

# Orthonormal Basis Lattice Neural Networks

Angelos Barmpoutis, and Gerhard X. Ritter, *Senior Member, IEEE*

**Abstract**—Lattice based neural networks are capable of resolving some difficult non-linear problems and have been successfully employed to solve real-world problems. In this paper a novel model of a lattice neural network (LNN) is presented. This new model generalizes the standard basis lattice neural network (SB-LNN) based on dendritic computing. In particular, we show how each neural dendrite can work on a different orthonormal basis than the other dendrites. We present experimental results that demonstrate superior learning performance of the new Orthonormal Basis Lattice Neural Network (OB-LNN) over SB-LNNs.

## I. INTRODUCTION

The artificial neural model which employs lattice based dendritic computation has been motivated by the fact that several researchers have proposed that dendrites, and not neurons, are the elementary computing devices of the brain, capable of implementing logical functions. [1, 2]. Inspired by the neurons of the biological brain, a lattice based neuron that possesses dendritic structures was developed and is discussed in detail in [3, 4, 5].

Several applications of LNNs have been proposed, due to their high capability of resolving some difficult non-linear problems. LNNs were employed in applications for face and object localization [6, 7], Auto-Associative memories [8, 9, 10], color images retrieval and restoration [11, 12] etc. Furthermore, various models of fuzzy lattice neural networks (FLNN) were studied in [13, 14] and some of their applications in the area of text classification and classification of structured data domains were presented in [15, 16].

Despite the high capabilities of LNNs, training dendritic networks such as Lattice Based Morphological Perceptrons with Dendritic Structure [3], results in creating huge neural networks with a large number of neurons; the size of the trained network sometimes is comparable to the size of the training data.

In this work a new model of LNNs is proposed. In this model each neural dendrite can work on a different orthonormal basis than the other dendrites. The orthonormal basis of each dendrite is chosen appropriately in order to optimize the performance of the Orthonormal Basis Lattice Neural Network (OB-LNN). OB-LNNs have some useful properties such as automatic compression of the size of the neural network and they show significantly better learning capabilities than the standard basis LNNs. Validation

experimental results in synthetic and real datasets are presented and demonstrate superior learning performance of OB-LNNs over SB-LNNs.

The rest of the paper is organized into the following sections: In section 2, we make a brief review of the LNN model. In section 3, the Orthonormal Basis Lattice Neural Network is presented. This section is divided in two parts. In the first part the model of OB-LNNs is presented. This is followed by an algorithm for training such a neural network. Finally, in section 4, validation experimental results are presented which demonstrate superior learning performance of OB-LNNs over standard basis LNNs.

## II. LATTICE NEURAL NETWORKS

The primary distinction between traditional neural networks and LNNs is the computation performed by the individual neuron. Traditional neural networks use a multiply accumulate neuron with thresholding over the ring  $(\mathcal{R}, +, \times)$  given by the formula

$$\tau_j(x) = \sum_{i=1}^n x_i w_{ij} - \theta_j \quad (1)$$

where  $\tau_j(x)$  is the total input to the  $j^{\text{th}}$  neuron,  $x_i$  are the values of the input neurons connected with the  $j^{\text{th}}$  neuron, and  $w_{ij}$  are their weights. Finally  $\theta_j$  are bias weights.

In the case of a lattice based neuron, lattice operators  $\vee$  (maximum) and  $\wedge$  (minimum) are used. These operators and the addition (+) form the rings  $(\mathcal{R} \cup -\infty, \vee, +)$  and  $(\mathcal{R} \cup \infty, \wedge, +)$ . The computation performed by a neuron is given by the formula

$$\tau_j(x) = p_j \bigvee_{i=1}^n r_{ij} (x_i + w_{ij}) \quad (2)$$

or

$$\tau_j(x) = p_j \bigwedge_{i=1}^n r_{ij} (x_i + w_{ij}) \quad (3)$$

where  $\tau_j(x)$  is the total input to the  $j^{\text{th}}$  neuron,  $x_i$  are the values of the input neurons connected with the  $j^{\text{th}}$  neuron, and  $w_{ij}$  are their weights. Parameters  $r_{ij}$  take +1 or -1 value if the  $i^{\text{th}}$  input neuron causes excitation or inhibition to the  $j^{\text{th}}$  neuron.  $p_j$  takes also +1 or -1 value if the type of the output response is excitatory or inhibitory. A more detailed presentation of the theory of lattice based morphological neurons and how their networks work can be found in [5,17].

Using the above computational framework, a lattice neural network can be constructed using layers of lattice based neurons which are connected to neurons of other

A. Barmpoutis is with the CISE Department, University of Florida, Gainesville, FL 32611 USA (e-mail: abarmpou@cise.ufl.edu).

G. X. Ritter is with the CISE Department, University of Florida, Gainesville, FL 32611 USA (e-mail: ritter@cise.ufl.edu).

layers. Each lattice based neuron consists of dendrites which connect the neuron with the previous layer's neurons. Figure 1 shows a lattice neural network with an input layer and a lattice based neuron layer. Each lattice neuron  $M_j$  consists of dendrites  $D_{ij}$ . The neurons of the input layer are connected to the next layer via the dendrites. The black and white circles denote excitatory and inhibitory connection respectively. Each dendrite can be connected with an input neuron at most twice (with one inhibitory and one excitatory connection).

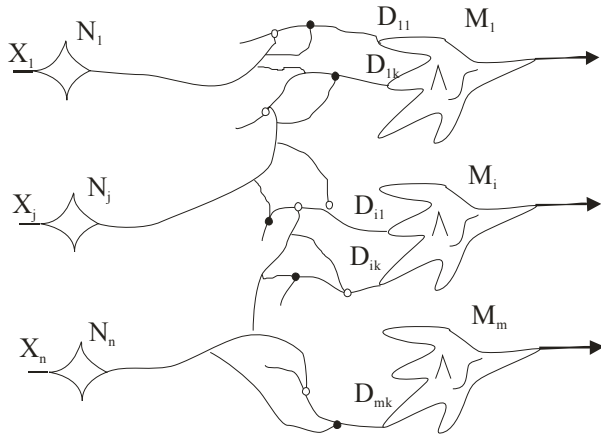


Fig. 1. An artificial neural network model which employs lattice based dendritic computations

The computation performed by a single (the  $k^{\text{th}}$ ) dendrite can be expressed using lattice operators by the formula:

$$\tau_k = p_k \bigwedge_{i \in I} \bigwedge_{l \in L} (-1)^{1-l} (x_i + w_{ik}^l) \quad (4)$$

where  $I$  is a subset of  $\{1, \dots, n\}$  which corresponds to the set of all input neurons  $N_i$  with terminal fibers that synapse on the  $k^{\text{th}}$  dendrite of the current lattice based neuron.  $L$  is a subset of  $\{1, 0\}$  and the other parameters are the same with those used in (2) and (3).

The geometrical interpretation of the computation performed by a dendrite is that every single dendrite defines a hyperbox. The borders of this hyperbox form the decision boundaries in a particular location of the input space. The left part of figure 2 shows a decision hyperbox defined by a dendrite. Its edges are parallel to the  $x$  and  $y$  axis of the input space ( $\mathcal{R}^2$ ). Due to this constraint about the orientation of the hyperboxes, the decision boundaries formed by lattice neural networks are not smooth and box patterns are annoyingly visible along the boundaries (fig.3 middle).

This hyperbox can be defined by a single dendrite via its weight values  $w_{ij}$ . Decision boundaries with more complex shapes, can be formed by using more dendrites. Furthermore boundaries which separate more than two classes can be formed, if more lattice based neurons are employed. The left part of figure 3 shows an example of decision boundaries with more complex shape, forming the letters ABC. In the middle of the same figure we can see how a group of hyperboxes (2-dimensional boxes in this case) can approximate the decision boundaries. Each box can be defined by a dendrite. Complicated figures require a large number of dendrites, in order to achieve satisfactory approximation of the decision boundaries.

Note that here the word “hyperbox” has a more general meaning, since some of the edges of the hyperboxes may have infinite length. For example in the  $\mathcal{R}^2$  domain, a straight line can be considered as a hyperbox, whose one of the edges is identical to this line, and the rest of the edges are located at  $\pm\infty$ . Following similar reasoning, two parallel lines can be also considered as a hyperbox, etc.

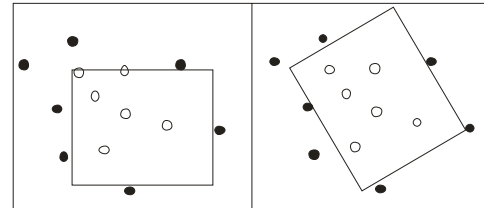


Fig. 2. Decision boundaries of a SB-LNN dendrite (left) and an OB-LNN dendrite (right).

An algorithm for training a lattice neural network with dendritic structure can be found in [17] and method for learning LNNs can be found in [4]. A comparison of various training methods for LNNs that employ dendritic computing is presented in [18]. The next section is divided into two parts. In the first part the model of Orthonormal Basis lattice neural network is presented. This is followed by an algorithm for training such a neural network.

### III. ORTHONORMAL BASIS LATTICE NNS

#### A. The OB-LNN Model

As it was discussed earlier, the geometrical interpretation of the computation performed by a dendrite is that every single dendrite defines a hyperbox in the space of input values. These hyperboxes are oriented parallel to the Cartesian axis of the space of input values. The left part of figure 2 presents a decision hyperbox defined by a dendrite. Its edges are parallel to the  $x$  and  $y$  axis of the input space ( $\mathcal{R}^2$ ). Due to this constraint about the orientation of the hyperboxes, the decision boundaries formed by lattice neural networks are not smooth and box patterns are annoyingly visible along the boundaries (fig.3 middle).

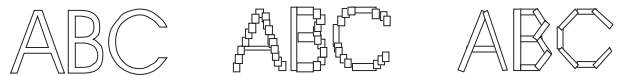


Fig. 3. This is an illustration of the decision boundaries that can be formed by a SB-LNN (middle) and an OB-LNN (right). The desired decision boundaries form the ABC letters (left).

To overcome these disadvantages, a new type of dendrite can be defined, which is able to form a hyperbox parallel to an arbitrary orthonormal system. The right part of figure 2 presents such a hyperbox, whose orientation is no longer parallel to the Cartesian axis of the input space. A neural network consisting of lattice based neurons with such dendrites would be able to produce smoother decision boundaries (fig. 3 right). In figure 3 the decision boundaries formed by such an Orthonormal Basis Lattice Neural Network are compared with these formed by a Standard Basis LNN. Notice that the number of hyperboxes (thus the

number of dendrites as well) required by the OB-LNN is much smaller than the number of those required by a SB-LNN.

Another advantage of the Orthonormal Basis LNNs is that they store information about the local orientation of the classes. This is demonstrated with an example in figure 4. Suppose that the samples of a class form the shape of the letter “A”. A neural network with Orthonormal Basis Dendrites can approximate this shape forming mainly 3 hyperboxes. The orientation of each hyperbox contains information about the local orientation of this class (fig.4 right). This useful information cannot be obtained by the standard basis LNN that was trained for the same purpose (fig. 4 left). This property is better illustrated in section 4

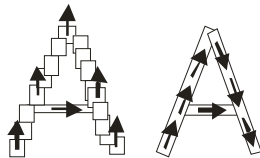


Fig. 4. Another advantage of OB-LBNN (right) is that they also store information about the local orientation of the classes.

(fig. 6), where training results of standard and orthonormal basis LNNs are presented.

The computation performed by the  $k^{\text{th}}$  Orthonormal Basis dendrite can be expressed using lattice operators, changing slightly equation (4) as follows

$$\tau_k = p_k \bigwedge_{i \in I} \bigwedge_{l \in L} (-1)^{l-1} [(R_k X)_i + w_{ik}^l] \quad (5)$$

where  $\mathbf{X}$  is the input value vector  $(x_1, x_2, \dots, x_l)^T$ ,  $\mathbf{R}_k$  is a square matrix whose columns are unit vectors forming an orthonormal basis, and  $(\mathbf{R}_k \mathbf{X})_i$  is the  $i^{\text{th}}$  element of the vector  $\mathbf{R}_k \mathbf{X}$ . Each dendrite now works in its own orthonormal basis defined by the matrix  $\mathbf{R}_k$ . The weights of the dendrite act on the elements of vector  $\mathbf{R}_k \mathbf{X}$ , hence the weights act on the rotated by the orthonormal basis  $\mathbf{R}_k$  space.

Note that Standard Basis Lattice Neural Networks are a sub group of the Orthonormal Basis Lattice Neural Networks where the matrix  $\mathbf{R}_k$  is the identity matrix. In this case it is obvious that  $(\mathbf{R}_k \mathbf{X})_i = x_i$ , thus equation (5) becomes equal to equation (4).

### B. Training OB-LNNs

The training of an Orthonormal Basis Lattice Neural Network is based on finding the best possible values for the weights and  $\mathbf{R}_k$  matrices. In other words, in order to train an OB-LNN, one must train its dendrites.

The training methods which can be used for training SB-LNNs can also be used for training OB-LNNs with an appropriate modification in order to adopt the fact that each dendrite works on its own orthonormal basis. The training procedure of an orthonormal basis dendrite can be treated as a maximization problem. The quantity that we are trying to maximize is the volume of the hyperbox which is defined by the dendrite.

By fixing the matrix  $\mathbf{R}_k$ , the weights of the dendrite can be

estimated by the training procedure described in [17]. The weights define the hyperbox; therefore its volume can be directly calculated from the weights. We can repeat the process by varying appropriately the matrix  $\mathbf{R}_k$ , until the volume of the hyperbox reaches a maximum.

The matrix  $\mathbf{R}_k$  is a rotation matrix, i.e. it rotates the vector  $\mathbf{X}$ . A variation of this matrix  $d\mathbf{R}$  is also a rotation matrix. A new matrix  $\mathbf{R}_k'$  can be obtained by multiplying  $\mathbf{R}_k$  with matrix  $d\mathbf{R}$ . ( $\mathbf{R}_k' = \mathbf{R}_k d\mathbf{R}$ ). A variation matrix  $d\mathbf{R}$  can be easily constructed by using the following equation:

$$d\mathbf{R} = e^{t\mathbf{S}} \quad (6)$$

where  $\mathbf{S}$  is a randomly generated skew symmetric matrix, and  $t$  is a scalar value. Note that the exponential is the matrix exponential. Note also that the matrix exponential of a skew symmetric matrix is always a rotation matrix. The smaller the absolute value of  $t$  is, the smaller the variation, which is caused by the matrix  $d\mathbf{R}$ , is.

By using the above, any cost minimization method can be used in order to minimize the negative of the hyperbox's volume (or to maximize its volume) in steps 1 and 3 of algorithm 1. Simulated annealing [19] and greedy searching for experiments in 2D are the methods used in the experiments presented in section 4, in order to find the matrix  $\mathbf{R}_k$  that gives the maximum hyperboxes' volume.

The training algorithm of an OB-LNN morphological perceptron is summarized below. This algorithm is an extension, in the space of OB-LNN, of the training algorithm for morphological perceptron proposed in [17]. The algorithm is presented for the case of 2 classes only, but it can be easily extended to problems with a larger number of classes. A more detailed description of the training algorithm in the case of SB-LNN is presented in [17].

The time complexity of this algorithm can be easily

#### ALGORITHM 1

**Input:** N training samples  $\mathbf{X}_i$ , and N outputs  $d_i = 0$  or 1 for class C1 or C2 respectively.  $i=1, \dots, N$

**Output:** The number of generated dendrites L, their weights  $\mathbf{W}_j$  and their orthonormal basis  $\mathbf{R}_j$ ,  $j=1, \dots, L$

Step 1: Find the smallest possible hyperbox containing all the samples of C1, and assign the appropriate values  $\mathbf{W}_1$  and  $\mathbf{R}_1$  to the first dendrite. Set  $L=1$ .

Step 2: If there are misclassified points of C2, pick arbitrarily a misclassified point  $\xi$  and go to step 3.

Step 3: Find the biggest hyperbox that contains  $\xi$ , but it does not contain any point of C1. Set  $L=L+1$ . Assign the appropriate values to  $\mathbf{W}_L$  and  $\mathbf{R}_L$ . Go to step 2.

proved that is equal to  $O_{OBLNN} = O_{SBLNN} \cdot O_{Min}$  where  $O_{SBLNN}$  is the time complexity of the training algorithm in the case of SB-LNN and  $O_{Min}$  is the time complexity of the

minimization process used in steps 1 and 3. Therefore, in order to obtain improved learning capability we loose in speed performance. The improvement in speed performance will be one of the research topics in our future work.

#### IV. EXPERIMENTAL RESULTS

In this section, validation experimental results are presented which demonstrate superior learning performance of OB-LNNs over SB-LNNs. The experiments were performed using synthetic 2-dimensional datasets and the well known Iris flower dataset.

The first dataset was synthetically generated and it forms two 2D spirals. The points of the two spirals obey the equations

$$\begin{aligned} [x_1(\vartheta), y_1(\vartheta)] &= [\vartheta \cos(\vartheta)2/\pi, \vartheta \sin(\vartheta)2/\pi] \quad \text{and} \\ [x_2(\vartheta), y_2(\vartheta)] &= [-x_1(\vartheta), -y_1(\vartheta)] \end{aligned} \quad (7)$$

respectively. Several versions of this datasets were generated with a) 130, b) 258, c) 514, d) 770, e) 1026, f) 1538 samples. The largest dataset (2538 samples) was used for the testing dataset. The samples of the smallest dataset (130 samples) are presented in figure 5. Small circles denote the samples of the one spiral, and small crosses denote the samples of the other one.

Two neural networks were trained in the previously described datasets: a) a Standard Basis Lattice Neural Network and b) the proposed Orthonormal Basis Lattice Neural Network. Both networks were lattice based morphological perceptrons with dendritic structure [17] so that their performance could be compared directly. In the case of the OB-LNN perceptron, the dendrites were Orthonormal Basis dendrites, which were trained in order to maximize the volume of the hyperboxes that they formed.

Table 1 presents the classification errors of the trained neural networks for different sizes of the training datasets (column 1). The size of the testing dataset for all the experiments was 1538 samples. In all cases, the classification errors made by the OB-LNN are significantly

TABLE I  
CLASSIFICATION ERRORS AND NUMBER OF DENDRITES NEEDED

TRAINING SAMPLES	OB-LNN		SB-LNN		(1 - RATIO) OF ERRORS
	DNDR	ERR	DNDR	ERR	
130	12	14.0%	14	18.34%	23.45%
258	14	6.5%	17	9.69%	32.20%
514	15	3.2%	19	4.55%	28.57%
770	16	2.3%	19	3.20%	26.88%
1026	15	1.5%	20	2.54%	40.94%

This table presents classification errors and number of dendrites needed for the training of an OB-LNN and a SB-LNN, using different sizes of training samples (see column 1). The 2<sup>nd</sup> and 4<sup>th</sup> columns present the number of dendrites needed for the correct classification of the training samples. The 3<sup>rd</sup> and 5<sup>th</sup> columns show the percentage of misclassified samples using always 1538 testing samples. The last column shows the quantity equals to one minus the ratio of the 3<sup>rd</sup> column over the 5<sup>th</sup> column.

smaller than these made by the SB-LNN. This conclusively demonstrates superior training performance of the proposed neural network over the standard basis lattice neural

networks.

Furthermore, the number of the dendrites, which are required to form the decision boundaries between some populations, measures the learning ability of a neural network. Table 1 also presents the final number of dendrites required by the lattice neural networks in order to classify correctly all the training samples (columns 2 and 4). In all cases, the number of dendrites trained by the OB-LNN is smaller than the number of dendrites trained by a standard basis LNN. This means that the OB-LNN compresses

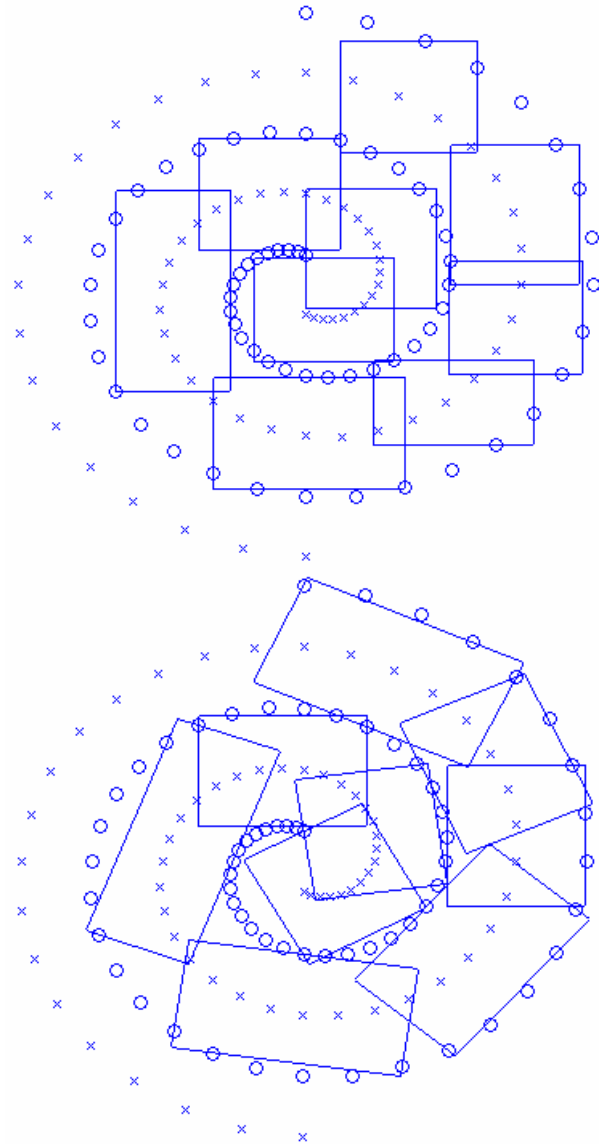


Fig. 5. This figure shows some of the hyperboxes formed by a SB-LNN (top) and a OB-LNN(bottom). The 130 training samples are denoted by circles and crosses and they are forming two spirals. The hyperboxes presented here for comparison, are the 9 smallest of each case.

automatically its size.

Figure 5 shows most of the hyperboxes formed by the dendrites of the Standard basis (top) and the Orthonormal basis LNNs (bottom). These hyperboxes were formed by the training process using 130 training samples. By observing

this figure we can see the differences between the decision boundaries formed by the OB-LNN and those formed by the SB-LNN. The hyperboxes generated by the Orthonormal basis LNN have bigger volume (area in the 2D domain) than those generated by the standard basis LNN. In the case of OB-LNN each hyperbox is rotated appropriately because of the fact that it is working on a different orthonormal basis than the other dendrites.

As it was discussed earlier in section 3, each hyperbox contains information about the local orientation of the classes. This property is illustrated in figure 6. In this figure the decision boundaries between the two spirals generated by a SB-LNN (top) and an OB-LNN (bottom) are presented. On each hyperbox several ellipses are plotted. The sizes of the principal axes of each ellipse are proportional to the size (length and width) of the relative hyperbox. The dominant axis of each ellipse is also plotted, forming a vector field.

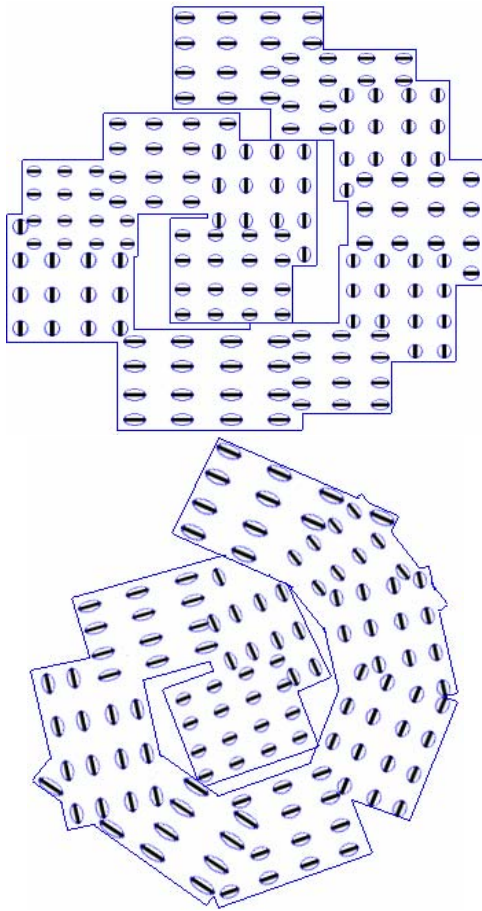


Fig. 6. Decision boundaries formed by a SB-LNN (top) and a OB-LNN (bottom). On each hyperbox several ellipses are plotted. The sizes of the principal axes of each ellipse are proportional to the sizes (length and width) of the relative hyperbox.

Observing the vector field generated by the SB-LNN (top) and this generated by the OB-LNN (bottom), one can conclude that the hyperboxes of the OB-LNN contain information about the local orientation of the classes. In the case of SB-LNN this property cannot be generally observed.

Another set of experiments was held by using the Iris

flower dataset. The Iris Flower Dataset is a popular multivariate dataset that was introduced by R.A. Fisher as an example for discriminant analysis. The data reports on four

TABLE II  
COMPARISON OF CLASSIFICATION ERRORS IN IRIS FLOWER DATA

TRAINING SAMPLES	OB-LNN	SB-LNN	PERCEPTRON
50%	8.33%	10.67%	13.33%
60%	6.41%	10.00%	12.21%
70%	4.20%	6.33%	7.34%
80%	3.12%	3.67%	6.54%
90%	2.01%	3.33%	6.38%
100%	0%	0%	3.67%

This table presents classification errors of the three neural networks for different amounts of training samples.

characteristics of the three species of the Iris Flower, sepal length, sepal width, petal length, and petal width. The goal of a discriminant analysis is to produce a simple function that, given the four measurements, will classify a flower correctly.

Several experiments were performed by using randomly different amounts of the Iris flower samples as testing samples (1<sup>st</sup> column of Table 2). The following three neural networks were trained: a) an OB-LNN perceptron, b) a SB-LNN perceptron and c) a multilayer perceptron (MLP) with one hidden layer. Several different architectures were used for the MLP with 1 hidden layer, and the results of table 2 are the best obtained. The whole Iris data set were used as the testing data set. The classification errors of the three neural networks are presented in table 2. In all cases, the classification errors made by the OB-LNN are significantly smaller.

Finally, another experiment was also held in order to compare the sizes of the trained neural networks. A synthetic dataset was generated forming two classes; one within an ellipse (fig. 7 left) and the other one outside of it. The sample points of the two classes were picked up randomly using uniform distribution.

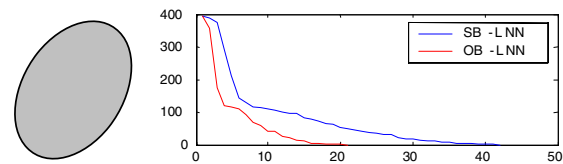


Fig. 7. Left: This dataset forms an ellipse. Right: Plot of the misclassified points over the number of dendrites required by a OB-LNN and a SB-LNN during the training process.

The same two lattice neural networks with dendritic structure were used: a) an OB-LNN perceptron and b) a SB-LNN perceptron. The right plate of figure 7 shows the plot of the number of misclassified sample points over the number of dendrites generated by the two neural networks during the training process. The final number of dendrites required by the OB-LNN in order to classify correctly all the training samples is significantly smaller than the number of dendrites trained by a standard basis LNN. This also conclusively demonstrates superior learning performance of OB-LNNs over SB-LNNs.

## V. CONCLUSION

In this paper, a novel model of Lattice Neural Networks, called Orthonormal Basis Lattice Neural Network, was presented. Comparisons of the proposed model of neural networks with the standard basis model of Lattice neural networks were presented. Validation experimental results were also presented demonstrating the advantages of the proposed model. Our future work will be focused on applying this model in applications for face and object localization, Auto-Associative memories and color images retrieval and restoration, in which areas the standard basis lattice neural networks have been applied [6-12]. Extension of the proposed model in the area of Fuzzy Lattice Neurocomputing [13, 14] and improvement in speed performance will also be some of our future research topics.

## REFERENCES

- [1] Eccles, J. *The Understanding of the Brain*, McGraw-Hill, New York, 1977.
- [2] Segev, I., Dendritic Processing, In *The Handbook of Brain Theory and Neural Networks*, M.A. Arbib, Ed., MIT Press, Boston, 1988, pp. 282-289
- [3] Ritter, G. and Sussner P. "An Introduction to Morphological Neural Networks". In *Proceedings of the 13<sup>th</sup> International Conference on Pattern Recognition*, (25-29 August, 1996), vol. 4, pp. 709-717.
- [4] Sussner, P. "Morphological Perceptron Learning". In *Proceedings of the IEEE ISIC/CIRA/ISAS Joint Conference* (Gaithersburg, MD, 14-17 September, 1998), pp. 477-482
- [5] Ritter, G., and Urcid, G. "Lattice algebra approach to single-neuron computation". In *IEEE Transactions on Neural Networks*, vol. 14, no. 2, pp. 282 – 295, March 2003.
- [6] Grana, M., and Raducanu, B. "Some applications of morphological neural networks". In *Proceedings of the International Joint Conference on Neural Networks*, (15-19 July 2001). vol. 4, pp. 2518-2523.
- [7] Raducanu, B., and Grana, M. "Morphological Neural Networks for Vision Based Self-Localization". In *Proceedings of the IEEE International Conference on Robotics and Automation*, (2001 ICRA), vol. 2, pp. 2059-2064.
- [8] Ritter, G. X., and Iancu, L. "A Morphological Auto-Associative Memory based on Dendritic Computing". In *Proceedings of IEEE International Joint Conference on Neural Network*. (25-29 July 2004), vol. 2, pp. 915-920.
- [9] Sussner, P. "Binary Autoassociative Morphological Memories Derived from the Kernel Method and the Dual Kernel Method". In *Proceedings of the International Joint Conference on Neural Networks*, (20-24 July 2004), vol. 1, pp. 236-241.
- [10] Sussner, P. "A Fuzzy Autoassociative Morphological Memory". In *Proceedings of the International Joint Conference on Neural Networks*, (20-24 July 2004), vol. 1, pp. 326-331.
- [11] Yun, Z., Ling, Z., and Yimin, Y. "Using Multi-Layer Morphological Neural Network for color images retrieval". In *Proceedings of the 5<sup>th</sup> World Congress on Intelligent Control and Automation*, (Hangzhou, P.R. China, June 15-19, 2004), vol. 5, pp. 4117-4119.
- [12] Zhang, L., Zhang, Y. and Yang, Y. "Color images restoration with Multi-Layer Morphological (MLM) Neural Network". In *Proceedings of the Second International Conference on Machine Learning and Cybernetics*, (2-5 November 2003) pp. 2831-2834
- [13] Petridis, V. and Kaburlasos, V. "Fuzzy lattice neural network (FLNN): a hybrid model for learning", In *IEEE Transactions on Neural Networks*, vol. 9, no. 5, pp. 877-890 Sept. 1998.
- [14] Kaburlasos V, and Petridis V. "Fuzzy Lattice Neurocomputing (FLN) models", *Neural Networks*, vol. 13, no. 10, pp. 1145-1170.
- [15] Petridis, V., and Kaburlasos, V. "Clustering and classification in structured data domains using Fuzzy Lattice Neurocomputing (FLN)", In *IEEE Transactions on Knowledge and Data Engineering*, vol. 13, no. 2, pp. 245 – 260, March-April 2001
- [16] Petridis, V., Kaburlasos, V., Fragkou, P., and Kehagias, A. "Text classification using the  $\sigma$ -FLNMAP neural network", In *Proceedings of International Joint Conference on Neural Networks*, (15-19 July 2001), vol. 2, pp. 1362 – 1367
- [17] Ritter, G. X., Iancu, L., and Urcid, G. "Morphological Perceptrons with Dendritic Structure". In *Proceedings of the IEEE International Conference on Fuzzy Systems*. (25-28 May 2003) vol. 2, pp. 1296-1301.
- [18] Ritter G., and Schmalz M. "Learning in lattice neural networks that employ dendritic computing", In *Proceedings of IEEE World Congress on Computational Intelligence*, (16-21 July 2006), vol. ,pp.
- [19] Saul A. Teukolsky, S., Flannery, B., Press, W., and Vetterling, W., *Numerical Recipes Example Book (C++)*, Cambridge University Press, 2003.