Morphological Perceptrons with Dendritic Structure

Gerhard X. Ritter
CISE Dept., Univ. of Florida
Gainesville, FL 32611-6120
Email: ritter@cise.ufl.edu

Laurentiu Iancu
CISE Dept., Univ. of Florida
Gainesville, FL 32611-6120
Email: liancu@cise.ufl.edu

Gonzalo Urcid
Optics Department, INAOE
Tonantzintla, Pue. 72000, Mexico
Email: gurcid@inaoep.mx

Abstract— **Recent advances in neurobiology and the biophysics of neural computation have brought to the foreground the importance of dendritic structures of neurons. These structures are now viewed as the primary basic computational units of the neuron, capable of realizing logical operations. Based on these new biophysical neural models, we develop a new paradigm for single layer perceptrons that incorporates dendritic processes. The basic computational processes in dendrites as well as neurons are based on lattice algebra. The computational capabilities of this new perceptron model is demonstrated by means of several illustrative examples and two theorems.**

I. Introduction

The number of synapses on a *single* neuron in the cerebral cortex ranges from 500 to 200,000. Most of the synapses occur on the dendritic tree of the neuron, and it is here where information is processed [18], [4], [19], [8]. Dendrites make up the largest component in both surface area and volume of the brain. Part of this is due to the fact that pyramidal cell dendrites span all cortical layers in all regions of the cerebral cortex [1], [4], [18]. Thus, when attempting to model artificial brain networks that bear more than just a passing resemblance to biological brain networks, one cannot ignore dendrites (and their associated spines) which can make up more than 50% of the neuron's membrane. This is especially true in light of the fact that some brain researchers have proposed that dendrites and not the neuron are the elementary computing devices of the brain. Neurons with dendrites can function as many, almost independent, functional subunits with each unit being able to implement a rich repertoire of logical operations [22], [5], [18], [4], [8]. Possible mechanisms for dendritic computation of such logical functions as XOR, AND and NOT have been proposed by several researchers [18], [4], [8], [6], [3], [20]. In a recent paper we used these proposed mechanisms in order to construct a neuron with dendritic structure and proved its computational superiority over neurons commonly used in artificial neural networks [16]. In this manuscript, we propose a novel single layer perceptron for solving multi-class pattern recognition problems. The output neurons of the proposed perceptron are the neurons described in [16] and are discussed in Section II. In Section III we provide several examples in order to illustrate the computational processes and capabilities of the morphological perceptron with dendritic structure. Section IV establishes the core mathematical results of the computational capability of a single layer morhological perceptron, namely that any finite collection of m disjoint compact subsets of n-dimensional Euclidean space $\mathbb{R}^n$ can be classified as m distinct classes to within any desired degree of accuracy. We conclude this paper with several pertinent observations concerning training of morphological perceptrons and differences between morphological perceptrons and classical perceptrons.

II. Morphological Neurons and Morphological Perceptrons

Artificial neural networks whose computation is based on lattice algebra have become known as *morphological neural networks* (MNNs). The primary distinction between traditional neural networks and MNNs is the computation performed by the individual neuron. Traditional neural networks use a multiply accumulate neuron with thresholding over the ring $(\mathbb{R}, +, \times)$ given by the formula

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

where $\mathbf{x} \in \mathbb{R}^n$, x_i denotes the value of the ith neuron, w_{ij} denotes the synaptic strength between the ith neuron and the jth neuron, θ_j the threshold of the jth neuron, and τ_j the total input to the jth neuron. Morphological neural networks use lattice operations $\vee$ (maximum), or $\wedge$ (minimum), and $+$ from the semirings $(\mathbb{R}_{-\infty}, \vee, +)$ or $(\mathbb{R}_{\infty}, \wedge, +)$, where $\mathbb{R}_{-\infty} = \mathbb{R} \cup \{-\infty\}$ and $\mathbb{R}_{\infty} = \mathbb{R} \cup \{\infty\}$. The computation at a neuron in an MNN for input $\mathbf{x} = (x_1, \ldots, x_n)$ is given by

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

or

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

where $r_{ij} = \pm 1$ denotes whether the ith neuron causes excitation or inhibition on the jth neuron, and $p_j = \pm 1$ denotes the output response (excitation or inhibition) of the jth neuron to the neurons whose axons contact the jth neuron. For excitatory input and output action, $r_{ij} = 1$ and $p_j = 1$, respectively. Inhibitory input and output responses have the values $r_{ij} = -1$ and $p_j = -1$, respectively. The computational model for a neuron in an MNN that uses the maximum operator is illustrated in Figure 1. The activation function $f(\tau_j)$ in an MNN is generally a hard limiter.

There are several advantages for replacing the arithmetic operations used in traditional neural computing with lattice based

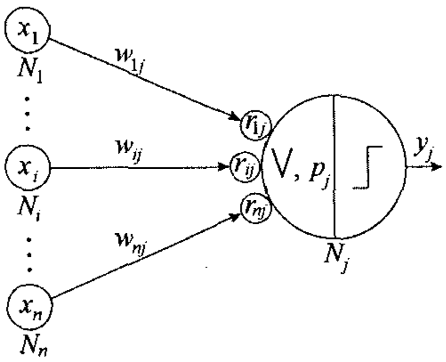

Fig. 1. Computational model of a morphological neuron.

operations. It is apparent from Eqs. (2) and (3) that morphological neural computation does not involve multiplications, but only the operations of OR or AND, addition, and subtraction. This provides for extremely fast neural computation and easy hardware implementation. Convergence problems and lengthy training algorithms are often nonexistent [11], [13]. It has been shown that morphological associative memories are extremely robust in the presence of noise and have unlimited storage capacity [10], [12], [14], [21], [15]. There is a close, natural connection between lattice based computing and fuzzy set theory making lattice based networks amenable to handling more general data types and particular types of learning models [7]. Finally, morphological neural networks are capable of solving any conventional computational problem and, thus, can be inherently useful in a wide variety of application domains [9].

The simple morphological neural model embodied in Eqs. (2) and (3) and Figure 1 lacks dendritic structures. The proposed perceptron extends this model by providing input neurons with axonal trees and output neurons with dendritic structures. A *single layer morphological perceptron* (SLMP) is a single layer, feedforward neural network consisting of n input neurons and m output neurons. Thus, in contrast to the standard model of a single layer perceptron (SLP), the output neurons of an SLMP have dendritic structures and the basic computation in the dendrites and neuron are morphological operations based on lattice algebra. The value of an input neuron N_i $(i = 1, \ldots, n)$ propagates through its axonal tree all the way to the terminal branches that make contact at synaptic sites of the dendrites of the output neurons. The weight of an axonal branch of neuron N_i terminating on the kth dendrite of an output neuron M_j $(j = 1, \ldots, m)$ is denoted by w_{ijk}^{ℓ}, where the superscript $\ell \in \{0, 1\}$ distinguishes between *excitatory* $(\ell = 1)$ and *inhibitory* $(\ell = 0)$ input to the dendrite. The kth dendrite of M_j will respond to the total input received from the input neurons and will either accept or inhibit the received input. The computation performed by the kth dendrite is given by

$$\tau_k^j = p_{jk} \bigwedge_{i \in I} \bigwedge_{\ell \in L} (-1)^{1-\ell}(x_i + w_{ijk}^{\ell}), \tag{4}$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n$ denotes the input value of the input neurons, $I \subseteq \{1, \ldots, n\}$ corresponds to the set of all input neurons N_i with terminal fibers that synapse on the kth dendrite of M_j, $L \subseteq \{0, 1\}$ corresponds to the set of terminal fibers of N_i that synapse on the kth dendrite of M_j, and $p_{jk} \in \{-1, 1\}$ denotes the excitatory $(p_{jk} = 1)$ or inhibitory $(p_{jk} = -1)$ response of the kth dendrite of M_j to the received input. As observed in [16], $I \neq \emptyset$ and $L \neq \emptyset$ as there exists at least one axonal fiber from one of the input neurons which has a synapse on the kth dendrite. The input response of branch fiber w_{ijk}^{ℓ} is represented by $r_{ijk}^{\ell} = (-1)^{1-\ell} = \pm 1$, where $r_{ijk}^{1} = +1$ denotes excitation and $r_{ijk}^{0} = -1$ denotes inhibition.

The value $\tau_k^j(\mathbf{x})$ is passed to the cell body and the state of M_j is a function of the input received from all its dendrites. The total value received by M_j is given by

$$\tau^j(\mathbf{x}) = \bigwedge_{k=1}^{K_j} \tau_k^j(\mathbf{x}), \tag{5}$$

where K_j denotes the total number of dendrites (synaptic regions) of M_j. The *next* state of M_j is then determined by an activation function f, namely

$$y_j = f(\tau^j(\mathbf{x})). \tag{6}$$

In this exposition we restrict our discussion to the hard limiter

$$f(\tau^j(\mathbf{x})) = \begin{cases} 1 & \text{if } \tau^j(\mathbf{x}) \geq 0 \\ 0 & \text{if } \tau^j(\mathbf{x}) < 0 \end{cases}. \tag{7}$$

The total state computation of M_j is therefore given by

$$y_j(\mathbf{x}) = f\left[\bigwedge_{k=1}^{K_j}\left(p_{jk} \bigwedge_{i \in I} \bigwedge_{\ell \in L} (-1)^{1-\ell}\left(x_i + w_{ijk}^{\ell}\right)\right)\right]. \tag{8}$$

Figure 2 provides a graphical representation of this model.

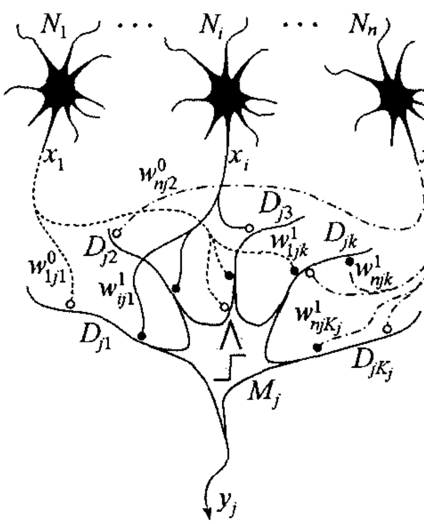

Fig. 2. Morphological perceptron with dendritic structure. Here D_{jk} denotes dendrite k of M_j and K_j its number of dendrites. Neuron N_i can synapse D_{jk} with excitatory or inhibitory fibers, e.g. weights w_{1jk}^1 and w_{nj2}^0 denote respective excitatory and inhibitory fibers from N_1 to D_{jk} and N_n to D_{j2}.

III. Examples

Having defined the computational model of a single layer morphological perceptron, it will be instructive to provide a few examples in order to illustrate the computational processes and capabilities of an SLMP.

Example 1

The simplest case occurs when the single layer morphological perceptron consists of just one input neuron, N, and one output neuron, M. Here the notation can be simplified by descarding subscript j (as being $j = 1$). Figure 3 illustrates such an SLMP.

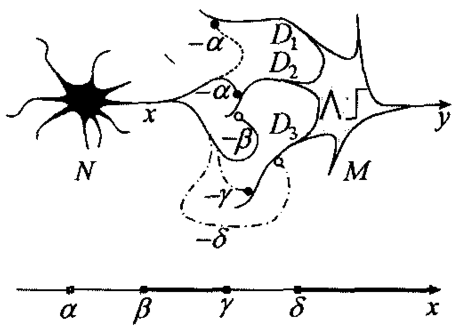

Fig. 3. The output neuron M will fire ($y = 1$) for input values from the set $X = \{\alpha\} \cup [\beta, \gamma] \cup [\delta, \infty)$. If $x \in \mathbb{R} \setminus X$, then $y = 0$.

The axonal branch of neuron N synapses on dendrites D_1, D_2, and D_3 of neuron M. The corresponding terminal axonal branching weights, and input and output responses are given in Table I. For algebraic consistency as well as numerical computation when using Eq. (8), *unused* terminal fibers with a hypothetical excitatory or inhibitory input will be assigned a weight of $+\infty$ or $-\infty$, respectively. Also, as explained earlier, $r^1_{1k} = +1$ and $r^0_{1k} = -1$ for all $k \in \{1, \ldots, K\}$. Hence, in subsequent examples, we will omit these columns.

TABLE I
Weights and Synaptic Responses, Ex. 1

D_k	w^1_{1k}	r^1_{1k}	w^0_{1k}	r^0_{1k}	p_k
D_1	$-\alpha$	$+1$	$-\infty$	-1	$+1$
D_2	$-\alpha$	$+1$	$-\beta$	-1	-1
D_3	$-\gamma$	$+1$	$-\delta$	-1	-1

If $\alpha < \beta < \gamma < \delta$, the neuron will fire (state $y = f(\tau(x)) = 1$) when receiving inputs $x = \alpha$ or any value from the intervals $[\beta, \gamma]$ and $[\delta, \infty)$. The neuron will not fire (state $y = f(\tau(x)) = 0$) for all other values from $\mathbb{R}$. Applying Eq. (5), the total value received by neuron M is, in this case:

$$\begin{aligned} \tau(x) &= +[+(x-\alpha)] \\ &\quad \wedge -[+(x-\alpha) \wedge -(x-\beta)] \\ &\quad \wedge -[+(x-\gamma) \wedge -(x-\delta)]. \end{aligned} \tag{9}$$

The neuron will fire when

$$\begin{aligned} \tau(x) \geq 0 &\iff x - \alpha \geq 0 \text{ and } (x-\alpha) \wedge (\beta - x) \geq 0 \\ &\qquad \text{and } (x-\gamma) \wedge (\delta - x) \geq 0 \\ &\iff x \geq \alpha \text{ and } (x \leq \alpha \text{ or } x \geq \beta) \\ &\qquad \text{and } (x \leq \gamma \text{ or } x \geq \delta) \\ &\iff x \in \{\alpha\} \cup [\beta, \gamma] \cup [\delta, \infty), \end{aligned} \tag{10}$$

as depicted on the axis at the bottom of Figure 3.

Example 2

A single layer morphological perceptron with two input neurons, N_1 and N_2, and two output neurons, M_1 and M_2, can be used to solve the XOR problem, formulated as a two-class problem. The SLMP illustrated in Figure 4 classifies all points $\mathbf{x} \in \mathbb{R}^2$ as belonging to class C_1 if $\mathbf{x} \in \{(0,1), (1,0)\}$ and, respectively, to class C_2 if $\mathbf{x} \in \{(0,0), (1,1)\}$. All remaining points are classified as not belonging to either of these two classes. As in classical perceptron theory, solving this problem requires two output neurons. However, in contrast to classical perceptron theory, no hidden layer is necessary for the morphological perceptron to solve the problem.

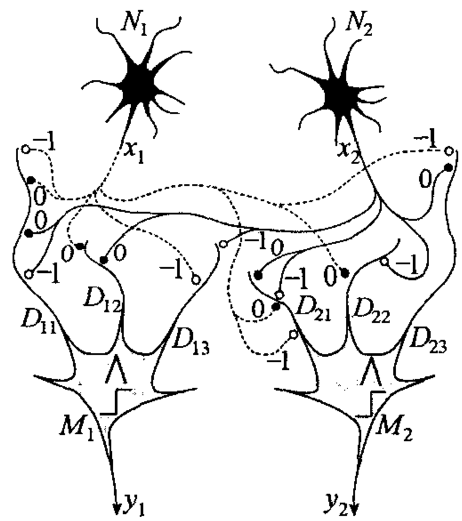

Fig. 4. Single layer morphological perceptron that solves the two-class XOR problem for points from the domain $\mathbf{x} = (x_1, x_2) \in \mathbb{R}^2$.

If $\mathbf{y} = (y_1, y_2)$, where y_j denotes the output signal of neuron M_j ($j = 1, 2$), then the desired network output is:

$$\mathbf{y} = \begin{cases} (1,0) \text{ if } \mathbf{x} \in C_1 \\ (0,1) \text{ if } \mathbf{x} \in C_2 \\ (0,0) \text{ if } \mathbf{x} \in \mathbb{R}^2 \setminus (C_1 \cup C_2). \end{cases} \tag{11}$$

Here, $\mathbf{y} = \left(f\left(\tau^1(\mathbf{x})\right), f\left(\tau^2(\mathbf{x})\right)\right)$, $\tau^j(\mathbf{x}) = \bigwedge_{k=1}^{K_j} \tau^j_k(\mathbf{x})$, denotes the computation performed by M_j, and K_j denotes the number of dendrites of M_j. The values of the axonal branch weights w^ℓ_{ijk} and output responses p_{jk} are specified in Table II.

TABLE II
TWO-CLASS XOR NETWORK PARAMETERS, EX. 2

D_{jk}	w^1_{1jk}	w^0_{1jk}	w^1_{2jk}	w^0_{2jk}	p_{jk}
D_{11}	0	-1	0	-1	$+1$
D_{12}	0	$-\infty$	0	$-\infty$	-1
D_{13}	$+\infty$	-1	$+\infty$	-1	-1
D_{21}	0	-1	0	-1	$+1$
D_{22}	0	$-\infty$	$+\infty$	-1	-1
D_{23}	$+\infty$	-1	0	$-\infty$	-1

Example 3

As noted before, a single layer morphological perceptron is capable of solving any conventional computational problem. A 3-dimensional version of the two-class XOR problem can thus be solved with an SLMP having three input neurons and two output neurons.

Let $X \subset \mathbb{R}^3$ be the set $\{(x_1, x_2, x_3) : x_i \in \{0,1\},\ i = 1,2,3\}$, which consists of all 8 binary triples, i.e., $X = \{(0,0,0),(0,0,1),\ldots,(1,1,1)\}$. Set X is divided in two classes, based on the parity of the triples, i.e. on the result of the XOR of x_1, x_2, and x_3. The class of pattern $\mathbf{x}^\xi$ for $\xi = 1,\ldots,8$ is C_1 if $x_1^\xi \oplus x_2^\xi \oplus x_3^\xi = 1$ (odd parity), and C_2 if $x_1^\xi \oplus x_2^\xi \oplus x_3^\xi = 0$ (even parity). If the set X is embedded in $\mathbb{R}^3$, all points $\mathbf{x}$ not belonging to either C_1 or C_2 are assigned to class C_0 (the *rejection* class). Figure 5 shows the corresponding learning set $T = \{(\mathbf{x}^\xi, c_\xi) : \mathbf{x}^\xi \in X,\ c_\xi \in \{1,2\}\}$ for this two-class, 3-dimensional XOR recognition problem.

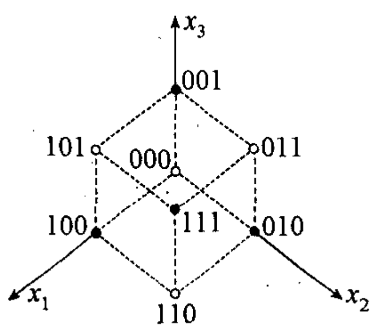

Fig. 5. Training set for the two-class 3D XOR problem. Solid bullets (•) represent patterns in class C_1; circles (∘) represent patterns in class C_2. All points $\mathbf{x} \in \mathbb{R}^3 \setminus (C_1 \cup C_2)$ will be rejected (classified into C_0).

An SLMP that solves this problem is illustrated in Figure 6. Output neurons M_1 and M_2 have five dendrites each. Their respective output signals, y_1 and y_2, encode the class a pattern $\mathbf{x} = (x_1, x_2, x_3) \in \mathbb{R}^3$ is classified to:

$$\mathbf{y} = (y_1, y_2) = \begin{cases} (1,0) & \text{if } \mathbf{x} \in C_1, \text{ i.e. } x_1 \oplus x_2 \oplus x_3 = 1 \\ (0,1) & \text{if } \mathbf{x} \in C_2, \text{ i.e. } x_1 \oplus x_2 \oplus x_3 = 0 \\ (0,0) & \text{if } \mathbf{x} \in \mathbb{R}^3 \setminus (C_1 \cup C_2). \end{cases} \tag{12}$$

The values of the axonal branch weights w^ℓ_{ijk} and output responses p_{jk}, with $i = 1,2$, $j = 1,2,3$, $k = 1,\ldots,5$, and $\ell \in \{1,0\}$, are specified in Table III.

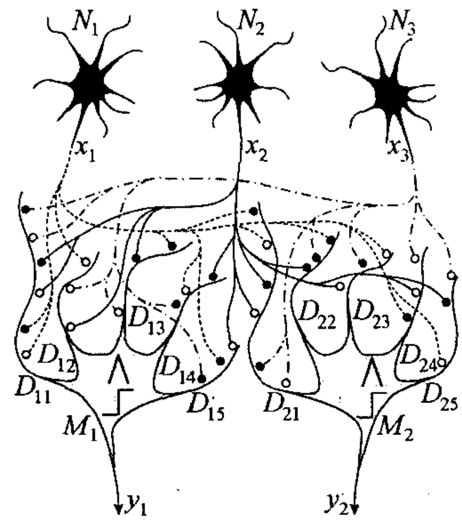

Fig. 6. Single layer morphological perceptron that solves the two-class, 3D XOR problem. All excitatory axonal fibers terminating with a solid dot (•) have a weight of value 0; all inhibitory axonal fibers terminating with a hollow circle (∘) carry a weight of value -1. The leftmost dendrite of both output neurons (dendrites D_{11} and D_{21}) are excitatory ($p_{j1} = +1$); all other dendrites are inhibitory ($p_{j2} = \cdots = p_{j5} = -1$).

TABLE III
WEIGHTS AND SYNAPTIC RESPONSES, EX. 3

D_{jk}	w^1_{1jk}	w^0_{1jk}	w^1_{2jk}	w^0_{2jk}	w^1_{3jk}	w^0_{3jk}	p_{jk}
D_{11}	0	-1	0	-1	0	-1	$+1$
D_{12}	$+\infty$	-1	$+\infty$	-1	$+\infty$	-1	-1
D_{13}	0	$-\infty$	0	$-\infty$	$+\infty$	-1	-1
D_{14}	$+\infty$	-1	0	$-\infty$	0	$-\infty$	-1
D_{15}	0	$-\infty$	$+\infty$	-1	0	$-\infty$	-1
D_{21}	0	-1	0	-1	0	-1	$+1$
D_{22}	0	$-\infty$	0	$-\infty$	0	$-\infty$	-1
D_{23}	$+\infty$	-1	$+\infty$	-1	0	$-\infty$	-1
D_{24}	0	$-\infty$	$+\infty$	-1	$+\infty$	-1	-1
D_{25}	$+\infty$	-1	0	$-\infty$	$+\infty$	-1	-1

IV. COMPUTATIONAL CAPABILITY OF A MORPHOLOGICAL PERCEPTRON

Analogous to the classical single layer perceptron (SLP) with one output neuron, a *single layer morphological perceptron* (SLMP) with one output neuron also consists of a finite number of input neurons that are connected via axonal fibers to the output neuron. However, in contrast to an SLP, the output neuron of an SLMP has a dendritic structure and performs the lattice computation embodied by Eq. (8). Figure 2 provides a pictorial representation of a general SLMP with a single output neuron. As the examples of the preceding section illustrate, the computational capability of an SLMP is vastly different from that of an SLP as well as that of classical perceptrons in general. No hidden layers were necessary to solve the XOR problem or to specify the points of the non-convex region of Figure 7. Observing differences by examples, however, does not provide answer as to the specific computational capabilities of an SLMP with one output neuron. Such an answer is given by the following two theorems.

Theorem 1

If $X \subset \mathbb{R}^n$ is compact and $\varepsilon > 0$, then there exists a single layer morphological perceptron that assigns every point of X to class C_1 and every point $\mathbf{x} \in \mathbb{R}^n$ to class C_0 whenever $\mathrm{d}(\mathbf{x}, X) > \varepsilon$.

The expression $\mathrm{d}(\mathbf{x}, X)$ in Theorem 1 refers to the distance of the point $\mathbf{x} \in \mathbb{R}^n$ to the set X. Figure 7 illustrates this concept. All points of X will be classified as belonging to class C_1 and all points outside the *banded* region of thickness ε will be classified as belonging to class C_0. Points within the banded region may be misclassified. As a consequence, any compact configuration, whether it is convex or non-convex, connected or not connected, contains a finite or infinite number of points, can be approximated to any desired degree of accuracy $\varepsilon > 0$ by an SLMP with one output neuron.

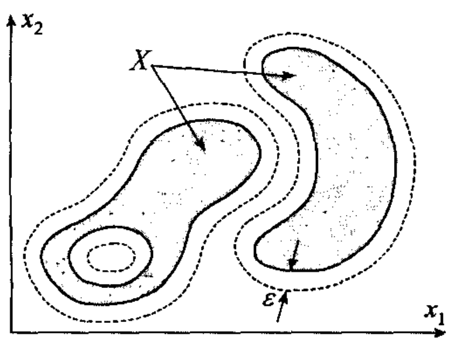

Fig. 7. The compact region X (shaded) and the banded region of thickness ε (dashed).

The proof of Theorem 1 requires tools from elementary point set topology and is given in [16]. Although the proof is an existence proof, part of it is constructive and provides the basic idea for a training algorithm.

Theorem 2 is a generalization of Theorem 1 to multiple sets. Suppose $X_1, X_2, \ldots, X_m$ denotes a collection of disjoint compact subsets of $\mathbb{R}^n$. The goal is to classify, $\forall j = 1, \ldots, m$, every point of X_j as a point belonging to class C_j and not belonging to class C_i whenever $i \neq j$. For each $p \in \{1, \ldots, m\}$, define $Y_p = \bigcup_{j=1, j\neq p}^{m} X_j$. Since each Y_p is compact and $Y_p \cap X_p = \emptyset, \varepsilon_p = \mathrm{d}(X_p, Y_p) > 0 \ \forall p = 1, \ldots, m$. Let $\varepsilon_0 = \frac{1}{2}\min\{\varepsilon_1, \ldots, \varepsilon_p\}$.

Theorem 2

If $\{X_1, X_2, \ldots, X_m\}$ is a collection of disjoint subsets of $\mathbb{R}^n$ and ε a positive number with $\varepsilon < \varepsilon_0$, then there exists a single layer morphological perceptron that assigns each point $\mathbf{x} \in \mathbb{R}^n$ to class C_j whenever $\mathbf{x} \in X_j$ and $j \in \{1, \ldots, m\}$, and to class $C_0 = \neg \bigcup_{j=1}^{m} C_j$ whenever $\mathrm{d}(\mathbf{x}, X_i) > \varepsilon$, $\forall i = 1, \ldots, m$. Furthermore, no point $\mathbf{x} \in \mathbb{R}^n$ is assigned to more than one class.

The proof of this theorem is somewhat lengthy and because of page limitation could not be included. The proof is given in [17]. Based on the proofs of these two theorems, we constructed training algorithms for SLMPs [16], [17]. During the learning phase, the output neurons grow new dendrites and input neurons expand their axonal branches to terminate on the new dendrites. The algorithms always converge and have rapid convergence rate when compared to backpropagation learning in traditional perceptrons.

V. Conclusions

We presented a new paradigm based on lattice algebra for single layer perceptrons that takes into account the dendritic processes of neurons. Theorems 1 and 2, as well as the examples presented, make it obvious that morphological perceptrons have several advantages over both traditional single layer perceptrons and perceptrons with hidden layers. For instance, morphological perceptrons need no hidden layers for solving the classical XOR problem. Also, as was shown by M. Gori and F. Scarselli [2], multilayer perceptrons are not adequate for pattern recognition and verification. In particular, Gori and Scarselli proved that multilayer perceptrons with sigmoidal units and a number of hidden units less than or equal to the number of inputs are unable to model patterns distributed in typical clusters, since these networks draw open separation surfaces in pattern space. In this case, all patterns not members of the cluster but contained in the open area determined by the surfaces will be misclassified. When using more hidden units than input units, the separation *may* be closed but, unfortunately, determining whether or not the perceptron draws closed separation surfaces in the pattern space is NP-hard. Our model does not suffer from this problem. Based on the proofs of the two theorems, we developed training algorithms which *always* draw closed regions around pattern clusters [16].

Questions may be raised as to whether dendrites merely represent hidden layers in disguise. Such questions are valid in light of the fact that, theoretically, a two hidden layer perceptron can also classify any compact region in $\mathbb{R}^n$. However, there are some major differences between the model presented here and hidden layer perceptrons. In comparison to hidden layer neurons which generally use sigmoidal activation functions, dendrites have no activation functions. They only compute the basic logic functions of AND, OR, and NOT. Activation takes place only within the neuron via the hard-limiter function. Also, with hidden layers the number of neurons within a hidden layer is predetermined before training of weights, which traditionally involves back propagation methods. In our model, dendrites are grown automatically as the neuron learns its specific task. Furthermore, *no* error remains after training. *All* pattern vectors of the training set will *always* be correctly identified after the training stops [16].

References

[1] J.C. Eccles, *The Understanding of the Brain*, McGraw-Hill, New York, 1977.

[2] M. Gori, F. Scarselli, "Are Multilayer Perceptrons Adequate for Pattern Recognition and Verifi cation?," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 20(11), 1998, pp. 1121–1132.

[3] W.R. Holmes and W. Rall, "Electronic Models of Neuron Dendrites and Single Neuron Computation" in *Single Neuron Computation*, T. McKenna, J. Davis, S.F. Zornetzer, Eds., Academic Press, San Diego, 1992, pp. 7–25.

[4] C. Koch and I. Segev, Eds. *Methods in Neuronal Modeling: From Synapses to Networks*, MIT Press, Boston, 1989.

[5] B.W. Mel in *Dendrites*, G. Stuart, N. Spruston, and M. Häusser, Eds., Oxford University Press, Oxford, England, 1999, pp. 271–289.

[6] B.W. Mel, "Synaptic integration in excitable dendritic trees," *Journal of Neurophysiology*, vol. 70, 1993, pp. 1086–1101.

[7] V. Petridis and V.G. Kaburlasos, "Fuzzy lattice neural network (FLNN): a hybrid model for learning," *IEEE Trans. Neural Networks*, vol. 9(5), September 1998, pp. 877–890.

[8] W. Rall and I. Segev, "Functional possibilities for synapses on dendrites and dendritic spines" in *Synaptic Function*, G.M. Edelman, E.E. Gall, and W.M. Cowan, Eds., Wiley, New York, 1987, pp. 605–636.

[9] G.X. Ritter and P. Sussner, "An introduction to morphological neural networks," in *Proc. 13th International Conference on Pattern Recognition*, Vienna, Austria, 1996, pp. 709–717.

[10] G.X. Ritter and P. Sussner, "Associative memories based on lattice algebra," in *IEEE Inter. Conf. Systems, Man, and Cybernetics*, Orlando, FL, October 1997, pp. 3570–3575.

[11] G.X. Ritter and P. Sussner, "Morphological perceptrons," in *ISAS '97, Intelligent Systems and Semiotics*, Gaithersburg, Maryland, 1997, pp. 221–226.

[12] G.X. Ritter, P. Sussner, and J.L. Diaz de Leon, "Morphological associative memories," *IEEE Trans. on Neural Networks*, vol. 9(2), March 1998, pp. 281–293.

[13] G.X. Ritter and T. Beavers, "An introduction to morphological perceptrons," in *ANNIE '99, Artificial Neural Networks in Engineering*, St. Louis, MO, 1999.

[14] G.X. Ritter, J.L. Diaz de Leon, and P. Sussner, "Morphological bidirectional associative memories," *Neural Networks*, vol. 12, March 1999, pp. 851–867.

[15] G.X. Ritter, G. Urcid, and L. Iancu, "Reconstruction of noisy patterns using morphological associative memories," accepted for publication in *Journal of Mathematical Imaging and Vision*.

[16] G.X. Ritter and G. Urcid, "Lattice Algebra Approach to Single Neuron Computation," accepted for publication in *IEEE Trans. on Neural Networks*.

[17] G.X. Ritter and L. Iancu, "Morphological Percpetrons," preprint submitted to *IEEE Trans. on Neural Networks*.

[18] I. Segev, "Dendritic Processing" in *The Handbook of Brain Theory and Neural Networks*, M. Arbib, Ed., MIT Press, Boston, 1998, pp. 282–289.

[19] T.J. Sejnowski and N. Qian, "Synaptic integration by electro-diffusion in dendritic spines" in *Single Neuron Computation*, T. McKenna, J. Davis, S.F. Zornetzer, Eds., Academic Press, San Diego, 1992, pp. 117–139.

[20] G.M. Shepherd, "Canonical Neurons and their Computational Organization" in *Single Neuron Computation*, T. McKenna, J. Davis, S.F. Zornetzer, Eds., Academic Press, San Diego, 1992, pp. 27–55.

[21] P. Sussner, "Observations on morphological associative memories and the kernel method," Elsevier *Neurocomputing*, vol. 31, 2000, pp. 167–183.

[22] D.S. Wei et al., "Compartmentalized and binary behavior of terminal dendrites in hippocampal pyramidal neurons," *Science*, vol. 293(5538), 2001, pp. 2272–2275.