Constructive Morphological Neural Networks: Some Theoretical Aspects and Experimental Results in Classification

Peter Sussner and Estevão Laureano Esmi

Abstract. Morphological neural networks are rooted in mathematical morphology (MM). Several constructive learning algorithms for morphological neural networks have been proposed during the last decade. Since MM can be conducted very generally in the complete lattice setting, MNNs are closely related to other lattice-based neurocomputing models.

This paper reviews and analyzes some important types of constructive morphological neural networks including their learning algorithms from the lattice-theoretical perspective of mathematical morphology. In particular, we present an improved version of the learning algorithm for the morphological perceptron (MP). Moreover, we incorporate competitive nodes into the two variants of the MP and introduce an approach for training these models. Finally, we compare the performance of several constructive morphological models and of conventional multi-layer perceptrons in some classification problems.

1 Introduction

Mathematical Morphology (MM) is a theory that uses concepts from set theory, geometry and topology to analyze geometrical structures in an image [21, 28, 45, 44]. MM has found wide-spread applications over the entire imaging spectrum [7, 19, 20, 26, 32, 47, 48]. Morphological operators were originally developed for binary and grayscale image processing. The subsequent generalization to complete lattices

Peter Sussner
Department of Applied Mathematics, IMECC, University of Campinas,
Campinas, SP 13084 – 970
e-mail: sussner@ime.unicamp.br

Estevão Laureano Esmi
Department of Applied Mathematics, IMECC, University of Campinas,
Campinas, SP 13084 – 970
e-mail: ra050652@ime.unicamp.br

is widely accepted today as the appropriate theoretical framework for mathematical morphology [21, 42, 44]. In the complete lattice setting, there are four elementary morphological operators - namely erosion, dilation, anti-erosion, and anti-dilation - which allow for the decomposition of every mapping between complete lattices [4].

Morphological neural networks incorporate morphological operators into the artificial neural network setting. More precisely, a morphological neural network performs a morphological operation at every node. Since the concept of morphological operator is not clearly defined [21], we have suggested to formally define a MNN as an artificial neural network that performs one of the four elementary operators of MM, possibly followed by the application of an activation function, at every node [55].

Several particular morphological models and their respective training algorithms have been proposed in recent years, including morphological perceptrons (MPs) [54], morphological perceptrons with dendrites (MPDs) [38], (fuzzy) morphological associative memories [55, 50, 51, 52, 55], modular morphological neural networks [3], and morphological shared-weight and regularization neural networks [22, 25]. This paper clarifies that fuzzy lattice neural networks (FLNNs) [24] can also be viewed as MNNs. Morphological and hybrid morphological/rank/linear neural networks [30] have been successfully applied to a variety of problems such as pattern recognition [24, 46], prediction [1, 50], automatic target recognition [25], handwritten character recognition [30], control of vehicle suspension [15], self-localization, and hyperspectral image analysis [34, 17].

Although the theory of morphological neural networks (MNNs) and its applications has experienced a steady and consistent growth in the last few years [53], only a brief review and comparison of MNNs has appeared in the literature in the form of a conference paper [29]. The present article focusses on constructive MNNs which automatically update their architecture during the learning phase. Additionally, we provide more background information on MNNs, on the connections between individual models of MNNs, and on the learning algorithms of MPs and FLNNs, the main constructive morphological models [40, 54, 31, 23]. For instance, we present an improved version of the training algorithm for MPs, introduce MPs and MPDs with competitive neurons and show how to train them. Finally, we elaborate on the foundations of morphological perceptrons (MPs) and fuzzy lattice neural networks (FLNNs) in lattice theory [4, 5]. Moreover we explain why fuzzy lattice neural networks can be viewed as morphological models, include some further comparisons with MPs, and provide additional details with respect to the experimental results and the computational effort involved in using morphological models.

The paper is organized as follows. After presenting the lattice background of MNNs, we investigate the most important types of constructive MNNs, namely morphological perceptrons (MPs), morphological perceptrons with dendrites (MPDs), and fuzzy lattice neural networks (FLNNs). Section 4 compares the performances of morphological models and MLPs in some classification problems. We finish the paper with some concluding remarks.

2 Lattice Background for Morphological Neural Networks

Morphological neural networks are geared at merging techniques of artificial neural networks and mathematical morphology. Although mathematical morphology was conceived as a set-theoretic approach to image processing, its theoretical foundations can be found in lattice algebra. This paper concentrates on morphological models of neural networks whose operations can not only be described in terms of set-theoretic ideas but also in terms of the complete lattice framework of mathematical morphology [21, 42, 44].

A partially ordered set $\mathbb{L}$ is called a *lattice* if and only if every finite, non-empty subset of $\mathbb{L}$ has an infimum and a supremum in $\mathbb{L}$. For simplicity, we assume that a partially ordered set is non-empty [18]. A lattice $\mathbb{L}$ is *complete* if every non-empty (finite or infinite) subset has an infimum and a supremum in $\mathbb{L}$ [5]. Every (non-empty) complete lattice has a least element denoted by $0_{\mathbb{L}}$ and a greatest element denoted by $1_{\mathbb{L}}$. The extended real numbers $\bar{\mathbb{R}}$ and the unit interval $[0,1]$ represent specific examples of complete lattices. For any $Y \subseteq \mathbb{L}$, we denote the infimum of Y by the symbol $\bigwedge Y$ and we write $\bigwedge_{j\in J} y_j$ instead of $\bigwedge Y$ if $Y = \{y_j, j \in J\}$ for a index set J. We use similar notations to denote the *supremum* of Y.

If $\mathbb{L}_1, \ldots \mathbb{L}_n$ are lattices, a partial order on $\mathbb{L} = \mathbb{L}_1 \times \ldots \times \mathbb{L}_n$ can be defined by setting

$$(x_1, \ldots, x_n) \leq (y_1, \ldots, y_n) \;\Leftrightarrow\; x_i \leq y_i \quad \forall i \in \{1, \ldots, n\}. \tag{1}$$

The resulting partially ordered set $\mathbb{L}$ is also a lattice and is called the *product lattice* with constituents $\mathbb{L}_1, \ldots \mathbb{L}_n$. If the lattices $\mathbb{L}_1, \ldots \mathbb{L}_n$ are complete then the product lattice $\mathbb{L} = \mathbb{L}_1 \times \ldots \times \mathbb{L}_n$ is complete as well. For notational convenience, the product lattice corresponding to the product of n copies of $\mathbb{L}$ is denoted using the symbol $\mathbb{L}^n$. Suppose that $\mathbb{L}$ and $\mathbb{M}$ are lattices. A function $\varphi : \mathbb{L} \to \mathbb{M}$ that satisfies the following equations for all $x \in \mathbb{L}$ and for all $y \in \mathbb{M}$ is called *lattice homomorphism.*

$$\varphi(x \vee y) = \varphi(x) \vee \varphi(y) \quad \text{and} \quad \varphi(x \wedge y) = \varphi(x) \wedge \varphi(y). \tag{2}$$

A bijective lattice homomorphism is called *lattice isomorphism*. Equivalently, we have that $\varphi : \mathbb{L} \to \mathbb{M}$ is a lattice isomorphism if φ is bijective and order preserving, that is $\varphi(x) \leq \varphi(y)$ for all $x \leq y$.

A central issue in mathematical morphology is the decomposition of mappings between complete lattices in terms of elementary operations.

Definition 1. *Let $\varepsilon, \delta, \bar{\varepsilon}, \bar{\delta}$ be operators from the complete lattice $\mathbb{L}$ to the complete lattice $\mathbb{M}$, and let $Y \subseteq \mathbb{L}$.*

$$\varepsilon \quad \textit{is called erosion} \Leftrightarrow \varepsilon\left(\bigwedge Y\right) = \bigwedge_{y\in Y} \varepsilon(y); \tag{3}$$

$$\delta \quad \textit{is called dilation} \Leftrightarrow \delta\left(\bigvee Y\right) = \bigvee_{y\in Y} \delta(y); \tag{4}$$

$$\bar{\varepsilon} \quad \textit{is called anti-erosion} \Leftrightarrow \bar{\varepsilon}(\bigwedge Y) = \bigvee_{y \in Y} \bar{\varepsilon}(y); \tag{5}$$

$$\bar{\delta} \quad \textit{is called anti-dilation} \Leftrightarrow \bar{\delta}(\bigvee Y) = \bigwedge_{y \in Y} \bar{\delta}(y). \tag{6}$$

The following theorem establishes representations of anti-dilations and anti-erosions in terms of erosions, dilations and negations.

Theorem 1. *Let $\mathbb{L}$ and $\mathbb{M}$ be complete lattices with negations $\nu_{\mathbb{L}}$ and $\nu_{\mathbb{M}}$, respectively.*

- *An operator $\bar{\delta} : \mathbb{L} \to \mathbb{M}$ is an anti-dilation $\Leftrightarrow$ $\bar{\delta} = \varepsilon \circ \nu_{\mathbb{L}}$ or $\bar{\delta} = \nu_{\mathbb{M}} \circ \delta$, where δ is a dilation and ε is a erosion.*
- *An operator $\bar{\varepsilon} : \mathbb{M} \to \mathbb{L}$ is an anti-erosion $\Leftrightarrow$ $\bar{\varepsilon} = \delta \circ \nu_{\mathbb{M}}$ or $\bar{\varepsilon} = \nu_{\mathbb{L}} \circ \varepsilon$, where ε is an erosion and δ is a dilation.*

Banon and Barrera [4] showed that for every mapping $\psi : \mathbb{L} \longrightarrow \mathbb{M}$ there exist erosions ε^i and anti-dilations $\bar{\delta}^i$ for some index set I such that

$$\psi = \bigvee_{i \in I} (\varepsilon^i \wedge \bar{\delta}^i). \tag{7}$$

Similarly, the mapping ψ can be written as an infimum of supremums of pairs of *dilations* and *anti-erosions*. In the special case that ψ is increasing, ψ can be represented as a supremum of erosions or as an infimum of dilations.

Many models of MNNs can alternatively be defined in terms of certain matrix products in minimax algebra [13, 11]. Minimax algebra is a lattice algebra whose origins lie in the field of operations research and machine scheduling [9, 12, 16, 57].

In minimax algebra, we consider certain algebraic structures called *belts* and *bounded lattice ordered groups*. For our purposes, it is enough to consider the bounded lattice ordered group $(\mathbb{G}, \vee, \wedge, +, +')$, where the symbol $\mathbb{G}$ denotes $\bar{\mathbb{R}} = \mathbb{R} \cup \{-\infty, \infty\}$ or $\bar{\mathbb{Z}} = \mathbb{Z} \cup \{-\infty, \infty\}$. The symbols $\vee$ and $\wedge$ denote the binary operations of maximum and minimum, respectively. The operations $+$ and $+'$ act like the usual sum operation on $\mathbb{G}$ and only differ from each other in the following respect:

$$\infty + (-\infty) = (-\infty) + \infty = \infty \tag{8}$$

$$\infty +' (-\infty) = (-\infty) +' \infty = -\infty \tag{9}$$

There are two types of matrix products with entries in $\mathbb{G}$. Given a matrix $A \in \mathbb{G}^{m \times p}$ and a matrix $B \in \mathbb{G}^{p \times n}$, the matrix $C = A \boxdot B$, called the *max-product* of A and B, and the matrix $D = A \boxtimes B$, called the *min-product* of A and B, are defined by the following equations:

$$c_{ij} = \bigvee_{k=1}^{p} (a_{ik} + b_{kj}), \quad d_{ij} = \bigwedge_{k=1}^{p} (a_{ik} +' b_{kj}). \tag{10}$$

Let $A \in \mathbb{G}^{n\times m}$. Consider the following operators ε_A and δ_A:

$$\varepsilon_A(x) = A^t \boxed{\vee}\, x, \tag{11}$$
$$\delta_A(x) = A^t \boxed{\wedge}\, x. \tag{12}$$

Note that the operators ε_A and δ_A represent erosions and dilations from the complete lattice $\mathbb{G}^n$ to the complete lattice $\mathbb{G}^m$, respectively. The theory of minimax algebra includes a theory of conjugation. For more information, we refer the reader to the treatises of Cuninghame-Green [13, 11]. The bounded lattice ordered group $(\mathbb{G}, \vee, \wedge, +, +')$ is self-conjugate. The conjugate of an element $x \in \mathbb{G}$ is denoted using the symbol x^* and is defined as follows:

$$x^* = \begin{cases} -x, & \text{if } x \in \mathbb{G} \setminus \{-\infty, +\infty\} \\ +\infty, & \text{if } x = -\infty \\ -\infty, & \text{if } x = \infty \end{cases} \tag{13}$$

The operator of conjugation gives rise to a negation ν_* on $\mathbb{G}^n$ which maps the i-th component of $\mathbf{x}$ to its conjugate. Formally, we have

$$(\nu_*(\mathbf{x}))_i = (x_i)^* \;\; \forall\, i = 1, \ldots, n. \tag{14}$$

3 Some Constructive Morphological Neural Network Models

Morphological neural networks are equipped with morphological neurons. We speak of a morphological neuron if its aggregation function corresponds to an elementary morphological operation. As mentioned before, the emphasis in this paper is on constructive MNNs. To our knowledge, the class of constructive MNNs consists of morphological perceptrons, morphological perceprons with dendrites, and - as shown in this section - fuzzy lattice neural networks (FLNNs).

This sections provides a new perspective on constructive MNNs by exhibiting the relations between these models. In addition, we introduce a modified version of the training algorithm for morphological perceptron which has led to better experimental results in Section 4 when compared to the original algorithm [54].

3.1 Morphological Perceptron (MP)

Morphological perceptrons [40, 54] grew out of the minimax subalgebra of image algebra [11, 41, 39]. Although MPs have been formulated in terms of matrix products in minimax algebra, it was not until a recent conference paper that MPs were viewed in terms of the complete lattice framework of mathematical morphology [29]. Here, we provide some more details on this issue.

Recall that $\bar{\mathbb{R}}$ and $\bar{\mathbb{R}}^n$ represent complete lattices. Given a vector of inputs $\mathbf{x} \in \bar{\mathbb{R}}^n$ (in practice, we restrict ourselves to input vectors $\mathbf{x} \in \mathbb{R}^n$), a vector of synaptic weights $\mathbf{w} \in \bar{\mathbb{R}}^n$ and an activation function f, a neuron of the morphological perceptron calculates the output y according to one of the following rules:

$$y = f(\varepsilon_{\mathbf{w}}(\mathbf{x})), \text{ where } \varepsilon_{\mathbf{w}}(\mathbf{x}) = \mathbf{w}^t \boxed{\wedge}\, \mathbf{x} = \bigwedge_{i=1}^{n} (x_i + w_i); \tag{15}$$

$$y = f(\delta_{\mathbf{w}}(\mathbf{x})), \text{ where } \delta_{\mathbf{w}}(\mathbf{x}) = \mathbf{w}^t \boxed{\vee}\, \mathbf{x} = \bigvee_{i=1}^{n} (x_i + w_i); \tag{16}$$

$$y = f(\bar{\varepsilon}_{\mathbf{w}}(\mathbf{x})), \text{ where } \bar{\varepsilon}_{\mathbf{w}}(\mathbf{x}) = \delta_{\mathbf{w}}(\mathbf{x}) \circ \nu_*(\mathbf{x}) = \bigvee_{i=1}^{n} (x_i^* + w_i); \tag{17}$$

$$y = f(\bar{\delta}_{\mathbf{w}}(\mathbf{x})), \text{ where } \bar{\delta}_{\mathbf{w}}(\mathbf{x}) = \varepsilon_{\mathbf{w}}(\mathbf{x}) \circ \nu_*(\mathbf{x}) = \bigwedge_{i=1}^{n} (x_i^* + w_i). \tag{18}$$

Note that $\varepsilon_{\mathbf{w}}$ represents an erosion from the complete lattice $\bar{\mathbb{R}}^n$ to the complete lattice $\bar{\mathbb{R}}$ in the special form of Equation 11. Similarly, $\delta_{\mathbf{w}}$ represents a dilation $\bar{\mathbb{R}}^n \to \bar{\mathbb{R}}$ in the special form of Equation 12. By Theorem 1, the composition of the negation ν_* followed by the erosion $\varepsilon_{\mathbf{w}}$ yields an anti-erosion that we denoted by $\bar{\varepsilon}_{\mathbf{w}}$ and the composition of the negation ν_* followed by the dilation $\delta_{\mathbf{w}}$ yields an anti-dilation that we denoted by $\bar{\delta}_{\mathbf{w}}$.

The values of the morphological perceptron's weights must be determined before it can act as a classifier. More precisely, the weights are determined using a supervised learning algorithm [54] that constructs n-dimensional boxes around sets of points which share the same class value. Convergence occurs in a finite number of steps.

In this paper, we present an improved version of the original training algorithm for MPs that was proposed to solve two-class classification problems [54]. To this end, let us introduce some relevant notations.

The vectors $\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^k \in \mathbb{R}^n$ denote the given training patterns. The set of training patterns belonging to class 0 is denoted using the symbol C_0 and the set of training patterns belonging to class 1 is denoted using the symbol C_1. We define the following index sets:

$$K(0) = \{j \in \{1, \ldots, k\} : \mathbf{x}^j \in C_0\}. \tag{19}$$

$$K(1) = \{j \in \{1, \ldots, k\} : \mathbf{x}^j \in C_1\}. \tag{20}$$

Let P be the hyperbox $\text{box}(\mathbf{p}^{\perp}, \mathbf{p}^{\top}) = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{p}^{\perp} \leq \mathbf{x} \leq \mathbf{p}^{\top}\}$, where the symbols $\mathbf{p}^{\perp}$ and $\mathbf{p}^{\top} \in \bar{\mathbb{R}}^n$ represent the lower vertex and upper vertex, respectively. The symbol P° stands for the interior of P and the symbol ∂P stands for the boundary of P, that is $P^{\circ} = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{p}^{\perp} < \mathbf{x} < \mathbf{p}^{\top}\}$ and $\partial P = P \setminus P^{\circ}$. Furthermore, let us define the following half-spaces $H_i^+(\mathbf{x})$ and $H_i^-(\mathbf{x})$ for every input pattern $\mathbf{x} \in \mathbb{R}^n$:

$$H_i^+(\mathbf{x}) = \{\mathbf{y} \in \bar{\mathbb{R}}^n : y_i \geq x_i\} \text{ and } H_i^-(\mathbf{x}) = \{\mathbf{y} \in \bar{\mathbb{R}}^n : y_i \leq x_i\}. \tag{21}$$

The training algorithm will automatically produce the arquitecture depicted in Figure 1**(a)**. The staircase symbol for the activation function at the output node represents the Heaviside step function given by

$$f(x) = \begin{cases} 1, & \text{if } x \geq 0, \\ 0, & \text{if } x < 0. \end{cases} \tag{22}$$

Given an arbitrary input pattern $\mathbf{x} \in \mathbb{R}^n$, the MP computes the output in terms of $f(g(\mathbf{x}))$, where g is some function $\mathbb{R}^n \to \bar{\mathbb{R}}$. Let $L(0)$ denote the set of indices $j \in \{1,\ldots,k\}$ such that $f(g(\mathbf{x}^j)) \in C_0$ and let $L(1)$ denote the set of indices $j \in \{1,\ldots,k\}$ such that $f(g(\mathbf{x}^j)) \in C_1$. The symbol D refers to the set of indices corrsponding to class 1 patterns that are currently misclassified. Formally, we have $D = \{j \in L(0) \cap K(1)\}$.

We initialize the function $g : \mathbb{R}^n \to \bar{\mathbb{R}}$ by setting $g(\mathbf{x}) = -\infty$ for all $\mathbf{x} \in \mathbb{R}^n$. Thus, initially we have $L(0) = \{1,\ldots,k\}$ and $D = K(1)$.

Step 1 (Find a Hyperbox containing only Class 1 Patterns)

1 *While* $D \neq \emptyset$

1.1 Let $P = \text{box}(\mathbf{p}^\perp, \mathbf{p}^\top)$ where the vertices $\mathbf{p}^\perp$ and $\mathbf{p}^\top$ satisfy:

$$p_i^\top = \bigvee_{j \in D} x_i^j \ \forall i = 1,\ldots,n, \tag{23}$$

$$p_i^\perp = \bigwedge_{j \in D} x_i^j \ \forall i = 1,\ldots,n. \tag{24}$$

1.2 If there exists an index i_0 such that $p_{i_0}^\top = p_{i_0}^\perp$ then perform the following steps.

(a) If, in addition, the set $P \cap C_0$ is empty then select the pattern $\mathbf{x}^j \in P \cap C^1$ such that j is minimal and set $P = \{\mathbf{x}^j\}$.

(b) In any event, modify the upper and lower corner of P as follows:

$$p_i^\perp = \sup\{x_i < p_i^\perp : \mathbf{x} \in C_0\}. \tag{25}$$

$$p_i^\top = \inf\{x_i > p_i^\top : \mathbf{x} \in C_0\}. \tag{26}$$

Here, the supremum and the infimum are taken in $\bar{\mathbb{R}}$. In particular, we have $\sup \emptyset = -\infty$ and $\inf \emptyset = \infty$.

1.3 Otherwise, proceed as follows. Consider the set $S = C_0 \cap P$.

(a) If $S = \emptyset$ then use Equations 25 and 26 to expand the hyperbox P.

(b) If $S \neq \emptyset$ then continue as follows

(i) For all $j = 1,\ldots,k$ such that $\mathbf{x}^j \in C_0 \cap P$ execute the following steps. Set $\mathbf{x} = \mathbf{x}^j$. Consider the hyperboxes $H_i^+(\mathbf{x}) \cap P$ and $H_i^-(\mathbf{x}) \cap P$ for $i = 1,\ldots n$. Among these $2n$ hyperboxes, choose the hyperbox $P' \subseteq P$ that contains the largest number of currently misclassified patterns in C_1 such that $\mathbf{x}$ does not belong to P' (if more than one of the hyperboxes $H_i^\pm(\mathbf{x}) \cap P$ meets these criteria then randomly select one of the these). Update P by setting $P = P'$.

(ii) Expand the hyperbox P that was obtained in item (a) by applying Equations 25 and 26.

1.4 Determine the smallest hyperbox $B = \text{box}(\mathbf{b}^{\perp}, \mathbf{b}^{\top})$ which contains all the misclassified patterns of class 1 in the interior of P. Formally, we have

$$b_i^{\top} = \bigvee_{\mathbf{x}^j \in P^{\circ}} x_i^j \ \forall i = 1, \ldots, n \tag{27}$$

$$b_i^{\perp} = \bigwedge_{\mathbf{x}^j \in P^{\circ}} x_i^j \ \forall i = 1, \ldots, n \tag{28}$$

If $B = \emptyset$ then choose the pattern $\mathbf{x}^j \in \partial P \cap C_1$ such that j is minimal, redefine P as $P = \{\mathbf{x}^j\}$, and return to Step 1.3(b)(ii) (the next time around, $B \neq \emptyset$).

1.5 Determine an intermediary hyperbox C whose upper and lower corner, denote respectively by $\mathbf{c}^{\top}$ and $\mathbf{c}^{\perp}$, are given by the averages of the corresponding vertices of B and P.

$$c_i^{\top} = \frac{b_i^{\top} + p_i^{\top}}{2}, \tag{29}$$

$$c_i^{\perp} = \frac{b_i^{\perp} + p_i^{\perp}}{2}. \tag{30}$$

Step 2 (Update the Architecture of the Morphological Perceptron) At the end of this step, the patterns in the hyperbox C are assigned to class 1.

2.1 Update the function g as follows:

$$g(\mathbf{x}) = g(\mathbf{x}) \vee \left[\bigwedge_{i=1}^{n} (x_i - c_i^{\top}) \wedge \bigwedge_{i=1}^{n} (c_i^{\perp} - x_i) \right]. \tag{31}$$

2.2 Compute $f(g(\mathbf{x}))$ for all $\mathbf{x} \in D$, update the set D, and return to Step 1.

Note that the function g determines a union of hyperboxes. An arbitrary input pattern $\mathbf{x}$ is assigned to class 1 if and only if it lies in this union of hyperboxes. The term $\bigwedge_{i=1}^{n}(x_i - c_i^{\top}) \wedge \bigwedge_{i=1}^{n}(c_i^{\perp} - x_i)$ of Equation 31 is non-negative if and only if $\mathbf{x}$ is between the upper and lower vertices of C. The term $\bigwedge_{i=1}^{n}(x_i - c_i^{\top})$ corresponds to the erosion $\varepsilon_{\mathbf{v}}$ where $\mathbf{v} = -\mathbf{c}^{\top}$ and the term $\bigwedge_{i=1}^{n}(c_i^{\perp} - x_i)$ corresponds to an anti-dilation $\bar{\delta}_{\mathbf{w}} = \varepsilon_{\mathbf{w}} \circ \nu_*$ where $\mathbf{w} = \mathbf{c}^{\perp}$. During the training phase, pairs of erosive and anti-dilative neurons are added to the hidden layer of the MP that is pictured in Figure 1(**a**). This process ends once all training patterns are classified correctly. After convergence, the output of the MP is determined by the following equation for some $m \in \mathbb{N}$:

$$y = f\left(\bigvee_{j=1}^{m} (\varepsilon_{\mathbf{v}_j}(\mathbf{x}) \wedge \bar{\delta}_{\mathbf{w}_j}(\mathbf{x}))\right) \tag{32}$$

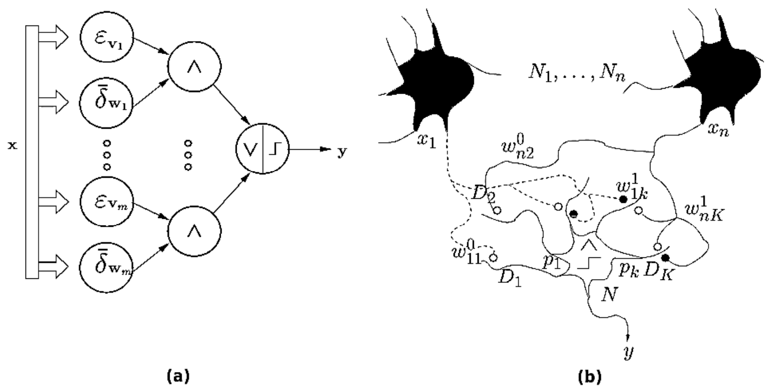

Fig. 1 Architectures of a morphological perceptron (**a**) and of an MPD (**b**), respectively.

An arbitrary input pattern **x** is classified as belonging to class 1 if and only if $y = 1$. According to Equation 34 and Figure 1(**a**), the MP calculates a maximum of pair-wise minimums of erosions and anti-dilations which approximates the decomposition suggested by Banon and Barrera [4] (cf. Equation 7) followed by the application of a hard-limiting function f.

The modifications of the original algorithm proposed in [54] can be found in Steps 1.2, 1.3, and 1.4. We have in particular taken additional measures in order to circumvent situations in which the original algorithm failed to converge. The modified version is guaranteed to converge in a finite number of steps yielding a decision surface that perfectly separates the class 0 and the class 1 training data and led to better results in the classification problems described in Section 4.

In a previous conference paper, we have allocated three morphological neurons to perform a minimum of an erosion $\varepsilon_{\mathbf{v}}$ and an anti-dilation $\bar{\delta}_{\mathbf{w}}$ although one could argue that forming

$$\varepsilon_{\mathbf{v}}(\mathbf{x}) \wedge \bar{\delta}_{\mathbf{w}}(\mathbf{x}) = \bigwedge_{i=1}^{n} (x_i + v_i) \wedge \bigwedge_{i=1}^{n} (x_i^* + w_i) \tag{33}$$

only requires one single morphological neuron or processing element with inputs of the form $(x_1, \ldots, x_n, x_1^*, \ldots, x_n^*)$ and weights in $\mathbb{R}^{2n}$. Therefore, we associate only one morphological neuron to the computation of Equation 33. In other words, the number of hidden morphological neurons corresonds to the number of hyperboxes that are generated during the learning phase.

Suppose we have an S-class classification problem. If $S > 2$ then the MP approach has to be adapted so as to be able to deal with *multiple classes*. Let $\bar{C}_s \subseteq \{\mathbf{x}^1, \ldots, \mathbf{x}^k\}$ denote the set of training patterns belonging to the sth class where $s = 1, \ldots, S$. For each $s = 1, \ldots S$, we simply set $C_1 = \bar{C}_s$ and $C_0 = \bigcap_{t \neq s} \bar{C}_t$ and apply the MP training algorithm. This procedure generates weights $\mathbf{v}^s_j$ and $\mathbf{w}^s_j$ for every $s = 1, \ldots S$. Thus, we obtain S MPs with threshold activation functions at their respective output nodes of the form pictured in Figure 1(**a**). Removing the thresholds at the outputs yields S MPs that are given by the following equations for $s = 1, \dots S$:

$$y_s = \bigvee_{j=1}^{m_s} (\varepsilon_{\mathbf{v}_j^s}(\mathbf{x}) \wedge \bar{\delta}_{\mathbf{w}_j^s}(\mathbf{x})) \tag{34}$$

An MP for multi-class classification problems arises by joining the S MPs and by introducing *competitive output neurons*. In other words, an input pattern $\mathbf{x}$ will be classified as belonging to class $y = \arg\max_s y_s$. For simplicity, we use the acronym MP/C to denote the resulting *morphological perceptron with competitive neurons*.

3.2 Morphological Perceptrons with Dendrites (MPD)

Recent research in neuroscience has given considerable importance to dendritic structures in a single neuron cell [43]. Ritter and Urcid developed a new paradigm for computing with morphological neurons where the process occurs in the dendrites [37, 38]. Figure 1(**b**) provides a graphical representation of an MPD that has a single output neuron N.

The architecture of an MPD is not determined beforehand. During the training phase, the MPD grows new dendrites while the input neurons expand their axonal branches to synapse on the new dendrites. The weight of an axonal branch of input neuron N_i terminating on the k-th dendrite of the output neuron N is denoted by w_{ki}^l where the superscript $l \in \{0,1\}$ distinguishes between *excitatory* ($l = 1$) and *inhibitory* ($l = 0$) *input* to the dendrite. The k-th dendrite of N will produce either an *excitatory* ($p_k = 1$) or an *inhibitory* ($p_k = -1$) *response* to the total input received from the input neurons N_i. To summarize, the computation performed by the k-th dendrite is given by

$$\tau_k(\mathbf{x}) = p_k \bigwedge_{i=1}^{n} \bigwedge_{l \in L} (-1)^{l+1}(x_i + w_{ki}^l), \tag{35}$$

where $L \subseteq \{0,1\}$ corresponds to the set of terminal fibers on N_i that synapse on the k-th dendrite of N. After passing the value $\tau_k(\mathbf{x})$ to the cell body, the state of N is given by $\bigvee_{k=1}^{K} \tau_k(\mathbf{x})$, where K denotes the total number of dendrites of N. Finally, an application of the hard limiting function f defined in Equation 22 yields the next state of N, in other words the output y of the MPD depicted in Figure 1(**b**).

$$y = f(\bigwedge_{k=1}^{K} \tau_k(\mathbf{x})) = f(\bigwedge_{k=1}^{K} p_k \bigwedge_{i=1}^{n} \bigwedge_{l \in L} (-1)^{l+1}(x_i + w_{ki}^l)). \tag{36}$$

Leaving the biological motivation aside, the MPD training algorithm that was proposed for binary classification problems [38] resembles the one for MPs [54]. As is the case for MPs, the MPD training algorithm is guaranteed to converge in a finite number of steps and, after convergence, all training patterns will be classified correctly. Learning is based on the construction of n-dimensional hyperboxes. Given two classes C_0 and C_1, an input pattern $\mathbf{x}$ is classified as belonging to class C_1 if and only if $\mathbf{x}$ is contained in one of the constructed hyperboxes. In fact, each dendrite corresponds to a hyperbox. Therefore, we can convert an MPD into an MP and express the computation performed by a dendrite in terms of Equation 33.

When faced with multi-class classification problems, we propose to construct a MPD with competitive output units and to proceed in the same way as we did with MPs at the end of Section 3.1. We will refer to the resulting MNN as *morphological perceptron with dendrites and competitive neurons* (MPD/C).

3.3 Fuzzy Lattice Neural Network-FLNN

The theoretical framework of FLNN constitutes a successful combination of fuzzy sets [56], lattice theory [21] and adaptive resonance theory [8]. Figure 2 illustrates the architecture of the FLNN that consists of an *input layer* and a *category layer*. The input layer has N artificial neurons used for storing and comparing input data. The category layer has L artificial neurons that define M classes.

Given a vector of inputs $\mathbf{x}$ and a vector of synaptic weights $\mathbf{w}$, a neuron of an FLNN [31, 24] computes the degree of inclusion of $\mathbf{x}$ in $\mathbf{w}$ in terms of $p(\mathbf{x},\mathbf{w})$ where p is a fuzzy partial order relation. In general, we refer to p as a *fuzzy partial order* on a lattice $\mathbb{L}$ if p is a function $\mathbb{L}\times\mathbb{L}\rightarrow[0,1]$ that satisfies the equation $p(\mathbf{x},\mathbf{y})=1$ if and only if $\mathbf{x}\leq\mathbf{y}$. (We prefer to speak of a fuzzy partial order instead of a fuzzy membership function or fuzzy inclusion measure because p generalizes the conventional partial order.) A pair $(\mathbb{L},p)$ consisting of a lattice $\mathbb{L}$ and a fuzzy partial order p is called a *fuzzy lattice*.

In the case of FLNNs, both the input vector $\mathbf{x}$ and the vector of synaptic weights $\mathbf{w}$ are hyperboxes in $\mathbb{L}^N$ where $\mathbb{L}$ is a complete lattice. For the special case where

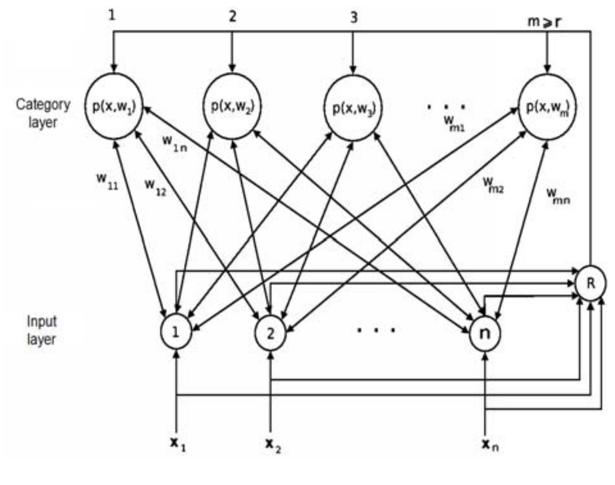

Fig. 2 Architecture of the FLNN.

$N = 1$, a hyperbox in $\mathbb{L}^N$ corresponds to a closed interval in $\mathbb{L}$ and can be written in the form $[a,b]$ where $a,b \in \mathbb{L}$. In particular, if $\mathbb{L} = [0,1]$ we obtain a closed subinterval of the unit interval. The partial order on a given lattice $\mathbb{L}$ induces a partial order on set of intervals $\mathscr{I}_{\mathbb{L}} = \{[\mathbf{a},\mathbf{b}] : \mathbf{a},\mathbf{b} \in \mathbb{L} \text{ and } \mathbf{a} \leq \mathbf{b}\}$ which turns $\mathscr{I}_{\mathbb{L}}$ into a lattice as well:

$$[\mathbf{a},\mathbf{b}] \leq [\mathbf{c},\mathbf{d}] \Leftrightarrow \mathbf{a} \geq \mathbf{c} \text{ and } \mathbf{b} \leq \mathbf{d}\,. \tag{37}$$

Unfortunately, the lattice of the closed intervals is not complete even if $\mathbb{L}$ is complete because $\bigwedge \mathscr{I}_{\mathbb{L}}$ does not exist in $\mathscr{I}_{\mathbb{L}}$. There are however two closely related complete lattices. The first one, called the complete lattice of the *generalized intervals*, is denoted using the symbol $\mathbb{PL}$ and arises by leaving away the restriction $\mathbf{a} \leq \mathbf{b}$. Formally, we have $\mathbb{PL} = \{[\mathbf{a},\mathbf{b}] : \mathbf{a},\mathbf{b} \in \mathbb{L}\}$. If $0_{\mathbb{L}}$ and $1_{\mathbb{L}}$ denote the least element of $\mathbb{L}$ and the greatest element of $\mathbb{L}$, respectively, then the least element of $\mathbb{PL}$ is given by $[1_{\mathbb{L}}, 0_{\mathbb{L}}]$ and the greatest element of $\mathbb{PL}$ is given by $[0_{\mathbb{L}}, 1_{\mathbb{L}}]$. The second complete lattice of interest is denoted by $\mathbb{V}_{\mathbb{L}}$ and is given by adjoining $[1_{\mathbb{L}}, 0_{\mathbb{L}}]$ to $\mathscr{I}_{\mathbb{L}}$. We obtain $\mathbb{V}_{\mathbb{L}} = \mathscr{I}_{\mathbb{L}} \cup \{[1_{\mathbb{L}}, 0_{\mathbb{L}}]\}$.

The FLNN model employs a fuzzy partial order relation $p : (\mathbb{V}_{\mathbb{L}})^N \times (\mathbb{V}_{\mathbb{L}})^N \rightarrow [0,1]$. In this context, the fuzzy partial order is of a special form. Specifically, the fuzzy partial order relation employed in the FLNN is based on a "function-h" or - as we prefer to call it - a *generating function* [31]. Similar fuzzy lattice models use fuzzy partial order relations based on *positive valuation functions* [5, 24, 23].

In applications of the FLNN to classification tasks such as the ones discussed in Section 4, it suffices to consider - after an appropriate normalization - the complete lattice $\mathbb{U} = [0,1]$, i.e., the unit interval. Thus, the input and weight vectors are N-dimensional hyperboxes in $(\mathbb{V}_{\mathbb{U}})^N$. In this case, we can show that $p(.,\mathbf{w}) : (\mathbb{V}_{\mathbb{U}})^N \rightarrow [0,1]$ represents an elementary operation on mathematical morphology, namely both an anti-erosion and an anti-dilation, if the underlying generating function $h : \mathbb{U} \rightarrow \mathbb{R}$ is continuous. The proof of this result is beyond the scope of this paper since it involves further details on fuzzy lattice neuro-computing models, in particular the construction of a fuzzy partial order from a generating function. Therefore, we postpone the proof to a future paper where we will show a more general result that uses additional concepts of lattice theory. Anyway, the fact that every node in the category layer of an FLNN with a continuous generating function performs an elementary operation of MM has led us to classify FLNNs as belonging to the class of morphological neural networks.

FLNNs can be trained in supervised or unsupervised fashion [31, 24]. Both versions generate hyperboxes that determine the output of the FLNN. In this paper, we focus on the supervised learning algorithm that is used in classification tasks. Here we have a set of n training patterns $\mathbf{x}^1, \ldots, \mathbf{x}^n \in (\mathbb{V}_{\mathbb{L}})^N$ together with their class labels $c_1, \ldots, c_n \in \{1, \ldots, M\}$. Therefore, the basic arquitecture of the FLNN has to be adapted so as to accomodate the class information. This can be achieved by allowing for the storage of a class index in each node of the category layer, by augmenting the input layer by one node that carries the class information c_i corresponding to the pattern $\mathbf{x}^i$ during the training phase, and by fully interconnecting the two layers.

During the training phase, the FLNN successively constructs nodes in the category layer each of which is associated with an N-dimensional hyperbox $\mathbf{w}_i$ - i.e., an element of $(\mathbb{V}_\mathbb{L})^N$ - together with its respective class label c_i. Training in the original FLNN model is performed using an exhaustive search and takes $\mathscr{O}(N^3)$ operations [31]. (A modification of the FLNN called *fuzzy lattice reasoning* (FLR) classifier requires $\mathscr{O}(N^2)$ for training if one renounces on optimizing the outcome of training [23].) After training, we obtain L N-dimensional hyperboxes $\mathbf{w}_i \in (\mathbb{V}_\mathbb{L})^N$ that correspond to the L nodes that appear in the category layer of the FLNN (cf. Figure 2(**b**)). A class label $c_i \in \{1,\ldots,M\}$ is associated with each hyperbox $\mathbf{w}_i \in (\mathbb{V}_\mathbb{L})^N$.

In the testing phase, an input pattern $\mathbf{x} \in (\mathbb{V}_\mathbb{L})^N$ is presented to the FLNN and the values $p(\mathbf{x},\mathbf{w}_i)$ are computed for $i = 1,\ldots,L$. A competition takes place among the L nodes in the category layer and the input pattern $\mathbf{x}$ is assigned to the class c_i that is associated with the hyperbox $\mathbf{w}_i$ exhibiting the highest value $p(\mathbf{x},\mathbf{w})$. Informally speaking, the degree of inclusion of $\mathbf{x}$ in $\mathbf{w}_i$ is higher than the degree of inclusion of $\mathbf{x}$ in $\mathbf{w}_j$ for all $j \neq i$. In particular, if $\mathbf{x}$ is contained in $\mathbf{w}_i$ but not contained in $\mathbf{w}_j$ for all $j \neq i$, i.e., $p(\mathbf{x},\mathbf{w}_i) = 1$ and $p(\mathbf{x},\mathbf{w}_j) < 1$ for all $j \neq i$, the $\mathbf{x}$ is classified as belonging to class c_i.

Occasionally, the training algorithms for FLNNs and its modifications produce overlapping hyperboxes with disparate class memberships although - according to Kaburlasos et al. - this event occurs rarely [23]. In the experiments we conducted in Section 4, this situation did in fact occur as evidenced by the decision surface that is visualized in Figure 5. For more information, we refer the reader to Section 4.

4 Experimental Results

In this section we compare the classification performance of the constructive morphological models and the conventional multi-layer perceptron in a series of experiments on two well known datasets: Ripley's synthetic dataset [35, 36] and the image segmentation dataset that can be found in the UCI Machine Learning Repository [6]. In addition, we have used Ripley's synthetic dataset to visualize the respective decision surfaces. In contrast to our previous conference paper, we have decided to omit morphological models with a fixed architecture such as the modular morphological neural network (MMNN) and the hybrid morphological/rank/linear neural network (MRL-NN) since these models produced poor classification results in our simulations [29].

Tables 1 and 2 display the percentages of the misclassified training and testing patterns. Since the type of operations performed by the individual models in a training epoch varies greatly from one model to another we have also included the average CPU time (on a AMD Athlon 64 X2 Dual Core Processor 4200+ with a processing speed of 2.221 GHz) of each individual model until convergence of the training algorithm. We trained the constructive morphological models until their decision surfaces succeeded in perfectly separating the two classes of training data.

Table 1 Percentage of misclassified patterns for training (E_{tr}) and testing (E_{te}) in the experiments, CPU time in seconds for learning (T_{cpu}) and number of hidden artificial neurons (hyperboxes for MNNs) (H_a).

	Ripley's Synthetic Dataset			
Model	$E_{tr}(\%)$	$E_{te}(\%)$	T_{cpu}	H_a
MP	0.0	11.70	1.1	18
MPD	0.0	17.80	0.58	19
MP/C	0.0	10.80	2.63	38
MPD/C	0.0	13.90	1.26	38
FLNN	0.0	11.4/12.0	75.41	46
MLP	8.41	12.55	144.48	10

Table 2 Percentage of misclassified patterns for training (E_{tr}) and testing (E_{te}) in the experiments, CPU time in seconds for learning (T_{cpu}) and number of hidden artificial neurons (hyperboxes for MNNs) (H_a).

	Image Segmentation Dataset			
Model	$E_{tr}(\%)$	$E_{te}(\%)$	T_{cpu}	H_a
MP/C	0.0	17.38	3.29	88
MPD/C	0.0	13.05	0.21	20
FLNN	0.0	10.00	26.87	17
MLP	15.71	26.81	58.41	20

All the models and algorithms were implemented using MATLAB which favors linear operations over morphological operations. We believe that the CPU times for learning in the constructive MP, MPD, MPD/C, and MPD/C models would be even lower if more efficient implementations of the max-product and min-product were used [33].

For a fair comparison of the number of artificial neurons or processing elements in the constructive MNNs, we have implicitly expressed each individual model as a feedforward model with one hidden layer and competitive output nodes and we have counted the number of hidden nodes or hyperboxes that were constructed during the learning phase. The same sequence of training patterns appearing on the respective internet sites were employed for training the constructive MNNs [36, 6].

4.1 Ripley's Synthetic Problem

Ripley's synthetic dataset [36] consists of data samples from two classes [35, 36]. Each sample has two features. The data are divided into a training set and a test set consisting of 250 and 1000 samples, respectively, with the same number of samples belonging to each of the two classes. Thus, we obtain a binary classification problem in $\mathbb{R}^2$. Figures 3, 4, and 5 provide for more insight into the constructive

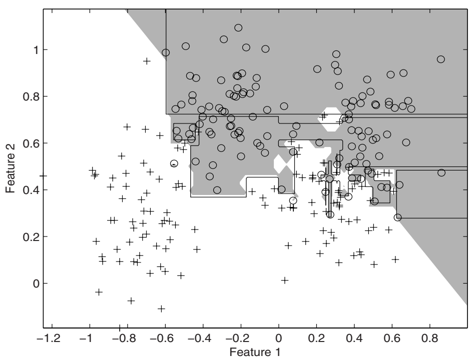

Fig. 3 Decision surfaces of an MP represented by by the continuous line and of an MP with competitive output nodes represented by the difference in shading. Training patterns belonging to class 0 are plotted using "$+$" symbols. Training patterns belonging to class 1 are plotted using "$\circ$" symbols.

MNNs by visualizing the decision surfaces that are generated by these models after training (Figure 5 also includes the decision surface corresponding to an MLP with ten hidden nodes). Here, we have used the same order in which the training patterns appear on Ripley's internet site. We would like to clarify that the decision surfaces vary slightly depending on the order in which the training patterns are presented to the constructive morphological models.

Recall that the decision surfaces of the constructive morphological models are determined by N-dimensional hyperboxes, i.e., rectangles for $N = 2$. This fact is clearly visible in the decision surfaces of the MP and the MPD with hardlimiting output units, that are pictured by means of the continuous lines in Figures 3 and 4.

In addition, Figures 3, 4, and 5 reveal that the decision surfaces generated by the MP/C, MPD/C, and FLNN models deviate from rectangular appearance of the ones generated by the basic MP and MPD models. In this context, recall that the MP/C and MPD/C models construct separate families of hyperboxes for the training patterns of each class. Each family of hyperboxes is associated to a different class and corresponds to a certain output node. Upon presentation of an input pattern $\mathbf{x}$ to the MP/C or MPD/C model a competition among the output nodes occurs that determines the result of classification.

In the FLNN model a similar competition occurs in the category layer. More precisely, the FLNN uses information on the degrees of inclusion $p(\mathbf{x}, \mathbf{w}_i)$ of an input pattern $\mathbf{x}$ in the hyperboxes $\mathbf{w}_i$ for classification by associating $\mathbf{x}$ to the class of the hyperbox $\mathbf{w}_i$ in which $\mathbf{x}$ exhibits the highest degree of inclusion. This property of the FLNN is evidenced by the diagonal lines in its decision surface (cf. Figure 5).

The training algorithms of all types of constructive MNNs are guaranteed to produce decision surfaces that perfectly separate the training patterns with different class labels. However, the training algorithms of the MP/C, the MPD/C, and the FLNN may result in overlapping hyperboxes with distinct class memberships

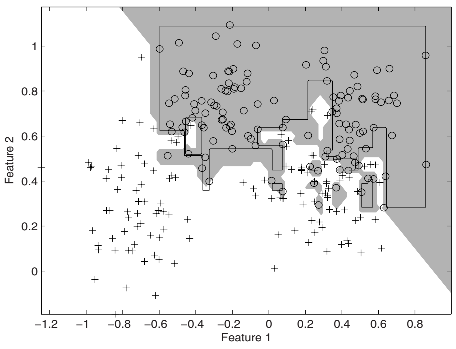

Fig. 4 Decision surfaces of an MPD represented by by the continuous line and of an MPD with competitive output nodes represented by the difference in shading.

although this event did not happen in our simulations with Ripley's synthetic dataset when using the MP/C and MPD/C models. Concerning the FLNN, Figure 5 depicts intersections of rectangles with distinc class labels using a darker shade of gray. These sets of intersection correspond to regions of indecision since a pattern $\mathbf{x}$ that is contained in both $\mathbf{w}_i$ and $\mathbf{w}_j$ with $c_i \neq c_j$ satisfies $p(\mathbf{x}, \mathbf{w}_i) = 1 = p(\mathbf{x}, \mathbf{w}_j)$.

Table 1 exhibits the results that we obtained concerning the classification performance and the computational effort required by the individual models. The MP training algorithm described in Section 3.1.1 automatically generated 19 hyperboxes corresponding to 19 (augmented) hidden neurons capable of evaluating Equation 33. In a similar manner, training an MPD using the constructive algorithm of Ritter and Urcid [38] yielded 19 dendrites that correspond to 19 hidden computational units. In contrast to the basic MP and MPD models, the MP/C and MPD/C generate one family of hyperboxes for each class of training patterns. Since Ripley's synthetic problem represents a binary classification problem, the number of hidden computational units in the MP/C and MPD/C models is approximately twice as high as in the MP and MPD. The FLNN grew 46 neurons in the category layer during the training phase. Moreover, we compared the morphological models with an MLP with ten hidden nodes that was trained using gradient descent with momentum and adaptive step backpropagation rule (learning rate $\eta = 10^{-4}$, increase and decrease parameters 1.05 and 0.5 respectively, momentum factor $\alpha = 0.9$). In addition, we used 25-fold cross-validation in conjunction with the MLP and chose the weights that led to the least validation error.

Table 1 reveals that the MP and MPD models including their variants with competitive nodes converge rapidly to a set of weights that yield perfect separation of the training data. The FLNN model also produces no training error but the convergence of the training algorithm is slower yet not quite as slow as MLP training. All the models we tested exhibited satisfactory results in classification. The MPD yields the highest classification error for testing which is due to the fact that, in contrast to the MP training algorithm, no expansion of the hyperboxes corresponding to C_1 patterns takes place in MPD training (this is why the MPD learns faster than the MP). This lack of expansion does not cause any problems if competing hyperboxes are constructed for patterns of both classes as is the case for the MPD/C.

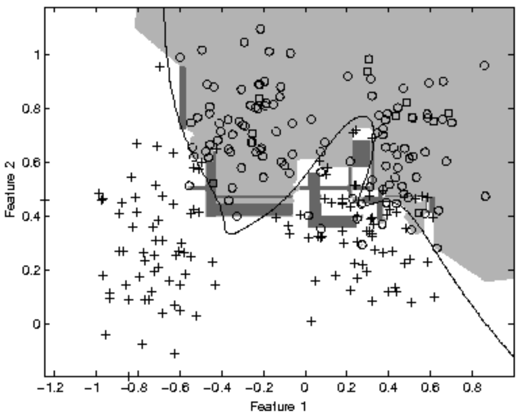

Fig. 5 Decision surfaces of an FLNN represented by the difference in shading (dark regions refer to areas of uncertainty where hyperboxes with different class labels overlap) and of an MLP represented by the continuous line.

As the reader may recall by taking a brief glance at Figure 5, the FLNN produces areas of indecision, i.e., overlapping hyperboxes with distinct class memberships. If these areas of indecision are assigned to either one of the two classes, the percentages of misclassification for testing are 11.40% and 12.00%, respectively. Otherwise, if no decision is taken then we obtain a classification error of 14.2%. In any case, the MP/C model exhibits the best classification performance.

4.2 Image Segmentation Problem

The Image Segmentation Dataset was donated by the Vision Group, University of Massachussets, and is included in the Machine Learning Repository of the University of California, Irvine [6]. This dataset consists of 210 samples for training and 2100 samples for testing. The data have 19 continous attributes. Each sample is decribed by 19 continous attributes and corresponds to a 3×3 region that was randomly drawn from an outdoor image. The images were handsegmented to create a classification for every pixel. The goal is to distinguish between 7 different classes: grass, cement, foliage, brickface, path, sky, and window. The MP and MPD can not be applied directly to such a multi-class problem.

Therefore, we considered the MP/C, the MPD/C, the FLNN, and a MLP. We chose to train an MLP with twenty hidden neurons using the Levenberg-Marquardt algorithm because this algorithm produced the lowest classification error for testing with the Image Segmentation Dataset in a recent paper [10]. Actually, we found a testing error of 26.81% that is slightly lower than the error of 28.13% found by Coskun and Yildirim.

The FLNN yields an excellent recognition rate of 90% of the test patterns without the use of any fine tuning as required by other networks [27]. In this case no action with respect to the region of indecision was taken. The classification error can be lowered to 9.6% by associating a pattern $\mathbf{x}$ contained in the region of indecision with the first class label having value 1. The MLD/C and MP/C also exhibit a better classification performance than the MLP. The high number of hidden neurons grown by the MP/C is probably due to the provisions that were taken to circumvent problems of convergence of the training algorithm. We suspect that these problems are caused by integer-valued attributes of training patterns with different class labels.

5 Conclusions

This paper provides an overview and a comparison of morphological neural networks (MNNs) for pattern recognition with an emphasis on constructive MNNs, which automatically grow hidden neurons during the training phase. We have defined MNNs as models of artificial neural networks that perform an elementary operation of mathematical morphology at every node followed by the application of an activation function. The elementary morphological operations of erosion, dilation, anti-erosion, and anti-dilation can be defined in an arbitrary complete lattice.

In many cases, the underlying complete lattice of choice is $\bar{\mathbb{R}}^n$ which has allowed researchers to formulate morphological neurons (implicitly) in terms of the additive maximum and additive minimum operations in the bounded lattice ordered group $(\bar{\mathbb{R}}, \vee, \wedge, +, +')$ - often without being aware of this connection to minimax algebra [13, 51]. In this setting, the elementary morphological operations can be expressed in terms of maximums (or minimums) of sums, which lead to fast neural computational and easy hardware implementation [33, 38].

As to the resulting models of morphological neurons, recent research results have revealed that the maximum operation lying at the core of morphological neurons is neurobiologically plausible [58]. We have to admit though that there is no neurophysiological justification for summing the inputs and the synaptic weights. This lack of neurobiological plausibility can be overcome by means of the isomorphism between the algebraic structure $(\bar{\mathbb{R}}, \vee, \wedge, +, +')$ and $(\mathbb{R}_{\infty}^{\geq 0}, \vee, \wedge, \cdot, \cdot')$ that transforms additive maximum/minimum operations into multiplicative maximum/minimum operations [52]. In this context, we intend to investigate the connections between MNNs and min-max or adaptive logic networks that combine linear operations with minimums and maximums [2].

In this paper, we have related another lattice based neuro-computing model, namely the fuzzy lattice neural network (FLNN), to MNNs. Specifically, we explained that the FLNN can be considered to be one of the constructive MNN models. Morphological perceptrons (MPs) and morphological perceptrons with dendrites (MPDs) also belong to the class of constructive MNN models. In this paper, we introduced a modified MP training algorithm that only requires a finite number of epochs to converge, resulting in a decision surface that perfectly separates the training data. Furthermore, incorporating competitive neurons into the MP and MPD models led to the MP/C and MPD/C models that can be trained using extensions of the MP and MPD training algorithms. Further research has to be conducted to devise more efficient training algorithms for these new morphological models.

Finally, this article has empirically demonstrated the effectiveness of constructive morphological models in simulations with two well-know datasets for classification [36, 6] by analyzing and comparing the error rates and the computational effort for learning. In general, the constructive morphological models exhibited very satisfactory classification results and - except for the FLNN - extremely fast convergence of the training algorithms. On one hand, the constructive morphological models often require more artificial neurons or computational units than conventional models. On the other hand, morphological neural computations based on max-products or min-products are much less complicated than the usual semi-linear neural computations.

Acknowledgements. This work was supported by CNPq under grant no. $306040/2006-9$ and by FAPESP under grant no. $2006/05868-5$.

References

1. Araújo, R.A., Madeiro, R., Sousa, R.P., Pessoa, L.F.C., Ferreira, T.A.E.: An evolutionary morphological approach for financial time series forecasting. In: Proceedings of the IEEE Congress on Evolutionary Computation, Vancouver, Canada, pp. 2467–2474 (2006)
2. Armstrong, W.W., Thomas, M.M.: Adaptive Logic Networks. In: Fiesler, E., Beale, R. (eds.) Handbook of Neural Computation, vol. C1.8, pp. 1–14. IOP Publishing, Oxford University Press, Oxford United Kingdom (1997)
3. Araújo, R.A., Madeiro, F., Pessoa, L.F.C.: Modular morphological neural network training via adaptive genetic algorithm for design translation invariant operators. In: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Toulouse, France (May 2006)
4. Banon, G., Barrera, J.: Decomposition of Mappings between Complete Lattices by Mathematical Morphology, Part 1, General Lattices. Signal Processing 30(3), 299–327 (1993)
5. Birkhoff, G.: Lattice Theory, 3rd edn. American Mathematical Society, Providence (1993)
6. Asuncion, A., Newman, D.J.: UCI Repository of machine learning databases, University of California, Irvine, CA, School of Information and Computer Sciences (2007), `http://www.ics.uci.edu/~mlearn/MLRepository.html`
7. Braga-Neto, U., Goutsias, J.: Supremal multiscale signal analysis. SIAM Journal of Mathematical Analysis 36(1), 94–120 (2004)
8. Carpenter, G.A., Grossberg, S.: ART 3: Hierarchical search using chemical transmitters in self-organizing pattern recognition architectures. In: Neural Networks, vol. 3, pp. 129–152 (1990)
9. Carré, B.: An algebra for network routing problems J. Inst. Math. Appl. 7, 273–294 (1971)
10. Coskun, N., Yildirim, T.: The effects of training algorithms in MLP network on image classification. Neural Networks 2, 1223–1226 (2003)

11. Cuninghame-Green, R.: Minimax Algebra and Applications. In: Hawkes, P. (ed.) Advances in Imaging and Electron Physics, vol. 90, pp. 1–121. Academic Press, New York (1995)
12. Cuninghame-Green, R., Meijer, P.F.J.: An Algebra for Piecewise-Linear Minimax Problems. Discrete Applied Mathematics 2(4), 267–286 (1980)
13. Cuninghame-Green, R.: Minimax Algebra: Lecture Notes in Economics and Mathematical Systems, vol. 166. Springer, New York (1979)
14. Davey, B.A., Priestley, H.A.: Introduction to lattices and order. Cambridge University Press, Cambridge (2002)
15. Gader, P.D., Khabou, M., Koldobsky, A.: Morphological Regularization Neural Networks. Pattern Recognition, Special Issue on Mathematical Morphology and Its Applications 33(6), 935–945 (2000)
16. Giffler, B.: Mathematical solution of production planning and scheduling problems. IBM ASDD, Tech. Rep (1960)
17. Graña, M., Gallego, J., Torrealdea, F.J., D'Anjou, A.: On the application of associative morphological memories to hyperspectral image analysis. In: Mira, J., Álvarez, J.R. (eds.) IWANN 2003. LNCS, vol. 2687, pp. 567–574. Springer, Heidelberg (2003)
18. Grätzer, G.A.: Lattice theory: first concepts and distributive lattices. W. H. Freeman, San Francisco (1971)
19. Haralick, R.M., Shapiro, L.G.: Computer and Robot Vision, vol. 1. Addison-Wesley, New York (1992)
20. Haralick, R.M., Sternberg, S.R., Zhuang, X.: Image Analysis Using Mathematical Morphology: Part I. IEEE Transactions on Pattern Analysis and Machine Intelligence 9(4), 532–550 (1987)
21. Heijmans, H.: Morphological Image Operators. Academic Press, New York (1994)
22. Hocaoglu, A.K., Gader, P.D.: Domain learning using Choquet integral-based morphological shared weight neural networks. Image and Vision Computing 21(7), 663–673 (2003)
23. Kaburlasos, V.G., Athanasiadis, I.N., Mitkas, P.A.: Fuzzy lattice reasoning (FLR) classifier and its application for ambient ozone estimation. International Journal of aproximate reasoning 45(1), 152–188 (2003)
24. Kaburlasos, V.G., Petridis, V.: Fuzzy lattice neurocomputing (FLN) models. Neural Networks 13, 1145–1170 (2000)
25. Khabou, M.A., Gader, P.D.: Automatic target detection using entropy optimized shared-weight neural networks. IEEE Transactions on Neural Networks 11(1), 186–193 (2000)
26. Kim, C.: Segmenting a low-depth-of-field image using morphological filters and region merging. IEEE Transactions on Image Processing 14(10), 1503–1511 (2005)
27. Kwok, J.T.: Moderating the Outputs of Support Vector Machine Classifiers. IEEE Transactions on Neural Networks 10(5), 1018–1031 (1999)
28. Matheron, G., Random Sets and Integral Geometry. Wiley, New York, (1975).
29. Monteiro, A.S., Sussner, P.: A Brief Review and Comparison of Feedforward Morphological Neural Networks with Applications to Classification. In: Kůrková, V., Neruda, R., Koutník, J. (eds.) ICANN 2008,, Part II. LNCS, vol. 5164, pp. 783–792. Springer, Heidelberg (2008)
30. Pessoa, L.F.C., Maragos, P.: Neural networks with hybrid morphological/rank/linear nodes: a unifying framework with applications to handwritten character recognition. Pattern Recognition 33, 945–960 (2000)
31. Petridis, V., Kaburlasos, V.G.: Fuzzy lattice neural network (FLNN): a hybrid model for learning. IEEE Transactions on Neural Networks 9(5), 877–890 (1998)

32. Pitas, I., Venetsanopoulos, A.N.: Morphological Shape Decomposition. IEEE Transactions on Pattern Analysis and Machine Intelligence 12(1), 38–45 (1990)
33. Porter, R., Harvey, N., Perkins, S., Theiler, J., Brumby, S., Bloch, J., Gokhale, M., Szymanski, J.: Optimizing digital hardware perceptrons for multispectral image classification. Journal of Mathematical Imaging and Vision 19(2), 133–150 (2003)
34. Raducanu, B., Graña, M., Albizuri, X.F.: Morphological Scale Spaces and Associative Morphological Memories: Results on Robustness and Practical Applications. Journal of Mathematical Imaging and Vision 19(2), 113–131 (2003)
35. Ripley, B.D.: Pattern Recognition and Neural Networks. Cambridge University Press, Cambridge (1996)
36. Ripley, B.D.: Datasets for Pattern Recognition and Neural Networks, Cambridge, United Kingdom (1996), `http://www.stats.ox.ac.uk/pub/PRNN/`
37. Ritter, G.X., Iancu, L., Urcid, G.: Morphological perceptrons with dendritic structure. In: The 12th IEEE International Conference, Fuzzy Systems, May 2003, vol. 2, pp. 1296–1301 (2003)
38. Ritter, G.X., Urcid, G.: Lattice Algebra Approach to Single-Neuron Computation. IEEE Transactions on Neural Networks 14(2), 282–295 (2003)
39. Ritter, G.X., Wilson, J.N.: Handbook of Computer Vision Algorithms in Image Algebra, 2nd edn. CRC Press, Boca Raton (2001)
40. Ritter, G.X., Sussner, P.: Morphological Perceptrons Intelligent Systems and Semiotics, Gaithersburg, Maryland (1997)
41. Ritter, G.X., Wilson, J.N., Davidson, J.L.: Image Algebra: An Overview. Computer Vision, Graphics, and Image Processing 49(3), 297–331 (1990)
42. Ronse, C.: Why Mathematical Morphology Needs Complete Lattices. Signal Processing 21(2), 129–154 (1990)
43. Segev, I.: Dendritic processing. In: Arbib, M.A. (ed.) The handbook of brain theory and neural networks, MIT Press, Cambridge (1998)
44. Serra, J.: Image Analysis and Mathematical Morphology. In: Theoretical Advances, vol. 2, Academic Press, New York (1998)
45. Serra, J.: Image Analysis and Mathematical Morphology. Academic Press, London (1982)
46. Khabou, M.A., Gader, P.D., Keller, J.M.: LADAR target detection using morphological shared-weight neural networks. Machine Vision and Applications 11(6), 300–305 (2000)
47. Sobania, A., Evans, J.P.O.: Morphological corner detector using paired triangular structuring elements. Pattern Recognition 38(7), 1087–1098 (2005)
48. Soille, P.: Morphological Image Analysis. Springer, Berlin (1999)
49. Sousa, R.P., Pessoa, L.F.C., Carvalho, J.M.: Designing translation invariant operations via neural network training. In: Proceedings of the IEE International Conference on Image Processing, Vancouver, Canada, pp. 908–911 (2000)
50. Sussner, P., Valle, M.E.: Morphological and Certain Fuzzy Morphological Associative Memories with Applications in Classification and Prediction. In: Kaburlasos, V.G., Ritter, G.X. (eds.) Computational Intelligence Based on Lattice Theory. Studies in Computational Intelligence, vol. 67, pp. 149–173. Springer, Heidelberg (2007)
51. Sussner, P., Valle, M.E.: Gray Scale Morphological Associative Memories. IEEE Transactions on Neural Networks 17(3), 559–570 (2006)
52. Sussner, P., Valle, M.E.: Implicative Fuzzy Associative Memories. IEEE Transactions on Fuzzy Systems 14(6), 793–807 (2006)

53. Sussner, P., Grana, M.: Guest Editorial: Special Issue on Morphological Neural Networks. Journal of Mathematical Imaging and Vision 19(2), 79–80 (2003)
54. Sussner, P.: Morphological Perceptron Learning. In: Proceedings of IEEE ISIC/CIRA/ISAS Joint Conference, Gaithersburg, MD, September 1998, pp. 477–482 (1998)
55. Valle, M.E., Sussner, P.: A General Framework for Fuzzy Morphological Associative Memories. Fuzzy Sets and Systems 159(7), 747–768 (2008)
56. Zadeh, L.A.: Fuzzy Sets. Information and Control 8(3), 338–353 (1965)
57. Zimmermann, U.: Linear amd Combinatorial Optimization of Ordered Algebraic Structures. North-Holland, Amsterdam (1981)
58. Yu, A.J., Giese, M.A., Poggio, T.: Biophysiologically Plausible Implementations of the Maximum Operation. Neural Computation 14(12), 2857–2881 (2002)