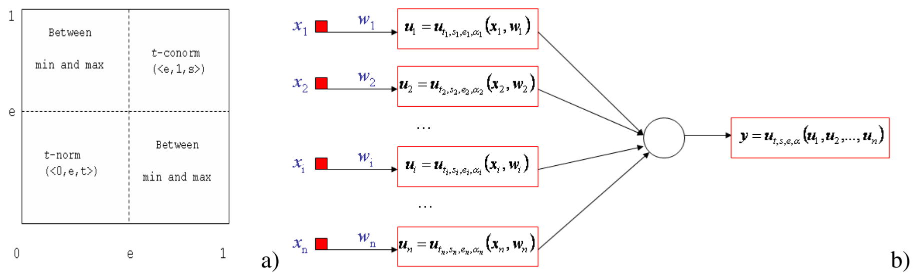
Between min and max
t-conorm (<e,1,s>)
t-norm (<0,e,t>)
Between min and max
1
e
0
e
1
a)
x_1 w_1 $u_1 = u_{t_1,s_1,e_1,\alpha_1}(x_1, w_1)$
x_2 w_2 $u_2 = u_{t_2,s_2,e_2,\alpha_2}(x_2, w_2)$
x_i w_i $u_i = u_{t_i,s_i,e_i,\alpha_i}(x_i, w_i)$
x_n w_n $u_n = u_{t_n,s_n,e_n,\alpha_n}(x_n, w_n)$
$y = u_{t,s,e,\alpha}(u_1, u_2, \dots, u_n)$
b)

Fig. 1. Uninorm based structures : a) uninorm with neutral element e; b) unineuron

developments along this line involve uninorms [12,6,9,14], nullnorms [1] and ordinal sums [15] of t-norms, just to name a few of them.

We start our discussion about uninorms by recalling that both a neutral element 1 of a t-norm and the neutral element 0 of a t-conorm are boundary points of the unit interval. A $uninorm$ is a binary operation $u : [0,1]^2 \to [0,1]$ which satisfies the properties of *Commutativity*, *Monotonicity*, *Associativity* and it has a *neutral element* $e \in [0,1]$ (see Figure 1a). Noticeably, we allow the values of the identity element e to vary in-between 0 and 1. As a result of this, we can implement switching between pure AND and OR properties of the logic operators occurring in this construct. In this study we confine ourselves to the following family of constructs that seem to be highly interpretative and thus intuitively appealing:

Let t be a t-norm, s be a t-conorm and $e \in [0,1]$. In the spirit of the construction of Ordinal Sums the following operation $\mathbf{u}_{t,s,e,\alpha} : [0,1]^2 \to [0,1]$ ($\alpha = \min$ or $\alpha = \max$) make $[0,1]$ into fully ordered semigroups with neutral element e:

$$\mathbf{u}_{t,s,e,\alpha}(x,y) = \begin{cases} e \cdot t(\frac{x}{e}, \frac{y}{e}) & \text{if } (x,y) \in [0,e]^2 \\ e + (1-e) \cdot s(\frac{x-e}{1-e}, \frac{y-e}{1-e}) & \text{if } (x,y) \in]e,1]^2 \\ \alpha(x,y) & \text{otherwise} \end{cases} \tag{1}$$

Obviously, $\mathbf{u}_{t,s,e,\min}$ is a conjunctive, and $\mathbf{u}_{t,s,e,\max}$ is a disjunctive uninorm.

Interestingly, we observe that the two intermediate regions deliver some flexibility to the specific realization of the uninorm [12].

3.1 Uninorm-Based Logic Neuron

The previous studies carried out in the realm of logic-based neurocomputing, we can distinguish between two general categories of neurons that are OR and AND neurons. We already proposed in literature extensions of these operators [5,4,12].

Let $\mathbf{x}$ be a vector in the unit hypercube, $\mathbf{x} \in [0,1]^n$ and y denote an element in $[0,1]$. Formally speaking, the underlying logic processing is governed by a composition between the individual inputs and the corresponding connections (weights) $\mathbf{w} \in [0,1]^n$. In detail, $L_1 : (x_i, w_i) \to [0,1]$ followed by some overall logic aggregation L_2 giving rise to the output y, that is

$$y = L_2[L_1(x_1, w_1), L_1(x_2, w_2), \dots, L_1(x_n, w_n)] \tag{2}$$

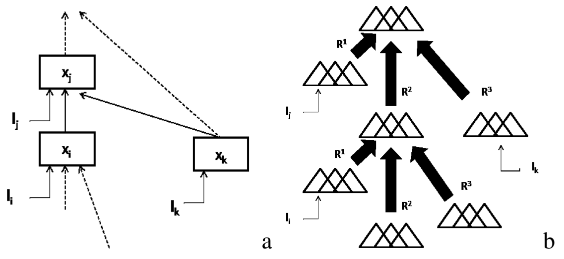

Fig. 2. Node connection: a) connection between nodes; b) fuzzyfication of the nodes

In the OR neuron, the semantics of L_1 is concerned with any realization of the AND operator (t-norm), L_1 = AND. L_2 concerns the realization of the OR operator (s-norm), L_2 = OR. On the contrary, in the AND neuron we have these two operations reversed, that is L_1 = OR and L_2 = AND.

By using the uninorms, that admit more flexibility into the overall aggregation, we lead to the expression (see Figure 1b)

$$y = u[u_1(x_1, w_1, e_1, \alpha_1), u_2(x_2, w_2, e_2, \alpha_2), \ldots, u_n(x_n, w_n, e_n, \alpha_n), e] \tag{3}$$

where $u_i = \mathbf{u}_{t_i, s_i, e_i, \alpha_i}$ is the i-th uninorm with parameters t_i, s_i, e_i and α_i to be estimated by the optimization process. Moreover, $u = \mathbf{u}_{t,s,e,\alpha}$ is the uninorm that permits to obtain the overall composition [5].

4 Uninorm Based Fuzzy Network

In this Section we introduce the proposed Uninorm Based Fuzzy Network (UFN) model. Our framework for tree processing must implement a function φ that computes an output $\varphi(\mathbf{T})$ for each tree $\mathbf{T}$. Each node, that can be considered as a state, is described by M real attributes x_n where the dimension M is a predefined parameter. The state x_n of the n-th node is fuzzy: K membership functions are used to describe the node information. We denote with $\mu(\mathrm{x}_n)$ this fuzzyfied state. In Figure 2a and 2b we show a possible connection between the nodes x_j, x_i and x_k with the corresponding fuzzy sets obtained from the fuzzyfication. The M attributes of the node are also fuzzified. As show in Figure 3 if we consider the n-th state then the M attributes $\mathrm{l}_n^1 \ldots \mathrm{l}_n^M$ are fuzzified obtaining the K fuzzy sets $\mu^1(\mathrm{l}_n^i) \ldots \mu^K(\mathrm{l}_n^i)$ for each attribute i. Now we stress that the membership functions $\mu(\mathrm{l}_n) = [\mu_1(\mathrm{l}_n) \ldots \mu_K(\mathrm{l}_n)]$ of the n-th node can be generated by uninorm-based neurons (as shown in Figure 3a). For this reason the composition of the k-th fuzzy set is

$$\mu_k(\mathrm{l}_n) = \bigvee_{i=1}^{M} \left(R_{att}(\mu^k(\mathrm{l}_n^i))_i^k t \mu^k(\mathrm{l}_n^i)\right) \tag{4}$$

where $R_{att}(\mu^k(\mathrm{l}_n^i))_i^k$ is the weight (relation) between the k-th fuzzy set of the i-th attribute and the k-th membership function of $\mu(\mathrm{l}_n)$, t is a t-norm and $\bigvee$ is an s-norm depending from the uni-neuron definition. To compose the fuzzy state $\mu(\mathrm{x}_n)$

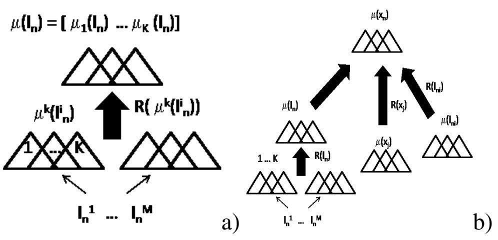

Fig. 3. Composition of the fuzzy sets for the node labels: a) attribute composition; b) node composition

we define a fuzzy relation between the k-th membership $\mu_k(\mathbf{l}_n)$ and the j-th fuzzy set $\mu_j(\mathbf{x}_n)$. We denote this relation as $R_{label}(\mu(\mathbf{l}_n))^j_k$. To simplify our derivation, in the following we consider $k = j$ (diagonal relation) and we denote the relation as $R_{label}(\mu(\mathbf{l}_n))^j$.

In the same way we can define the membership functions of the edge labels. If we consider $\mathbf{l}_{ni}$ to be the label between the states n and i, respectively, then we denote as $\mu_k(\mathbf{l}_{ni})$ the membership function of the k-th fuzzy set. In this case the weights of the relation become $R_{edge}(\mu(\mathbf{l}_{ni}))^j_k$ (also in this case for $k = j$, $R_{edge}(\mu(\mathbf{l}_{ni}))^j$).

Further relations are defined between the states. We denote as $\mu_k(\mathbf{x}_i)$ the k-th fuzzy set of the i-th state and as $R_{state}(\mu(\mathbf{x}_{ni}))^j_k$ (for $k = j$, $R_{state}(\mu(\mathbf{x}_{ni}))^j$) the relation between the k-th fuzzy set of the i-th state and the j-th fuzzy set of the n-th state.

Finally we obtain that the j-th fuzzy set of the n-th state can be estimate as the union of these relations
$\mu_j(\mathbf{x}_n) =$

$$\left(\bigvee_{i=1}^{|\mathbf{ch}[n]|} \left(R_{state}(\mu(\mathbf{x}_{ni}))^j t \mu_j(\mathbf{x}_i)\right)\right) \vee \left(\bigvee_{i=1}^{|\mathbf{ch}[n]|} \left(R_{edge}(\mu(\mathbf{l}_{ni}))^j t \mu_j(\mathbf{l}_{ni})\right)\right) \vee \tag{5}$$

$$\left(\bigvee_{i=1}^{K} \left(R_{label}(\mu(\mathbf{l}_n))^j t \mu_j(\mathbf{l}_n)\right)\right). \tag{6}$$

To classify the graphs a defuzzyfication stage is added at the root state (or *super-source* $\mathbf{x}_0$). The defuzzyfication is obtained with a weighted sum of the memberships of the *super-source*

$$o_i = \sum_{k=1}^{K} w_k \mu_k(\mathbf{x}_0)) \tag{7}$$

where $\mu_k(\mathbf{x}_0)$ is the k-th membership of the *super-source*. The weights w_k are determined during the learning phase. The *super-source* is the only supervised node, i.e. with target $\mathbf{t}_i$ is assigned to it.

4.1 Optimization Process

In the learning process, the parameters to be estimated are the membership functions of both labels and edges, and the relations. To estimate those parameters we use a back-propagation algorithm. The output o_i can be calculated starting from the leaves and proceedings towards up to the super-source node, whereas the error is back-propagated from the root to the leaves of the UFN model. In our experiments, the optimization is obtained by minimizing a sum-of-squares error between the output of the model $o_i(\mathbf{T})$ and the target of the graphs t_i

$$E(\mathbf{T}) = \frac{1}{2}\sum_{i=1}^{N}(o_i(\mathbf{T}) - t_i)^2 \tag{8}$$

where n is the number of patterns in the training set. The error is successively back-propagated to each node. In the following, we show how we can obtain the gradient of this error w.r.t. the parameters to learn.

The partial derivative of the global error with respect to the weights w_k of the defuzzyfication is

$$\frac{\partial E(\mathbf{T})}{\partial w_k} = \frac{\partial E(\mathbf{T})}{\partial \mu_k(\mathbf{x}_0)}\frac{\partial \mu_k(\mathbf{x}_0)}{\partial w_k} = (o^i - t^i)\mu_k(\mathbf{x}_0) \tag{9}$$

To update the other parameters we need to calculate the gradient of the error w.r.t. this parameters and successively back-propagate the error from the top to the leaves. For example, let us consider the computation of the gradient at node 1 that is a child of the root. We consider the weight $R_{label}(\mu(\mathbf{l}_1))^j$ between the fuzzy set $\mu_j(\mathbf{l}_1)$ and the fuzzy set $\mu_j(\mathbf{x}_1)$. In this case the derivative is

$$\frac{\partial E(\mathbf{T})}{\partial R_{label}(\mu(\mathbf{l}_1))^j} = \frac{\partial E(\mathbf{T})}{\partial \mu(\mathbf{x}_1)^j}\frac{\partial \mu(\mathbf{l}_1)^j}{\partial R_{label}(\mu(\mathbf{l}_1))^j} = (o^i - t^i)w_j\frac{\partial \mu(\mathbf{x}_1)^j}{\partial R_{label}(\mu(\mathbf{l}_1))^j} \tag{10}$$

The second derivative in this expression depends on the definition of the uni-norms. Membership functions are Gaussian in our case. To learn the parameters of this memberships we apply the same back-propragation approach. If we consider the following j-th Gaussian membership of the i-th attribute

$$\mu^j(\mathbf{l}_1^i) = \exp\left(\frac{(\mathbf{l}_1^i - \mathbf{m}_i^j)^2}{(\sigma_i^j)^2}\right) \tag{11}$$

where $\mathbf{m}_i^j$ and σ_i^j are the mean and standard deviation, respectively, then the partial derivatives becomes

$$\frac{\partial E(\mathbf{T})}{\partial \mathbf{m}_i^j} == (o^i - t^i)w_j R_{label}(\mu(\mathbf{l}_1))^j \frac{\mathbf{l}_1^i - \mathbf{m}_i^j}{(\sigma_i^j)^2}\mu^j(\mathbf{l}_1^i) \tag{12}$$

and

$$\frac{\partial E(\mathbf{T})}{\partial \sigma_i^j} == (o^i - t^i)w_j R_{label}(\mu(\mathbf{l}_1))^j \frac{(\mathbf{l}_1^i - \mathbf{m}_i^j)^2}{(\sigma_i^j)^3}\mu^j(\mathbf{l}_1^i) \tag{13}$$

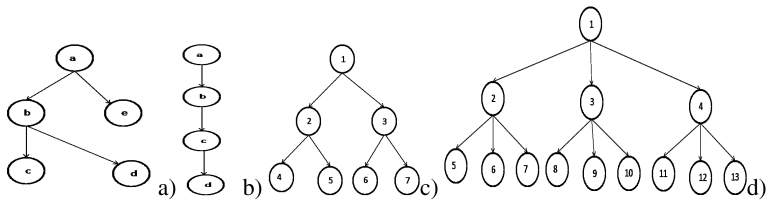

Fig. 4. Example of graphs: a) 4 nodes graph; b) 4 nodes graph; c) binary; d) ternary

5 Experimental Results

In this Section we investigate the performance of the proposed model to classify labeled trees. In a first experiment we analyze the capability of the model to discriminate two classes of trees. Examples of these trees are shown in Figures 4a and 4b, respectively. For both the graphs each node contains three attributes that identify a bi-dimensional position in the range $[0, 10]$ and a random attribute in the $[0, 1]$ interval. The objective of this experiment is to test the capability of the model to discriminate the trees using both label and positional information.

We created a data set of 500 trees (75% for the training and 25% for the test). The fuzzyfication is obtained by adopting 3 fuzzy Gaussian memberships and the composition of the uninorms is obtained by adopting the algebraic product as t-norm and the algebraic sum as t-conorm. We tested the model varying the neutral element e. We remark that the optimal performance were obtained varying e in the range $[0, 0.6]$, leading to 100% of perfect classification for training and test sets. For a neutral element higher than 0.6, in fact, the model is instable and a low rate of classification is achieved (near 70%).

In the second experiment we consider the same tree topology with the addition of edge labels. The UFN structure and parameters were set as in the previous experiment. The best performance were obtained varying e in the range $[0, 0.5]$, leading to 100% of perfect classification on training and test sets. This confirm also in this case that the uninorm composition must be OR-dominant.

In the third experiment we consider other two classes of trees. In this case the aim was to discriminate between binary and ternary trees. The parameters of the model are as in the previous experiments. In Figure 4c-d we show the topology of this trees. Each node contains 3 attributes that are randomly chosen in the interval $[0, 1]$. We generate 500 trees where the attributes are fixed but we randomly delete a node. With this experiment we wish to prove the capability of UFN to learn typologies of trees from corrupted patterns. Also in this case the best performance is obtained for OR-ness connections. We note, indeed, that varying e in the range $[0, 0.5]$ the performance is close to 100% of perfect classification on both training and test set. Moreover, also varying the attributes of the single nodes by adding a Gaussian random noise in the $[0, 0.1]$ range, the performance are comparable with that previous obtained.

6 Conclusions

In this work, a fuzzy recursive model based on uninorms to process structured data with a no flat representation has been reported. The model is based on a new category of logic connectives and logic neurons based on the concept of uninorms. Uninorms are a generalization of t-norms and t-conorms used for composing fuzzy sets. To optimize the parameters of the model we use an encoding network and a back-propagation algorithm. The learning issues are reported to classify classes of labeled trees. Reported experiments show the capability of the model to approach structured data as in the case of tRNA structures, Region Adjacent Graph (RAG), HTML or XML page classification, and so on. Ongoing work is the generalization of the model for more complex structures described by labeled graphs and the validation of the model from a theoretical and practical point of view.

References

1. Calvo, T., Mesiar, R.: Continuous Generate Associative Aggregation Operators. Fuzzy Sets and Systems 126, 191–197 (2002)
2. Bianchini, M., Maggini, M., Martinelli, E., Sarti, L., Scarselli, F.: Recursive Neural Networks for Processing Graphs with Labelled Edges: Theory and Applications. Neural Networks 18, 1040–1050 (2005)
3. Ciaramella, A., Pedrycz, W., Tagliaferri, R.: The Genetic Development of Ordinal Sums. Fuzzy Sets and Systems 151, 303–325 (2005)
4. Ciaramella, A., Pedrycz, W., Tagliaferri, R.: OR/AND Neurons for Fuzzy Set Connectives Using Ordinal Sums and Genetic Algorithms. In: Bloch, I., Petrosino, A., Tettamanzi, A.G.B. (eds.) WILF 2005. LNCS (LNAI), vol. 3849, pp. 188–194. Springer, Heidelberg (2006)
5. Ciaramella, A., Pedrycz, W., Tagliaferri, R.: The Genetic Development of Uninorms Based Neurons. In: Masulli, F., Mitra, S., Pasi, G. (eds.) WILF 2007. LNCS, vol. 4578, pp. 69–76. Springer, Heidelberg (2007)
6. Fodor, J.C., Yager, R.R., Rybalov, A.: Structure of Uninorms. Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems 5, 411–427 (1997)
7. Frasconi, P., Gori, M., Sperduti, A.: A General Framework for Adaptive Processing of Data Structures. IEEE Transaction on Neural Networks 9(5), 768–786 (1998)
8. Gori, M., Petrosino, A.: Encoding Nondeterministic Fuzzy Tree Automata Into Recursive Neural Networks. IEEE Transactions on Neural Networks 15(6), 1435–1449 (2004)
9. Hirota, K., Pedrycz, W.: OR/AND Neuron in Modeling Fuzzy Set Connectives. IEEE Transaction on Fuzzy Systems 2, 151–161 (1994)
10. Kuchler, A., Goller, C.: Inductive Learning in Symbolic Domains Using Structure-Driven Recurrent Neural Networks. In: Görz, G., Hölldobler, S. (eds.) KI 1996. LNCS, vol. 1137, pp. 183–197. Springer, Heidelberg (1996)
11. Klement, E.P., Mesiar, R., Pap, E.: Triangular Norms. Kluwer Academic Publishers, Dordrecht (2001)
12. Pedrycz, W.: Logic-Based Fuzzy Neurocomputing with Unineurons. IEEE Transaction on Fuzzy Systems 14(6), 860–873 (2006)
13. Sperduti, A., Starita, A.: Supervised Neural Networks for the Classification of Structures. IEEE Transaction on Neural Networks 8(3) (1997)
14. Yager, R.R.: Uninorms in Fuzzy Systems Modeling. Fuzzy Sets and Systems 122, 167–175 (2001)
15. Zuckerman, M.-M.: Ordinal Sum-Sets. Proceedings of Americal Mathematical Society 35(1), 242–248