PERGAMON

Neural Networks 11 (1998) 1571–1588

Neural Networks

Contributed article

The recursive deterministic perceptron neural network

Mohamed Tajine*, David Elizondo

LSIIT (CNRS URA 1871), Université Louis Pasteur, Département d'Informatique, 7 rue René Descartes, 67084 Strasbourg cedex, France

Received 26 April 1997; accepted 6 March 1998

Abstract

We introduce a feedforward multilayer neural network which is a generalization of the single layer perceptron topology (SLPT), called recursive deterministic perceptron (RDP). This new model is capable of solving any two-class classification problem, as opposed to the single layer perceptron which can only solve classification problems dealing with linearly separable sets (two subsets X and Y of $\mathbb{R}^d$ are said to be linearly separable if there exists a hyperplane such that the elements of X and Y lie on the two opposite sides of $\mathbb{R}^d$ delimited by this hyperplane). We propose several growing methods for constructing a RDP. These growing methods build a RDP by successively adding intermediate neurons (IN) to the topology (an IN corresponds to a SLPT). Thus, as a result, we obtain a multilayer perceptron topology, which together with the weights, are determined automatically by the constructing algorithms. Each IN augments the affine dimension of the set of input vectors. This augmentation is done by adding the output of each of these INs, as a new component, to every input vector. The construction of a new IN is made by selecting a subset from the set of augmented input vectors which is LS from the rest of this set. This process ends with LS classes in almost $n-1$ steps where n is the number of input vectors. For this construction, if we assume that the selected LS subsets are of maximum cardinality, the problem is proven to be NP-complete. We also introduce a generalization of the RDP model for classification of m classes ($m > 2$) allowing to always separate m classes. This generalization is based on a new notion of linear separability for m classes, and it follows naturally from the RDP. This new model can be used to compute functions with a finite domain, and thus, to approximate continuous functions. We have also compared — over several classification problems — the percentage of test data correctly classified, or the topology of the 2 and m classes RDPs with that of the backpropagation (BP), cascade correlation (CC), and two other growing methods.

Keywords: Classification problems; Convex hull; Decision region; Linear separability; NP-completeness; Perceptron; Recursive deterministic perceptron; Transformation algorithm

1. Introduction

One of the biggest limitations of the single layer perceptron topology (SLPT) is its inability to handle classification problems dealing with non-linearly separable sets. We propose a multilayer generalization of this topology, called a recursive deterministic perceptron feedforward neural network, which allows us to always separate the two classes (even if the two classes are NLS).

The idea behind the construction of a RDP is to increment the dimension of the input vector, by using the outputs of a sequence of INs. This allows for additional degrees of liberty for transforming the original NLS problem into a LS one (two subsets X and Y of $\mathbb{R}^d$ are said to be linearly separable if there exists a hyperplane such that the elements X and Y lie on the two opposite sides of $\mathbb{R}^d$ delimited by this hyperplane). These INs are added progressively, one at each time step, and each of them corresponds to a SLPT. If at step i, the augmented classes are NLS, a new IN is added. This new IN is obtained by selecting a subset of points from the augmented set of input vectors. This subset has to be chosen in such a way as to be LS from the subset containing the rest of the augmented input vectors. The algorithm stops when the two classes become LS. We prove that the algorithm terminates, in the worse case, after $n-1$ steps, where n is the number of input patterns. The final network can be viewed as a composition of several SLPTs, and it corresponds to a multilayer perceptron topology.

To illustrate how the algorithm for constructing a RDP works, we will look at the NLS 2-input Exclusive-OR (XOR) problem. This problem consists on classifying the two classes $X = \{(0,0),(1,1)\}$ and $Y = \{(0,1),(1,0)\}$, which are NLS. To perform the NLS to LS transformation, we

* Corresponding author. Requests for reprints should be sent to M. Tajine, Université Louis Pasteur, Dept. d'Informatique, 7 rue René Descartes, 67084 Strasbourg Cedex, France. Tel.: 33-3-88-41-63-42; Fax: 33-3-88-60-26-54; E-mail: tajine@dpt-info.u-strasbg.fr

0893–6080/98/$ - see front matter
PII: S0893-6080(98)00054-9

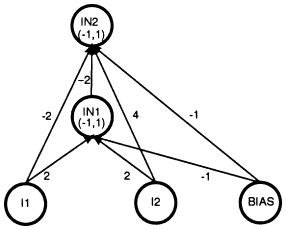

Fig. 1. RDP network for solving the XOR classification problem (the intermediate layer contains only one component, IN1, and the intermediate neuron IN2 corresponds to the output neuron of the RDP).

select a subset of patterns which is LS from the rest of the patterns. For example, we can select the subset $\{(0,0)\} \subset X \cup Y$, since $\{(0,0)\}$ and $\{(0,1),(1,0),(1,1)\}$ are LS (by the hyperplane $P_1 = \{(x_1,x_2) \in \mathbb{R}^2 | 2^*x_1 + 2^*x_2 - 1 = 0\}$). Thus, we create the intermediate neuron IN1 corresponding to the SLPT of weight vector $\mathbf{w} = (2, 2)$ and threshold $t = -1$ 'associated' to the hyperplane P_1. The output of IN1 allows us to add, to the input vectors, one component by assigning the value -1 to the input pattern $\{(0,0)\}$, and the value 1 to the remaining three input patterns $\{(0,1),(1,0),(1,1)\}$. So, this SLPT produces the following sets of augmented input vectors: $X' = \{(0, 0, \underline{-1}), (1, 1, \underline{1})\}$ and $Y' = \{(0, 1, \underline{1}), (1, 0, \underline{1})\}$. Now, X' and Y' are LS by the hyperplane $P_2 = \{(x_1,x_2,x_3) \in \mathbb{R}^3 | -2^*x_1 - 2^*x_2 + 4^*x_3 - 1 = 0\}$. Therefore, we create a second intermediate neuron IN2 (output neuron) which corresponds to the SLPT with the weight vector $\mathbf{w} = (-2, -2, 4)$, and threshold $t = -1$ 'associated' to the hyperplane P_2. The final result is a two layer RDP neural network solving the XOR classification problem since the output value of this neural network is -1 for the vector patterns $\{(0,0),(1,1)\}$, and 1 for the remaining vector patterns $\{(0,1),(1,0)\}$ (Fig. 1 shows a graphical representation of this RDP neural network).

Several people have studied growing algorithms, for constructing neural networks, that add new INs in the course of learning. The CC neural network presented in (Fahlman and Lebiere, 1990), illustrates such technique. This model is an example of input-addition-while-learning models, which is trained on an IN until no significant error reduction occurs after a certain number of training cycles (a user definable parameter). At this point, if the performance of the network is acceptable, the learning process is stopped, otherwise, a new IN is added and the learning process cycle is continued until the user is satisfied with the performance level. Since this algorithm uses BP (Rumelhart et al., 1986a, b) as a learning method, its convergence is not guaranteed.

In (Gallant, 1990) the author discusses a modification of the perceptron algorithm to enable a LS subset selection. This approach takes into consideration the fact that the perceptron algorithm is well behaved for discrete data. Thus, the inputs and the single output unit are restricted to take values in $\{-1,+1\}$. The new units are chosen based on the LS subset of maximum cardinality obtained after a number of cycles through the training process. This method will fit the data with arbitrarily high probability, provided enough iterations are taken while adding a new input. Thus, as in the previous model, its convergence is not guaranteed.

A geometrically based approach for building multilayer feedforward neural networks for pattern classification is presented in Bose and Garga (1993). This method can determine the topology of a network (in terms of number of layers, number of intermediate neurons, and connection weight values) based on both the convex hulls and the Voronoi diagram of the training patterns. The convex hulls are used for testing the linear separability between a set of points, and the Voronoi diagram to provide the hyperplane to linearly separate a set of NLS points. Each hyperplane is represented by an intermediate neuron. No explicit guarantee of convergence for this method is provided by the authors. The network topologies presented for four toy classification problems are over-loaded in terms of intermediate neurons. For example, the authors propose a topology with eight INs, for a classification problem composed of four input patterns, and another topology with thirteen INs for a five input pattern problem. It should be highlighted that the RDP will produce, in the worse case, a topology containing $n - 1$ INs, where n corresponds to the number of input patterns.

Another geometrically based growing method for training binary neural networks (BNN) called expanding-and-truncate learning (ETL) is proposed in Kim and Park (1995). The ETL algorithm for a three layer BNN is guaranteed to converge and can automatically determine the number of intermediate neurons in the middle layer. The weights and threshold values are of type integer. The algorithm decomposes a NLS problem into a set of LS functions based on a geometrical analysis of the training input patterns. Each LS function corresponds to a separating hyperplane, and is represented as a hidden unit. Once the algorithm has found the required separating hyperplanes, an output unit is used to combine the outputs of the hidden units. According to the authors, this algorithm has a greater learning speed than BP. As mentioned earlier, this method is limited to binary classification problems and integer weight vectors.

Another growing method is presented in Campbell and Perez Vincente (1995). The authors propose a procedure for constructing feedforward neural networks for binary classification tasks with binary or analogue data. For analogue input data and real value weights, the convergence of the method is restricted to input vectors of the same Euclidean length. This is a limiting restriction since most input vectors in real world data sets are of variable length if the components can take any value. A thermometer approach is suggested in order to convert a problem with analogue inputs into digital ones. The outputs of the networks are restricted to discrete values. This method has some points in common with the generalization strategy proposed for our general NLS to LS transformation method, since it also searches for LS subsets from within a NLS set of points. However,

in our method we have no restrictions for problems with analogue valued input and output units.

This paper is divided into six sections. In Section 2 we give some standard notations and definitions, and some general properties related to these notations and definitions. In Section 3 we present the LS transformation algorithms for two-class classification problems and we study the complexity of these algorithms. In Section 4 we present a generalization of the two-class RDP to m classes. In Section 5 both the 2 and m RDPs are evaluated by comparing the results obtained on six classification problems (in terms of topology size and percentage of test data correctly classified), to those obtained by the CC, BP, Rulex, and RuleNeg methods. In Section 6, we give some concluding remarks together with a discussion about the results obtained in this paper and some future work. The proofs of theorems and propositions are included in the Appendix A.

2. Background

In this section we introduce some of the standard notions used throughout this paper, together with some definitions and properties. In all of the following, we refer to a point as a vector.

2.1. Preliminaries

We use the following standard notions: Let E and F be two subsets of $\mathbb{R}^d$, Card(E) stands for the cardinality of E. $E\backslash F$ is the set of elements which belong to E and do not belong to F; $-E$ is the set of elements of the form $-\mathbf{e}$ with $\mathbf{e}\in E$; $E\oplus F$ is the set of elements of the form $\mathbf{e}+\mathbf{f}$ with $\mathbf{e}\in E$ and $\mathbf{f}\in F$; $E\ominus F$ stands for $E\oplus(-F)$. If $\mathbf{u}=(u_1,\ldots,u_d),\mathbf{v}=(v_1,\ldots,v_d)\in\mathbb{R}^d$, then $\mathbf{u}^{\mathrm{T}}\mathbf{v}$ stands for $u_1v_1+\ldots+u_dv_d$; $|\mathbf{u}|$ stands for $\mathbf{u}^{\mathrm{T}}\mathbf{u}$; and $\mathbf{u}(j)=u_j$ (i.e. $\mathbf{u}(j)$ is the jth component of $\mathbf{u}$); $\Pi_{\{i_1,\ldots,i_k\}}\mathbf{u}=(u_{i_1},\ldots,u_{i_k})$ (i.e. projection over a space of dimension k) and by extension, if $S\subset\mathbb{R}^d$ then $\Pi_{\{i_1,\ldots,i_k\}}S=\{\Pi_{\{i_1,\ldots,i_k\}}\mathbf{x}|\mathbf{x}\in S\}$ let $r\in\mathbb{R}$, $\mathrm{Adj}(\mathbf{u},r)=(u_1,\ldots,u_d,r)$ and by extension, if $S\subset\mathbb{R}^d$ then $\mathrm{Adj}(S,r)=\{\mathrm{Adj}(\mathbf{x},r)|\mathbf{x}\in S\}$; for $E\subset\mathbb{R}^d$ and $F\subset\mathbb{R}^{d+1}$ we define $\mathrm{Im}(E,F)=\{(x_1,\ldots,x_d,x_{d+1})\in F|(x_1,\ldots,x_d)\in E\}$. $\mathcal{P}(\mathbf{w},t)$ stands for the hyperplane $\{\mathbf{x}\in\mathbb{R}^d|\mathbf{w}^{\mathrm{T}}\mathbf{x}+t=0\}$ of $\mathbb{R}^d$; $\mathcal{S}_m$ stands for the set of permutations of $\{1,\ldots,m\}$; let $\mathbf{p}_1,\mathbf{p}_2$ be two points in $\mathbb{R}^d$, the set $\{t\mathbf{p}_1+(1-t)\mathbf{p}_2|0\le t\le 1\}$ is called the segment between $\mathbf{p}_1,\mathbf{p}_2$ and is denoted by $[\mathbf{p}_1,\mathbf{p}_2]$.

2.2. Some definitions and properties

In this section, we introduce the notions of convex hull, linear separability, and a new notion for enhancing the SLPT.

2.2.1. Linear separability

Definition 1, *Two subsets X and Y of* $\mathbb{R}^d$ *are said to be LS if there exists a hyperplane* $\mathcal{P}(\mathbf{w},t)$ *in* $\mathbb{R}^d$, *such that*: ($\forall\mathbf{x}\in X$, $\mathbf{w}^{\mathrm{T}}\mathbf{x}+t>0$ and $\forall\mathbf{y}\in Y$, $\mathbf{w}^{\mathrm{T}}\mathbf{y}+t<0$) or ($\forall\mathbf{x}\in X$, $\mathbf{w}^{\mathrm{T}}\mathbf{x}+t<0$ and $\forall\mathbf{y}\in Y$, $\mathbf{w}^{\mathrm{T}}\mathbf{y}+t>0$).

In the following we will denote the fact that X and Y are LS by $X\|Y$ *or* $X\|Y(\mathcal{P}(\mathbf{w},t))$ *if we want to specify the hyperplane which linearly separates X and Y. A discussion on different methods for testing linear separability can be found in* Tajine and Elizondo (1996); Elizondo (1997).

Intuitively, if X and Y are LS by the hyperplane P of $\mathbb{R}^d$, then P separates the space $\mathbb{R}^d$ in two 'opposite' regions with the points of X in one of the regions, and those of Y in the other.

Definition 2. *A subset D of* $\mathbb{R}^d$ *is said to be convex if, for any two points* $\mathbf{p}_1$ and $\mathbf{p}_2$ *in D, the segment* $[\mathbf{p}_1,\ \mathbf{p}_2]$ *is entirely contained in D.*

Definition 3. *Let S be a subset of* $\mathbb{R}^d$, *the convex hull of S, denoted by CH*(S), *is the smallest convex subset of* $\mathbb{R}^d$ *containing S.*

Proposition 4. (Preparata and Shamos, 1985) *Let S be a subset of* $\mathbb{R}^d$ $CH(S)=\{t_1\mathbf{x}_1+\ldots+t_k\mathbf{x}_k|\mathbf{x}_1,\ldots,\mathbf{x}_k\in S,\ t_1,\ldots,t_k\in[0,1]$ and $t_1+\ldots+t_k=1\}$, *and if S is finite, then there exists* $\mathbf{a}_1,\ldots,\mathbf{a}_n\in\mathbb{R}^d$ and $b_1,\ldots,b_n\in\mathbb{R}$ such that $CH(S)=\{\mathbf{x}\in\mathbb{R}^d|\mathbf{a}_i^{\mathrm{T}}\mathbf{x}\ge b_i$ for $1\le i\le n\}$.

Theorem 5. *Let X,Y be two finite subsets of* $\mathbb{R}^d$, *then* $X\|Y$ *iff* $CH(X)\cap CH(Y)=\varnothing$.

A more general result than the one shown in Theorem 5 can be found in Stoer and Witzgall (1970).

Proposition 6. *Let S be a finite nonempty subset of* $\mathbb{R}^d$, *then there exists* $\mathbf{x}\in S$ such that $(S-\{\mathbf{x}\})\|\{\mathbf{x}\}$.

Definition 7. *The points of the set S of* $\mathbb{R}^d$ *are said to be affinely independent if the dimension of the vectorial subspace generated by* $\{\mathbf{x}-\mathbf{a}|\mathbf{x}\in S\}$ *is equal to Card*(S) -1, *where* $\mathbf{a}$ *is a point in S.*

The following proposition clarifies the relationship between linear separability and affine dimension, and it is used to prove Theorems 12, 13, 14 and 15.

Proposition 8. *Let S be a finite subset of points of* $\mathbb{R}^d$, *and let ns(S) = Card({{A,S\A}; A* $\subset$ *S and A*$\|$(*S\A*)}), *then Card*(S) $\le ns(S)\le 2^{Card(S)-1}$. *Moreover, ns*($S$) = *Card*($S$) *iff there exists a straight line D such that all points of S are on D* (i.e. $S\subseteq D$). $ns(S)=2^{Card(S)-1}$ *iff a set of affinely independent points and thus, Card*(S) $\le d+1$.

2.2.2. The recursive deterministic perceptron

We define now the RDP which is a generalization of the SLPT introduced by Rosenblatt (1962). As highlighted in the introduction, one of the biggest limitations of the SLPT is that it cannot handle NLS problems. This is a drastic restriction since most classification problems are NLS.

The RDP overcomes this limitation. This new formalism is capable of always separating, in a deterministic manner, any two classes by recursively adding the output values of a series of INs to the input vector. These INs have a function similar to that of the hidden neurons in the BP algorithm. That is to say, they help to transform a NLS problem into a LS one. Each IN corresponds to a SLPT. This is an advantage over BP since the RDP constructing algorithm is guaranteed to always converge, and its embedded knowledge can always be expressed as a finite union of open polytopes.

Definition 9. *A recursive deterministic perceptron* (*RDP*) *P on* $\mathbb{R}^d$ *is a sequence* $[(\mathbf{w}_0, t_0, a_0, b_0), \ldots, (\mathbf{w}_n, t_n, a_n, b_n)]$ such that $\mathbf{w}_i \in \mathbb{R}^{d+i}$ *and* $t_i, a_i, b_i \in \mathbb{R}$, *and* $a_i < b_i$ *for* $0 \le i \le n$

- $(\mathbf{w}_i, t_i, a_i, b_i)$ for $0 \le i \le n$, *is called an Intermediate Neuron* (*IN*) *of the RDP P* (i.e. *an IN of RDP corresponds to a SPLT*).
- *height*(*P*) *corresponds to the number of INs in P* (i.e. *height*$(P) = n + 1$).
- *P*(*i*,*j*) *is the RDP* $[(\mathbf{w}_i, t_i, a_i, b_i), \ldots, (\mathbf{w}_j, t_j, a_j, b_j)]$ for $0 \le i \le j \le$ *height*$(P) - 1$. *So, P*(*i*,*j*) *is a RDP on* $\mathbb{R}^{d+1}$ *and P*(0,*n*) = *P*.

Definition 10. (*Semantic of RDP*)

Let P be a RDP on $\mathbb{R}^d$. *We associate to P the function* $\mathcal{F}(P)$, *defined almost everywhere in* $\mathbb{R}^d$, *such that*:

if height $(P) = 1$ (i.e. $P = [(\mathbf{w}, t, a, b)]$) *then*:

$$\mathcal{F}(P)(\mathbf{y}) = \begin{cases} a & \text{if } \mathbf{w}^{\mathrm{T}}\mathbf{y} + t < 0 \\ b & \text{if } \mathbf{w}^{\mathrm{T}}\mathbf{y} + t > 0 \end{cases}$$

and if height$(P) > 1$ (i.e. $P = [(\mathbf{w}_0, t_0, a_0, b_0), (\mathbf{w}_1, t_1, a_1, b_1), \ldots, (\mathbf{w}_n, t_n, a_n, b_n)]$ for $n \ge 1$) *then*:

$$\mathcal{F}(P)(\mathbf{y}) = \begin{cases} \mathcal{F}(P(1,n))(\mathrm{Adj}(\mathbf{y}, a_0)) & \text{if } \mathbf{w}_0^{\mathrm{T}}\mathbf{y} + t_0 < 0 \\ \mathcal{F}(P(1,n))(\mathrm{Adj}(\mathbf{y}, b_0)) & \text{if } \mathbf{w}_0^{\mathrm{T}}\mathbf{y} + t_0 > 0. \end{cases}$$

Definition 11. *Let X,Y be two subsets of* $\mathbb{R}^d$ *and let P be a RDP on* $\mathbb{R}^d$, *X and Y are said to be linearly separable by P, if* $(\forall \mathbf{x} \in X,\ \mathcal{F}(P)(\mathbf{x}) = c_1)$, and $(\forall \mathbf{y} \in Y,\ \mathcal{F}(P)(\mathbf{y}) = c_2)$, *where* $\{c_1, c_2\} = \{a_n, b_n\}$; *we denote this by* $X \|_P Y$.

Notation:

Let $P = [(\mathbf{w}_0, t_0, a_0, b_0), \ldots, (\mathbf{w}_n, t_n, a_n, b_n)]$ be a RDP on $\mathbb{R}^d$; we refer to the set $\mathcal{S}(P) = \{\mathbf{x} \in \mathbb{R}^d | \mathcal{F}(P)(\mathbf{x}) = b_n\}$, as the decision region of P [i.e. if $X \|_P Y$, then either $X \subseteq \mathcal{S}(P)$ or $Y \subseteq \mathcal{S}(P)$]. $\mathcal{S}(P)$ represents the knowledge embedded in the RDP P.

Remarks:

- $X \| Y(\mathcal{P}(\mathbf{w}, t))$ iff $X \|_{[(\mathbf{w}, t)]} Y$. In this case the RDP is a SLPT.
- If $P = [(\mathbf{w}_0, t_0, a_0, b_0), \ldots, (\mathbf{w}_n, t_n, a_n, b_n)]$ is a RDP on $\mathbb{R}^d$, then $\mathcal{F}(P)$ is defined on $\mathbb{R}^d \backslash Res(P)$ where $\mathrm{Res}(P) = E_0 \cup \ldots \cup E_n$ with:

  $E_0 = \{\mathbf{x} \in \mathbb{R}^d | \mathbf{w}_0^{\mathrm{T}}\mathbf{x} + t_0 = 0\}$ and $E_i = \{\mathbf{x} \in \mathbb{R}^d | \mathbf{w}_i(1)\mathbf{x}(1) + \ldots + \mathbf{w}_i(d)\mathbf{x}(d) + \mathbf{w}_i(d+1)\epsilon_1 + \ldots + \mathbf{w}_i(d+1)\epsilon_i + t_i = 0$ for $(\epsilon_1, \ldots, \epsilon_i) \in \{a_0, b_0\} \times \ldots \times \{a_{i-1}, b_{i-1}\}$ for $1 \le i \le n$. Thus, $\mathcal{F}(P)$ is definable almost everywhere, since *Res*(*P*) is a finite union of hyperplanes in $\mathbb{R}^d$ and, therefore, *Res*(*P*) is a negligible subset of $\mathbb{R}^d$.
- In most of the other neural models $a_i, b_i \in \{-1, 0, 1\}$, in our model, they can be assigned any value provided that $a_i < b_i$.
- The function $\mathcal{F}([(\mathbf{w}_0, t_0, a_0, b_0), \ldots, (\mathbf{w}_n, t_n, a_n, b_n)])$ can be expressed as a composition of the functions $\mathcal{F}([(\mathbf{w}_0, t_0, a_0, b_0)]), \ldots, \mathcal{F}([(\mathbf{w}_n, t_n, a_n, b_n)])$.
- A RDP $P = [(\mathbf{w}_0, t_0, a_0, b_0), \ldots, (\mathbf{w}_n, t_n, a_n, b_n)]$ on $\mathbb{R}^d$ of height n can be represented geometrically by a labeled acyclic graph with $\{I_1, I_2, \ldots, I_d, IN_0, IN_1, \ldots, IN_n, BIAS\}$ as a set of nodes (neurons). The node I_j corresponds to the jth component of the input vector, the node IN_j corresponds to the jth IN of the RDP, and the node *BIAS* corresponds to the threshold (all the edges, and only the IN nodes of the graph are labeled).

  The edges (connections) and labels of the graph are defined as follows:
- There exists an edge from I_j to IN_k iff $\mathbf{w}_k(j) \neq 0$, and this edge is labeled by $\mathbf{w}_k(j)$.
- There exists an edge from IN_i to IN_j iff $i < j$ and $\mathbf{w}_j(d+1) \neq 0$, and this edge is labeled by $\mathbf{w}_j(d+1)$.
- There exists an edge from *BIAS* to IN_j iff $t_j \neq 0$, and this edge is labeled by t_j.
- No edges exist between the I_js, neither between the I_js and the *BIAS*.
- The nodes IN_i are labeled by the couple (a_i, b_i), and the other nodes are not labeled.

We refer to the graph, without the labeling, as the topology of the RDP.

Example

To illustrate Definitions 9 and 10, we can look at the RDP

$P = [((2,2), -1, -1, 1), ((-2, -2, 4), -1, -1, 1)]$ which is a solution to the XOR classification problem presented earlier (height$(P) = 2$).

- The function $\mathcal{F}(P)$ associated to P is defined by:

$$\mathcal{F}(P)(z_1,z_2)=\begin{cases}\mathcal{F}(P)(1,1))(\mathrm{Adj}((z_1,z_2),-1)) & \text{if } 2z_1+2z_2-1<0\\ \mathcal{F}(P)(1,1))(\mathrm{Adj}((z_1,z_2),1)) & \text{if } 2z_1+2z_2-1>0.\end{cases}$$

So, $\mathcal{F}(P)(z_1, z_2)=\mathcal{F}([((-2, -2, 4), -1, -1, 1)])(z_1, z_2\mathcal{F}([((2, 2), -1, -1, 1)])(z_1, z_2))$.

$$\mathcal{F}(P)(z_1,z_2)=\begin{cases}-1 & \text{if } (-5<2z_1+2z_2<1) \text{ or } (2z_1+2z_2>3)\\ 1 & \text{if } (2z_1+2z_2<-5) \text{ or } (1<2z_1+2z_2<3)\end{cases}$$

Thus,

- $Res(P) = \{(z_1,z_2) \in \mathbb{R}^2 | 2z_1 + 2z_2 + t = 0$ for $t \in \{-5,1,3\}\}$. So $\mathcal{F}(P)$ is defined $\mathbb{R}^2 \backslash Res(P)$.
- The RDP P linearly separates $X = \{(0,0),(1,1)\}$ and $Y = \{(0,1),(1,0)\}$, because $\mathcal{F}(P)(0,0) = \mathcal{F}(P)(1,1) = -1$ and $\mathcal{F}(P)(0,1) = \mathcal{F}(P)(1,0) = 1$
- The decision region of P is $\mathcal{S}(P) = \{(z_1,z_2) \in \mathbb{R}^2 | 2z_1 + 2z_2 < -5$ or $1 < 2z_1 + 2z_2 < 3\}$. $\mathcal{S}(P)$ is presented in Fig. 2 by the black areas.
- The geometrical representation of P is given in Fig. 1 (in the introduction section).

3. NLS to LS transformation algorithms

In this section, we present two algorithms for building a RDP for linearly separating two finite disjoint set of patterns X,Y. This is done by 'augmenting' the affine dimension of the set of input vectors. Every possible RDP that linearly separates two classes can be obtained with the first algorithm by using some selection strategy. The second algorithm is obtained from the first algorithm by using a LS subset selection strategy. The first and the second algorithms stop, respectively, after at most $2^{\mathrm{Card}(X \cup Y)}$ and $\mathrm{Card}(X \cup Y) - 1$ steps.

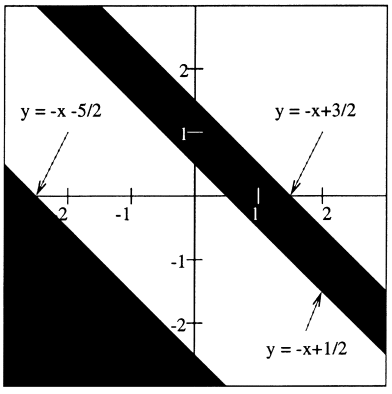

Fig. 2. Decision region for the RDP P corresponding to the XOR separability problem.

3.1. The general NLS to LS transformation algorithm

In this section, we present a general algorithm for building a RDP which transforms a NLS classification problem into a LS one. This algorithm allows us to find all possible topologies that linearly separate any given two-class classification problem, provided that the right selection strategy is used.

Notation:

Let $X,Y \subset \mathbb{R}^{d+k}$, the predicate $nmcc_d(X,Y)$ (not maximum common component) corresponds, intuitively, to the fact that X is not a maximal subset (with respect to the inclusion) of Y, having the same value in any of the k last components. This predicate is formally defined by $nmcc_d(X,Y)$ is: $((X \subset Y)$ and $(\forall j \in \{1,\ldots,k\}, \forall \mathbf{x} \in X, \exists \mathbf{y} \in (Y \backslash X)$ such that $\mathbf{y}(d+j) = \mathbf{x}(d+j)))$.

For example, if $Y = \{(1,1,4, -2), (2,3,3,2), (4,1,3,2), (2,3,4, -2), (1,1,2,3)\}$ and $X = \{(1,1,\underline{4}, \underline{-2}), (2,3,\underline{3},\underline{2})\}$, then we have $nmcc_2(X,Y)$ because the elements $(2,3,\underline{4}, \underline{-2})$, $(4,1,\underline{3},\underline{2}) \in (Y \backslash X)$ since they have the same value as X for the 3rd or 4th components.

The predicate $nmcc_d$ will be used as a termination control in the general NLS to LS transformation algorithm.

Table 1 presents the general NLS to LS transformation algorithm by building RDPs.

In Theorems 12 and 13, we prove the correctness and the guarantee of termination of the algorithm presented in Table 1. The variables X_i, Y_i, Z_i, $\mathbf{w}_i$, t_i, and S_i used in the following, correspond to those defined in this algorithm.

Theorem 12. *If X_i and Y_i are NLS then there exists $Z_i \neq \varnothing$ such that $nmcc_d(Z_i,S_i)$, and $Z_i \| (S_i \backslash Z_i)(\mathcal{P}(\mathbf{w}_i, t_i))$. This means that we can always find a LS subset from two NLS sets.*

Theorem 13. *if $i \neq j$ then $\Pi_{\{1,\ldots,d\}} Z_i \neq \Pi_{\{1,\ldots,d\}} Z_j$. Thus, the algorithm terminates after $n - 1$ steps, where $n \leq 2^{Card(X \cup Y)} - 1$, with LS sets. Therefore, there exists*

Table 1
General NLS to LS transformation algorithm

GNLS2LS (*X,Y,a,b*)
-data: two disjoint finite subsets *X,Y* of $\mathbb{R}^d$, and $a,b \in \mathbb{R}$ with $a < b$.
-result: A RDP $[(\mathbf{w}_0, t_0, a, b), \ldots, (\mathbf{w}_{n-1}, t_{n-1}, a, b)]$
which transforms *X* and *Y* into two LS classes (We obtain a RDP linearly separating *X,Y*, by adding one IN to the RDP constructed by this algorithm. This IN corresponds to the output neuron)
BEGIN
$S_0 := X \cup Y$; $X_0 := X$; $Y_0 := Y$; $i := 0$
WHILE *not*$(X_i \| Y_i)$ **DO**
BEGIN
Select $Z_i \subset S_i$ such that $nmcc_d(Z_i, S_i)$ and $Z_i \| (S_i \backslash Z_i)$ $(\mathcal{P}(\mathbf{w}_i, t_i))$;
$Z' := Adj(Z_i, a)$;
$Z'' := Adj(S_i \backslash Z_i, b)$;
$S_{i+1} := Z' \cup Z''$;
$X_{i+1} := Im(X_i, S_{i+1})$;
$Y_{i+1} := Im(Y_i, S_{i+1})$;
$i := i + 1$;
END;
END;

$\mathbf{w}_n \in \mathbb{R}^{d+n}, t_n \in \mathbb{R}$ *such that* $X_n \| Y_n (\mathcal{P}(\mathbf{w}_n, t_n))$. *So,* $[(\mathbf{w}_0, T_0, a, b), \ldots, (\mathbf{w}_{n-1}, t_{n-1}, a, b), (\mathbf{w}_n, t_n, a, b)]$ *is a RDP linearly separating X and Y where* $[(\mathbf{w}_0, t_0, a, b), \ldots, (\mathbf{w}_{n-1}, t_{n-1}, a, b)]$ *is the RDP constructed by the algorithm in* Table 1.

Remark:

Any RDP for linearly separating the classes *X* and *Y* can be found with this algorithm by using some LS subset selection strategy. Therefore, if we are given a certain network topology that can linearly separate *X* and *Y*, we can always construct, with the algorithm in Table 1, a RDP with the same topology by using a LS subset selection strategy. This selection strategy is embedded into the topology of the network. Thus, by decomposing the topology into each of its intermediate neurons, we can obtain the LS subsets necessary to create a given RDP.

3.2. The specialized NLS to LS transformation algorithm

3.2.1. Description of the algorithm

In this subsection we present an algorithm (Table 2) which is a specialization of the **GNLS2LS**. This algorithm uses a selection strategy for choosing LS subsets that

Table 2
Specialized NLS to LS transformation algorithm

SNLS2LS (*X,Y,a,b*)
-data: two disjoint finite subsets *X,Y* of $\mathbb{R}^d$, and $a,b \in \mathbb{R}$ with $a < b$.
-result: A RDP $[(\mathbf{w}_0, t_0, a, b), \ldots, (\mathbf{w}_{n-1}, t_{n-1}, a, b)]$
which transforms *X* and *Y* into two LS classes (We obtain a RDP linearly separating *X,Y*, by adding one IN to the RDP constructed by this algorithm. This IN corresponds to the output neuron)
$i := 0$; $X_0 := X$; $Y_0 := Y$; $X'_0 := X$; $Y'_0 := Y$; $S_0 = X \cup Y$;
WHILE *not*$(X_i \| Y_i)$ **DO**
BEGIN
SELECT: Select a non-empty subset Z_i from X'_i or from Y'_i (if it exists)
such that $Z_i \| S_i \backslash Z_i (\mathcal{P}(\mathbf{w}_i, t_i))$(i.e.$(Z_i \subset X'_i$ or $Z_i \subset Y'_i)$ *and* $Z_i \| (S_i \backslash Z_i))(\mathcal{P}(\mathbf{w}_i, t_i))$;
CASE $Z_i \subset X_i'$:
$S_{i+1} := Adj(Z_i, a) \cup Adj(S_i \backslash Z_i, b)$;
$X'_{i+1} := Im(X'_i, S_{i+1}) \backslash Im(Z_i, S_{i+1})$;
$Y'_{i+1} := Im(Y'_i, S_{i+1})$;
$X_{i+1} := Im(X_i, S_{i+1})$;
$Y_{i+1} := S_{i+1} \backslash X_{i+1}$;
$i := i + 1$;
CASE $Z_i \subset Y'_i$:
$S_{i+1} := Adj(Z_i, b) \cup Adj(S_i \backslash Z_i, a)$;
$Y'_{i+1} := Im(Y'_i, S_{i+1}) \backslash Im(Z_i, S_{i+1})$;
$X'_{i+1} := Im(X'_i, S_{i+1})$;
$X_{i+1} := Im(X_i, S_{i+1})$;
$Y_{i+1} := S_{i+1} \backslash X_{i+1}$;
$i := i + 1$;
END;

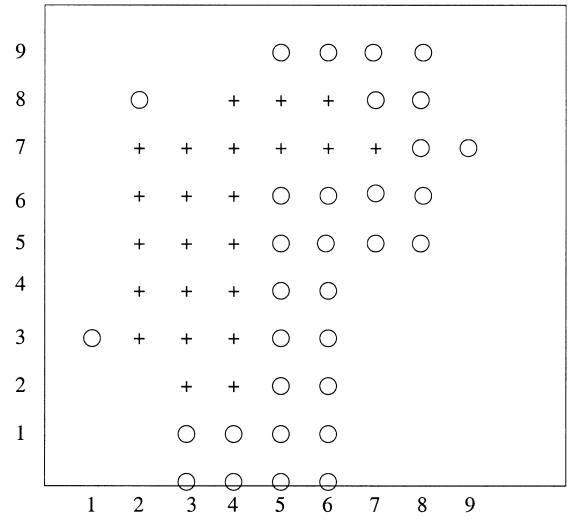

Fig. 3. 2D plot of the two-class classification problem (+ = class 1, ○ = class 2).

constrains the subsets to containing elements from only one of the two classes. The algorithm guarantees obtaining a RDP, which allows linear separation of any two-class classification problem, in at the most $n-1$ steps where n is the number of input patterns. We refer to it as the 'specialized' NLS to LS transformation algorithm. We also prove that, the process of selecting at each step a LS subset of maximum cardinality, is NP-Complete if the dimension of the input vector, d, is arbitrary, and polynomial otherwise.

In the following, $X_i, Y_i, Z_i, \mathbf{w}_i, t_i$, and S_i correspond to the variables defined in the algorithm described in Table 2;

Remarks:

Let $k_X(i) = \text{Card}(\{j | j < i \text{ and } Z_j \subset X_j\})$ and $k_Y(i) = \text{Card}(\{j | j < i \text{ and } Z_j \subset Y_j\})$, (i.e. $k_X(i) + k_X(i) = i$).

- If $\mathbf{x} \in X_i' \cup Y_i'$, then $\mathbf{x}$ contains exactly $k_X(i)$ times a and $k_Y(i)$ times b in its ith last components and all the elements of $X_i' \cup Y_i'$ have the same ith last components.
- If $\mathbf{x} \in (X_i \backslash X_i')$, then $\mathbf{x}$ contains $k_X(i) + 1$ times a and $k_Y(i) - 1$ times b in its last ith component. Further, $\mathbf{x}$ is different from $\mathbf{y} \in X_i' \cup Y_i'$ only in one component of the ith last component ($b \leftarrow a$).
- If $\mathbf{x} \in (Y_i \backslash Y_i')$, then $\mathbf{x}$ contains $k_X(i) - 1$ times a and $k_Y(i) + 1$ times b in its last ith component. Further, $\mathbf{x}$ is different from $\mathbf{y} \in X_i' \cup Y_i'$ only in one component of the ith last component ($a \leftarrow b$).

Theorem 14. *If X_i and Y_i are NLS, then there exists $Z_i \neq \varnothing$ such that ($Z_i \subset X_i'$, or $Z_i \subset Y_i'$), and $Z_i \| (S_i \backslash Z_i)(\mathcal{P}(\mathbf{w}_i, t_i))$. This means that we can always find a LS subset from two NLS sets.*

Theorem 15. *If $n-1$ is the number of steps of the algorithm in* Table 2, *then $n \leq \text{Card}(X \cup Y)$. Thus, there exists $\mathbf{w}_n \in \mathbb{R}^{d+n}, t_n \in \mathbb{R}$ such that $X_n \| Y_n(\mathcal{P}(\mathbf{w}_n, t_n))$, and therefore, the RDP $[(\mathbf{w}_0, t_0, a, b), \ldots, (\mathbf{w}_{n-1}, t_{n-1}, a, b), (\mathbf{w}_n, t_n, a, b)]$ linearly separates X and Y where $[(\mathbf{w}_0, t_0, a, b), \ldots, (\mathbf{w}_{n-1}, t_{n-1}, a, b)]$ is the RDP constructed by the algorithm in* Table 2. To minimize the number of steps in the specialized algorithm we will choose at each step Z_i having maximal cardinality. We will prove that this problem is NP-complete when both Card(X), Card(Y) and d are arbitrary. We refer to this problem as the *Max-separability* problem. This problem can be explored by using the *Open hemisphere* problem presented by Johnson and Preparata (1978) which can be stated as follows:

Given: positive integers d and M and a finite set $K \subseteq Q^d$

Question: does there exist a $\mathbf{w} \in \mathbb{R}^d$ such that $|\mathbf{w}| > 0$ and $\text{Card}(\{\mathbf{x} \in K : \mathbf{w}^T \cdot \mathbf{x} \geq 0\}) \geq M$? (where $|\mathbf{w}| = \mathbf{w}^T\mathbf{w}$).

Theorem 16.

- *The following problem is NP-complete*:

  Given: $d \in \mathbb{N}$, $X, Y \subset \mathbb{R}^d$ *such that* $X \cap Y = \varnothing$ *and* $M \in \mathbb{N}$

  Question: *does there exist* $\mathbf{w} \in \mathbb{R}^d$ *and* $t \in \mathbb{R}$ *such that* $(\forall \mathbf{x} \in X, \mathbf{w}^T\mathbf{x} + t > 0)$ *and* $(Card(\{\mathbf{y} \in Y | \mathbf{w}^T\mathbf{y} + t < 0\}) \geq M)$?
- *When the number d of dimensions is fixed, and if $n = Card(X)$, $n' = Card(Y)$ then there exists a polynomial time algorithm which solves the last problem in $O((nn')^d \log(nn'))$ time*.

3.2.2. Example

In this section we illustrate the use of the NLS to LS specialized transformation algorithm, presented in Table 2, by applying it to a NLS 2D classification toy example. The results obtained with several real world benchmarks are presented in Section 5. The NLS 2D classification problem consists of two classes X,Y:

$X = \{(3,2), (4,2), (2,3), (3,3), (4,3), (2,4), (3,4), (4,4),$
$(2,5), (3,5), (4,5), (2,6), (3,6), (4,6), (2,7), (3,7),$
$(4,7), (5,7), (6,7), (7,7), (4,8), (5,8), (6,8)\}$, and

$Y = \{(3,0), (4,0), (5,0), (6,0), (3,1), (4,1), (5,1), (6,1), (5,2),$
$(6,2), (1,3), (5,3), (6,3), (5,4), (6,4), (5,5), (6,5), (7,5),$
$(8,5), (5,6), (6,6), (7,6), (8,6), (8,7), (9,7), (2,8), (7,8),$
$(8,8), (5,9), (6,9), (7,9), (8,9)\}$

(see Fig. 3)

Once we apply the specialized NLS to LS algorithm in Table 2, with $a = 0$, and $b = 1$, to this problem, we obtain a

Table 3
LS subsets selected by the learning algorithm described in Table 1 applied to the two-dimensions, two-class classification problem

Step no.	Selected subset	Class
1	{(4,0),(5,0),(6,0),(4,1),(5,1),(6,1),(5,2),(6,2),(5,3),(6,3),(6,4),(7,5),(8,5),(7,6),(8,6),(8,7),(9,7),(8,8){}	X
2	{(2,8,0),(5,9,0),(6,9,0),(7,9,0),(8,9,0){}	X
3	{(3,0,0,0),(3,1,0,0),(1,3,0,0){}	X
4	{(3,2,0,0,0),(4,2,0,0,0),(2,3,0,0,0),(3,3,0,0,0),(4,3,0,0,0),(2,4,0,0,0)(3,4,0,0,0),(4,4,0,0,0),(2,5,0,0,0), (3,5,0,0,0),(4,5,0,0,0),(2,6,0,0,0),(3,6,0,0,0),(2,7,0,0,0),(3,7,0,0,0){}	Y
5	{(4,6,0,0,0,1),(4,7,0,0,0,1),(4,8,0,0,0,1),(5,8,0,0,0,1){}	Y
6	{(5,4,0,0,0,1,1),(5,5,0,0,0,1,1),(6,5,0,0,0,1,1),(5,6,0,0,0,1,1),(6,6,0,0,0,1,1)}	X

RDP containing seven INs which linearly separates *X* and *Y*. Table 3 shows the LS subsets selected for each IN (at each step, the selected LS subset was of maximal cardinality). Table 4 shows the weight vectors and thresholds found for each IN. A projection of the selected LS subsets used in the different steps for building the RDP is shown in Fig. 4. Fig. 5 shows the RDP topology found by the learning algorithm described in Table 2 for solving this problem. Notice that the INs are connected to the previous ones only if their connection has a value different than zero. We did not try to optimize the topology in terms of number of connections. One way to do this optimization is by choosing, at each construction step, a hyperplane for linearly separating the selected subset of input patterns from the remaining patterns, containing the greatest number of zero weight components.

4. Adapting the notion of linear separability from 2 to *m* classes

Based on a new notion of linear separability for *m* classes ($m \geq 2$), we adapt the RDP for two-class classification problems to any *m* classes NLS classification problem.

4.1. Linear separability for m classes

4.1.1. Definition of the linear separability for m classes

Definition 17. *Let* $X_1,\ldots,X_m \subset \mathbb{R}^d$ *and* $a_0 < a_1 < \ldots < a_m$. $X_1,\ldots,X_m$ *are said to be LS relatively to the ascending sequence of real numbers* $a_0,\ldots,a_m$ *if there exists a permutation* $\sigma \in \mathcal{S}_m$, $\exists \mathbf{w} \in \mathbb{R}^d$, $\exists t \in \mathbb{R}$ *such that* $\forall i, \forall \mathbf{x} \in X_{\sigma(i)}$, $a_{i-1} < \mathbf{w}^{\mathrm{T}}\mathbf{x} + t < a_i$.

Table 4
Weight vectors and threshold values obtained by the RDP network for each of the INs (at each step, $a_i = 0$, and $b_i = 1$)

i (step)	$\mathbf{w}_i$ (weight vector)	t_i (threshold)
1	(19.1, −12.4)	−4.0
2	(−3.5, 21.3, 0.0)	−15.0
3	(−6.7, −6.9, −1.0, 0.0)	4.0
4	(6.2, 1.8, 1.0, 1.0, 3.0)	−1.0
5	(21.6, −6.6, 1.0, 7.0, 8.0, −6.0)	2.0
6	(0.4, −1.5, −3.0, 0.0, −1.0, 2.0, 1.0)	−1.0
7	(5.8, 1.3, 4.0, 7.0, 5.0, 1.0, 3.0)	−5.0

Remarks:

Let $X_1,\ldots,X_m \subset \mathbb{R}^d$ and $a_0 < a_1 < a_2 < \ldots < a_m$,

- $X_1,\ldots X_m$ are LS relatively to $a_0,\ldots,a_m$ iff $CH(X_1),\ldots,CH(X_m)$ are LS relatively to $a_0,\ldots,a_m$.
- let $\sigma \in \mathcal{S}_m$. Put:

$$X^{\sigma} = \mathrm{Adj}(X_{\sigma(1)}, -a_0) \cup \mathrm{Adj}(X_{\sigma(2)}, -a_1) \ldots \cup \mathrm{Adj}(X_{\sigma(m)}, -a_{m-1}),$$

$$Y^{\sigma} = \mathrm{Adj}(X_{\sigma(1)}, -a_1) \cup \mathrm{Adj}(X_{\sigma(2)}, -a_2) \ldots \cup \mathrm{Adj}(X_{\sigma(m)}, -a_m),$$

then, $X_1,\ldots,X_m$ are LS relatively to $a_0,\ldots,a_m$ iff there exists $\sigma \in \mathcal{S}_m$ such that $X^{\sigma} \| Y^{\sigma}$. In other words, we reduce the problem of linear separability for *m* classes into the problem of linear separability for two classes. We do this by augmenting the dimension of the input vectors with the ascending sequence $a_0,\ldots,a_m$.

- If $X_1 \| X_2$ $(\mathcal{P}(\mathbf{w}, t))$ and $\alpha = \mathrm{Max}(\{|\mathbf{w}^{\mathrm{T}}\mathbf{x} + t|; \mathbf{x} \in (X_1 \cup X_2)\}$, then X_1, X_2 are LS relative to $-2*\alpha, 0, 2*\alpha$. Therefore, the classical notion of linear separability for two classes can be expressed with this new notion of linear separability.

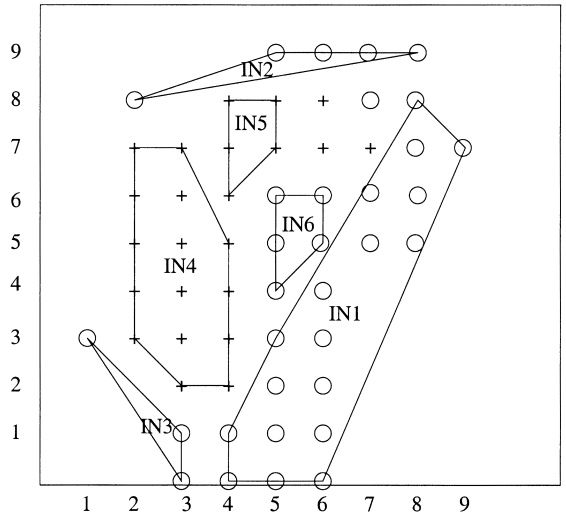

Fig. 4. Projection in a plane of the LS subsets selected to create the INs necessary to construct the RDP.

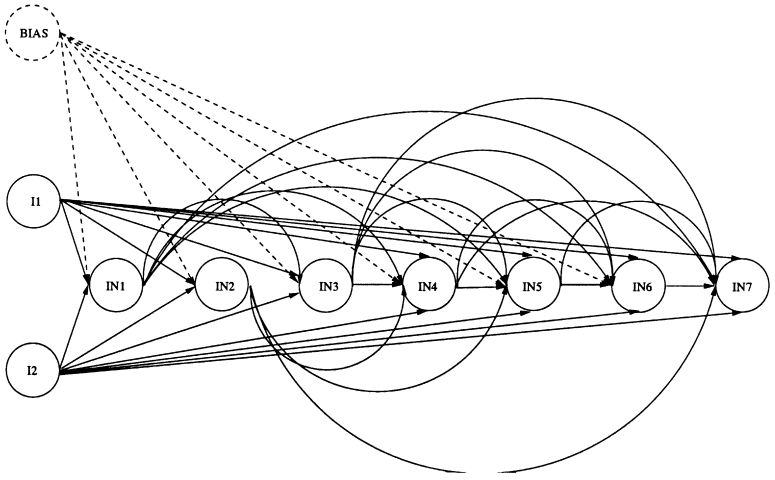

Fig. 5. The topology of the RDP for solving the two-dimensional classification problem (IN7 corresponds to the output neuron. This is a partially connected network, since there is no connection between nodes IN1 and IN2, IN2 and IN3, and IN2 and IN6 because they have zero as weight value).

Definition 18. *A sequence* $a_0,a_1,\ldots,a_n$ *of real numbers is said to be arithmetic if* $\exists b,\ h \in \mathbb{R}$ *such that* $\forall i \leq n, a_i = b + i*h$.

Proposition 19. *Let* $X_1,\ldots,X_m \subset \mathbb{R}^d$, $a,b \in \mathbb{R}, h,k > 0$ *and let* $a_i = a + ih, b_i = b + ik$, *for* $0 \leq i \leq m$, *then* $X_1,\ldots,X_m$ *are LS relative to* $a_0,\ldots,a_m$ *iff they are LS relatively to* $b_0,\ldots,b_m$. *In other words, the linear separability between m classes is independent of the arithmetic sequence*.

Definition 20. $X_1,\ldots,X_m \subset \mathbb{R}^d$ *are said to be LS if there exists* $a \in \mathbb{R}, h > 0$ *such that* $X_1,\ldots,X_m$ *are LS relative to* $a, a + h,\ldots,a + mh$.

With this new notion of linear separability, only one hyperplane is used to linearly separate m classes. This hyperplane is translated $m + 1$ times with the same translation vector.

4.1.2. Example

We present now an example of a three-class classification problem to illustrate this new notion of linear separability.

Let $X_1 = \{(0,0)\}$, $X_2 = \{(0,1),(1,0)\}$ $X_3 = \{(1,1)\}$, and $a_1 = 1$, $a_2 = 2$, $a_3 = 3$, and $a_4 = 4$. X_1, X_2, and X_3 are LS if there exist w_1,w_2,t such that:

$$1 < t < 2 \quad \text{Class 1} \tag{1}$$

$$2 < w_1 + t < 3 \quad \text{Class 2} \tag{2}$$

$$2 < w_2 + t < 3 \quad \text{Class 2} \tag{3}$$

$$3 < w_1 + w_2 + t < 4 \quad \text{Class 3} \tag{4}$$

Thus, $w_1 = w_2 = 1$, and $t = 3/2$ is a solution to this problem. Fig. 6 presents the hyperplanes, by dotted lines, that linearly separate the three classes.

4.2. The recursive deterministic perceptron for m classes

We will extend now the RDP defined for two-class classification problems to *m* classes by using this new notion of linear separability.

4.2.1. Definition of the RDP for m classes

Definition 21. *A RDP for m classes* (*m-RDP*) P *on* $\mathbb{R}^d$ *is a sequence* $[(\mathbf{w}_0, t_0, a_0, h_0, b_{0,1}, \ldots, b_{0,j_0}), \ldots, (\mathbf{w}_n, t_n, a_n, h_n, b_{n,1}, \ldots, b_{n,j_n})]$ *such that* $\mathbf{w}_i \in \mathbb{R}^{d+i}$, $t_i, a_i, h_i, b_{i,1}, \ldots, b_{i,j_i} \in \mathbb{R}$, *with* $h_i > 0, b_{i,1} < b_{i,2} < \ldots < b_{i,j_i}$, and $j_i \leq m$ for $0 \leq i \leq n$, and $j_n = m$.

- $(\mathbf{w}_i, t_i, a_i, h_i, b_{i,1}, \ldots, b_{i,j_i})$ *for* $0 \leq i \leq n$, *is called an Intermediate Neuron* (*IN*) *or m-SLPT of the m-RDP P* (i.e. *an IN of a m-RDP corresponds to a SPLT for m classes*).
- *height*(*P*) *corresponds to the number of INs in P* (i.e. *height*(*P*) $= n + 1$).
- $P(i,k)$ *is the RDP* $[(\mathbf{w}_i, t_i, a_i, h_i, b_{i,1}, \ldots, b_{i,j_i}, \ldots, (\mathbf{w}_k, t_k, a_k, h_k, b_{k,1}, \ldots, b_{k,j_k})]$ *for* $0 \leq i \leq k \leq$ *height*$(P) - 1$. *So,* $P(i,k)$ *is a RDP on* $\mathbb{R}^{d+i}$ *and* $P(0,n) = P$.

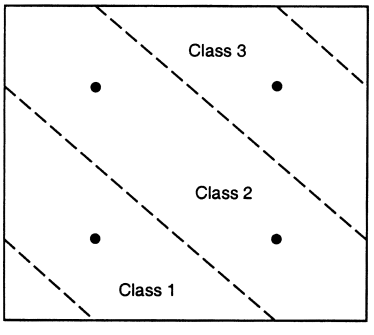

Fig. 6. Hyperplanes that linearly separate the three classes in the previous example.

Definition 22. (*Semantic of m-RDP*)

Let P a m-RDP on $\mathbb{R}^d$*, we associate to P the function* $\mathcal{F}(P)$ *defined almost everywhere on* $\mathbb{R}^d$*, such that*:

if height$(P) = 1$ (i.e. $P = [(\mathbf{w}_0, t_0, a_0, h_0, b_{0,1}, \ldots, b_{0,j_0})]$) *then*:

$$\mathcal{F}(P)(\mathbf{y}) = \begin{cases} b_{0,1} & \text{if } \mathbf{w}_0^{\mathrm{T}}\mathbf{y} + t_0 < a_0 \\ b_{0,i} & \text{if } a_0 + (i-1)h_0 < \mathbf{w}_0^{\mathrm{T}}\mathbf{y} + t_0 < a_0 + ih_0, \\ & \text{for } 1 \le i < j_0 - 1 \\ b_{0,j_0} & \text{if } \mathbf{w}_0^{\mathrm{T}}\mathbf{y} + t_0 > a_0 + (j_0 - 2)h_0 \end{cases}$$

and if height$(P) > 1$ (i.e. $P = [(\mathbf{w}_0,\ t_0,\ a_0,\ h_0,\ b_{0,1}, \ldots, b_{0,j_0}), \ldots, (\mathbf{w}_n,\ t_n,\ a_n,\ h_n, b_{n,1}, \ldots, b_{n,j_n})]$ *for* $n \ge 1$) *then*:

$$\mathcal{F}(P)(\mathbf{y}) = \begin{cases} \mathcal{F}(P(1,n))(\mathrm{Adj}(\mathbf{y}, b_{0,1})) & \text{if } \mathbf{w}_0^{\mathrm{T}}\mathbf{y}t_0 < a_0 \\ \mathcal{F}(P(1,n))(\mathrm{Adj}(\mathbf{y}, b_{0,i})) & \text{if } a_0 + (i-1)h_0 < \mathbf{w}_0^{\mathrm{T}}\mathbf{y} + t_0 < a_0 + ih_0 \text{ for } 1 \le i < j_0 - 1 \\ \mathcal{F}(P(1,n))(\mathrm{Adj}(\mathbf{y}, b_{0,j_0})) & \text{if } \mathbf{w}_0^{\mathrm{T}}\mathbf{y} + t_0 > a_0 + (j_0 - 2)h_0 \end{cases}$$

Definition 23. *The sets* $X_1,\ldots,X_m$ *are said to be LS by the m-RDP P, if* $\forall i \in \{1, \ldots, m\}(\forall \mathbf{x} \in X_i, \mathcal{F}(P)(\mathbf{x}) = c_i)$, *where* $\{c_1,\ldots,c_m\} = \{b_{n,1},\ldots,b_{n,m}\}$.

Remarks:

- A RDP can be viewed as a 2-RDP.
- All notions defined for a RDP, can be extended in a natural way to a *m*-RDP.
- It is evident that this new model can be used to compute functions with a finite domain and thus, to approximate continuous functions.
- In the same way as we have translated the test of linear separability for *m* classes to the classical test of linear separability for two classes, we can transform the problem of searching a *m*-RDP for linearly separating *m* classes to that of searching a RDP for linearly separating two classes. With this transformation, we double the number of input patterns. Thus, the number of INs can reach 2**n*, where *n* corresponds to the number of input patterns. In the following we propose a new algorithm which obtains a *m*-RDP containing $n - 1$ INs in the worse case.

4.2.2. The specialized NLS to LS transformation algorithm for m classes

We now present an extension of the specialized transformation algorithm from two to *m* classes. This extension is based on the new notion of linear separability for *m* classes described previously. Let $c \in \mathbb{R}, h > 0, b = -(m - (3/2))h$ and let

$$b_i = c + (m-i)b + \left(\frac{(m-1)(m-2)}{2} - \frac{(i-1)(i-2)}{2}\right)h$$

for $1 \le i < m$ and $b_m = c$.

Table 5 shows the specialized NLS to LS transformation algorithm for *m* classes. The sequence $b_1,\ldots,b_m$ will be used in this algorithm as outputs for the INs. We proceed as in the two-class specialized transformation algorithm. That is to say, at each step we select a LS subset which belongs to a single class and add an IN to the topology. The output of this new IN adds a new component to the input vector. Two cases for defining the output values of the INs are possible depending on the class to which the selected LS subset belongs:

1. If the selected LS subset belongs to the *j*th class, with $j < m$, we add to its input vector a new component with value b_j and we add to the rest of the input vector a new component with value b_{j+1}.
2. If the selected LS subset belongs to the last class (*m*th class), we add to its input vector a new component with value b_m and we add to the rest of the input vector a new component with value b_{m-1}.

In the following two theorems we prove the correctness and the termination of the algorithm presented in Table 5 which allows us to construct an *m*-RDP for linearly separating any given *m* classes. The variables used in the following two theorems correspond to those defined in the algorithm described in Table 5.

Theorem 24. *If* $X_1^i, \ldots, X_m^i$ *are NLS, then there exists* Z_i *such that* $(Z_i \subset X'^i_1$ *or* $\ldots$ *or* $Z_i \subset X'^i_m), Z_i \neq \varnothing$ *and* $Z_i \| (S_i \backslash Z_i)$.

Theorem 25. *If the algorithm in* Table 5 *stops, after* $n - 1$ *steps then* $n \le Card(X_1 \cup \ldots \cup X_m)$, *and there exists* $\mathbf{w} \in \mathbb{R}^{d+n}, t_n \in \mathbb{R}$ *such that* $X_1^i,\ldots,X_m^i$ *are LS relative to the arithmetic sequence* $a_0,\ldots,a_m$, *where* $a_0 = b - \frac{h}{2}$, $a_i = a_0 + ih$ *for* $1 \le i \le m$. *Therefore, the m-RDP* $[(\mathbf{w}_0,\ t_0,\ a_0,\ h_0,\ b_{0,1},\ b_{0,2}), \ldots, (\mathbf{w}_{n-1},\ t_{n-1},\ a_{n-1},\ h_{n-1},\ b_{n-1,1},\ b_{n-1,2}, (\mathbf{w}_n, t_n, a, h, b_1, \ldots, b_m)]$ *linearly separates* $X_1,\ldots,X_m$, *where* $[(\mathbf{w}_0 t_0,\ a_0,\ h_0,\ b_{0,1},\ b_{0,2}), \ldots, (\mathbf{w}_{n-1},\ t_{n-1},\ a_{n-1},\ h_{n-1},\ b_{n-1,1}, b_{n-1,2})]$ *is the RDP constructed by this algorithm with* $h_i = 2 * \max(\{|\mathbf{w}_i^{\mathrm{T}}\mathbf{x} + t_i|; \mathbf{x} \in X_1^i \cup \ldots \cup X_m^i\}), a_i = -h_i$, *and* $b_{j,1}, b_{j,2}$ *are* b_k, b_{k+1} *if at*

Table 5
Specialized NLS to LS transformation algorithm for m classes

SNLS2LS ($X_1,\ldots,X_m$)
-data; m disjoint finite subsets $X_1,\ldots,X_m$ of $\mathbb{R}^d$
-result: a RDP $[(\mathbf{w}_0,t_0,a_0,h_0,b_{0,1},b_{0,2}),\ldots,(\mathbf{w}_{n-1},t_{n-1},a_{n-1},h_{n-1},b_{n-1,1,},b_{n-1,2})]$
which transforms $X_1,\ldots,X_m$ into m LS classes (We obtain a m-RDP linearly separating $X_1,\ldots,X_m$, by adding one IN to the RDP constructed by this algorithm. This IN corresponds to the output neuron)
$i:=0; X_1^0:=X_1;\ldots;X_m^0:=X_m; X'^0_1:=X_1;\ldots;X'^0_m:=X_m; S_0=X_1\cup\ldots\cup X_m;$
WHILE $X_1^i,\ldots,X_m^i$ are NLS **DO**
BEGIN
SELECT: Select a non-empty subset Z_i from X'^i_i or … or from X'^i_m (if it exists) such that Z_i ($S_i\backslash Z_i$) are LS (i.e. ($Z_i\subset X'^i_1$ *or* … *or* $Z_i\subset X'^i_m$) and $Z_i\|(S_i\backslash Z_i))(\mathcal{P}(\mathbf{w}_i,t_i))$;
CASE:
Case $Z_i\subset X'^i_1$:
$S_{i+1}:=\mathrm{Adj}(Z_i,b_1)\cup\mathrm{Adj}(S_i\backslash Z_i,b_2);$
$X'^{i+1}_1:=Im(X'^i_1,S_{i+1})\backslash Im(Z_i,S_{i+1});$
$X'^{i+1}_2:=Im(X'^i_2,S_{i+1});\ldots;$
$X'^{i+1}_2:=Im(X'^i_m,S_{i+1});$
$X_1^{i+1}:=Im(X_1^i,S_{i+1});$
$X_2^{i+1}:=Im(X_2^i,S_{i+1});\ldots;$
$X_m^{i+1}:=Im(X_m^i,S_{i+1});$
$i:=i+1;$
……
Case $Z_i\subset X'^i_j$:
$S_{i+1}:=\mathrm{Adj}(Z_i,b_j)\cup\mathrm{Adj}(S_i\backslash Z_i,b_{j+1});$
$X'^{i+1}_1:=Im(X'^i_1,S_{i+1};\ldots;$
$X'^{i+1}_{j-1}:=Im(X'^i_{j-1},S_{i+1});$
$X'^{i+1}_j:=Im(X'^i_j,S_{i+1})\backslash Im(Z_i,S_{i+1});$
$X'^{i+1}_{j+1}:=Im(X'^i_{j+1},S_{i+1});\ldots;$
$X'^{i+1}_m:=Im(X'^i_m,S_{i+1});$
$X_1^{i+1}:=Im(X_1^i,S_{i+1});$
$X_2^{i+1}:=Im(X_2^i,S_{i+1});\ldots;$
$X_m^{i+1}:=Im(X_m^i,S_{i+1});$
$i:=i+1;$
……
Case $Z_i\subset X'^i_m$:
$S_{i+1}:=\mathrm{Adj}(Z_i,b_m)\cup\mathrm{Adj}(S_i\backslash Z_i,b_{m-1});$
$X'^{i+1}_1:=Im(X'^i_1,S_{i+1});\ldots;$
$X'^{i+1}_{m-1}:=Im(X'^i_{m-1},S_{i+1});$
$X'^{i+1}_m:=Im(X'^i_m,S_{i+1})\backslash Im(Z_i,S_{i+1});$
$X_2^{i+1}:=Im(X_2^i,S_{i+1});\ldots;$
$X_m^{i+1}:=Im(X_m^i,S_{i+1});$
$i:=i+1;$

step j, $Z_j\subset X_k{}'^j$ and $k\le m-1$ or b_{m-1},b_m if at step j, $Z_j\subset X_k{}'^j$.

4.2.3. Example

To illustrate how the specialized m-RDP works, we will look at the toy three-class NLS 2-input classification problem illustrated in Fig. 7. Results obtained for several real world benchmarks will be presented in Section 5. This toy problem consists on classifying the three classes $X=\{(1,2),(2,2),(4,2),(4,3)\}$, $Y=\{(2,5,1),(5,2),(2,3),(5,3)\}$, and $Z=\{(1,1),(3,2),(1,3),(3,3)\}$. For this problem, we select the values of $c=0$ and $h=1$. With these two values we obtain $b_1=-2.0$, $b_2=-0.5$, and $b_3=0.0$.

After applying the **SNLS2LS** algorithm (Table 5), we obtain the results shown in Table 6. We obtain a m-RDP linearly separating $X_1,\ldots,X_m$, by adding one IN to the RDP constructed by this algorithm. This IN corresponds to the output neuron of the RDP and has the weight vector (0,0,1,1,1,1,1,1,1,1), and the threshold value $t=2.5$, and it separates linearly the three classes relatively to the sequence $(-2,-1,0,1)$.

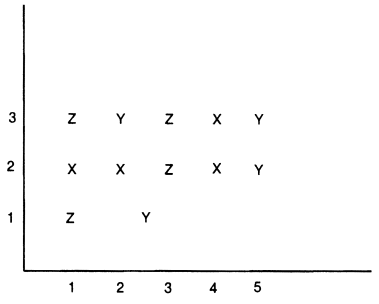

Fig. 7. 2D plot of the three-class classification problem (X = class 1, Y = class 2, and Z = class 3).

Table 6
RDPs obtained while transforming the NLS the three-class classification problem into a LS one

i (step)	Z_i (selected subset)	X'^i_j (class)	$\mathbf{w}_i$ (weight vector)	t_i (threshold)
1	{(5,2),(5,3){}	2	(−16,10)	47
2	{(4,2,0),3,3,0{}	1	(11,−9,36)	−51
3	{(3,2,0,−0.5),(3,3,0,−2){}	3	(−5.4,−25,26)	29
4	{(2,3,0,−0.5,−0.5){}	2	(−1,−7,−18,19,−14)	56
5	{(2.5,1,0,−0.5,−0.5,0){}	2	(1,7,7,2,−4,−3)	−3
6	{(1,1,0,−0.5,−0.5,0,0){}	3	(1.5,2,−1,0,0,0)	−1
7	{(1,3,0,−0.5,−0.5,0,0,−0.5){}	3	(10,−4,2,0,2,−6,2,−2)	2
8	{(1,2,−0.5,−0.5,0,0,−0.5,−0.5),(2,2,−0.5,−0.5,0,0,−0.5,−0.5)}	1	(4,−1,0,0,0,−2,0,−1,2)	0

5. Applications

In Sections 3 and 4, we presented our methods for constructing the RDP neural network topology, and used two toy classification examples to illustrate how these methods work. In this section, we present the results obtained after applying the specialized NLS to LS algorithm to six classification problems which include five benchmarks and a satellite image. These classification problems were used to test and validate our NLS to LS transformation algorithms and to compare their performance level with that of the BP and CC learning algorithms. For illustration purposes we also include a comparison (in terms of the topology size) of the results obtained with the RDP and two other rowing methods on four toy classification problems.

Table 7
Percentage of test data correctly classified for the RDP, BP, and CC learning algorithms

Problem	Algorithm	INs	Hidden units	Percentage
Soya C2	RDP	3		80.00
	BP	2		86.00
	CC	1		86.70
Soya C3	RDP	3		80.00
	BP	6		86.00
	CC	2		86.67
Circle	RDP	8		83.00
	BP	4		76.00
	CC	4		86.50
Iris-virginica	RDP	2		96.70
	BP	2		95.34
	CC	4		97.33
Iris-versicolor	RDP	3		94.00
	BP	3		94.67
	CC	5		96.67
Wine	RDP	3		90.00
	BP	0		90.00
	CC	0		97.60
Monks1	RDP	1		91.20
	BP	3		100.00
	CC	3		97.30
Monks3	RDP	3		81.48
	BP	6		88.65
	CC	4		87.60

5.1. Benchmark classification problems

The benchmark classification problems include the Soybean, the Circle, the Sonar (Gorman and Sejnowski, 1988), the Iris (Fisher, 1936; Gates, 1972; Dasarathy, 1980), the Wine, and the Monks (Thrun et al., 1991). The BP models contain, when necessary, a single hidden layer. The topologies obtained with the three learning methods (RDP, BP, and CC) are all fully interlayer connected.

Cross validation was applied on both the Iris and the Monks problems. On the latter, the six input decimal data set was used instead of the binary equivalent one containing 17 inputs. The Wine data set, consisting of three wine classes, was used for testing the m-class RDP.

5.1.1. Results

The results obtained with the RDP, BP, and CC on these six classification problems, are presented in Table 7.

The percentage of test data correctly classified by the RDP was higher than that of the BP for the Circle data set. The results obtained with both the RDP and the BP for the Iris data sets are very similar. Each of the three classes in the Wine data set is LS with respect to the union of the other two classes, but the three classes are not LS relative to the new notion of linear separability. Thus, we need four INs, one per class, and one corresponding to the output unit of the m-RDP. The same percentage of test data correctly classified is obtained with the RDP and BP methods. A higher percentage was obtained with the CC method.

5.2. The SPOT satellite image classification problem

The objective of this application was to compare the level of generalization of our growing method, without any optimization, with some optimized methods (Rulex, RuleNeg), as well as with another growing method (CC).

This three-dimensional classification problem consists on classifying four classes, artificial areas (roads, cities,…), rural areas (fields,…), natural areas (forest,…), and hydrography (lakes, rivers,…) from a SPOT satellite image of the region of Mulhouse, in France. This satellite produces 60 km × 60 km images with a resolution high of 20 m. The input vector contains three spectral bands which

Table 8
Percentage of test data correctly classified for the RDP, RuleNeg, Rulex, and CC learning algorithm on the SPOT satellite image of Mulhouse, France

Problem	Algorithm	INs Hidden units	Percentage
Artificial areas (Roads, Cities, ...)	RDP	55	79.38
	RuleNeg	2	87.32
	Rulex	6	82.8
	CC	25	73.74
Rural areas (Fields, ...)	RDP	35	89.31
	RuleNeg	2	88.39
	Rulex	6	89.3
	CC	25	70.31
Natural areas (Forest, ...)	RDP	39	88.24
	RuleNeg	1	83.06
	Rulex	6	89.8
	CC	25	79.50
Hydrography (Lakes, Rivers, ...)	RDP	28	89.00
	RuleNeg	0	97.10
	Rulex	6	99.96
	CC	25	78.32

represent the green (0.50–0.59 μm), red (0.61–0.68 μm), and infrared (0.79–0.89 μm). Each training and testing data set contains 655 patterns with 355 patterns belonging to the actual class, and 300 to the remaining three classes. They were randomly selected from the entire image. A RDP was created for each class. The results are compared with the ones obtained, by J. Diederich (pers. comm.) by applying the rule extraction techniques called Rulex (Andrews and Geva, 1995), and RuleNeg (Pop and Diederich, 1994), and the CC growing method. The Rulex technique is based on a constrained error BP network. It is similar in concept to the radial basis function network, with the difference that the local functions are constructed from sigmoids and not from gaussians. The RuleNeg technique is based on the extraction of rules by stepwise negation. The following thermometer coding for the input values (input values for the three spectral bands in a SPOT satellite image range from 0 to 255) was used by J. Diederich (pers. comm.) to implement the RuleNeg networks: (a) 1–50 very low, (b) 51–80 low, (c) 81–110 lower medium, (d) 11–160 upper medium (e) 161–200 high, and (f) 201–255 very high. No particular value distribution was used to select the upper and lower bounds of each input range. Thus, some problems on the generalization can be expected with this approach.

5.2.1. Results

Table 8 shows a summary of the results obtained with the RDP, RuleNeg, Rulex, and CC methods when applied to the Mulhouse SPOT satellite image. As can be seen, the topologies, obtained by our growing method, contain a much greater amount of INs than the ones obtained with RuleNeg and Rulex methods. As mentioned previously, the aim of this application was to show that our method without any optimization, allows us to obtain results, in terms of generalization, comparable to those obtained with some optimized methods based on the BP model as well as with another growing method (CC). Both the RuleNeg and Rulex methods are optimization methods for building BP models based on knowledge extraction techniques.

For the artificial areas and hydrography classes we obtain a generalization level which is about 10% lower than Rulex and RuleNeg. The results obtained for the rural and natural areas are equivalent. This encourages us to work towards optimization methods of the RDP model with the aim of improving both the generalization level, and the topology size. For all four classes, the results obtained with the RDP method are better than those obtained with the CC method.

5.3. Comparison with other growing methods

(Bose and Garga, 1993) and (Kim and Park, 1995) illustrate their growing methods by applying them to toy examples. We use two of the examples proposed in the first article to compare their growing method and the RDP. The first example is a times-4 XOR classification problem. Their solution involves three intermediate layers with a total of eight INs. The RDP can solve the problem with one IN. The second problem is a times-4 AND with an extra point lying in the center of the other four points. As with the other example, the RDP can solve this problem with a single IN, where as the authors solution involves a three layer topology with a total of thirteen INs. We used two of the examples proposed in the second article to compare the ETL and the RDP models. The first example consists of separating a circular region, of diameter four, located at the center of a square with sides of length eight. Both the RDP and ETL methods can solve this problem with five INs. The second example is a four-bit odd-parity function. This problem is solved by the RDP with two INs as opposed to four needed by the ETL algorithm.

6. Discussion and concluding remarks

We introduced the recursive deterministic perceptron (RDP) which is a feedforward multilayer neural network generalization of the single layer perceptron topology (SLPT). This new model is capable of solving any two-class classification problem including those non-linearly separable as opposed to the single layer perceptron which can only solve linearly separable problems. We proposed several growing methods for constructing a RDP. These growing methods are based on the idea of augmenting the dimension of the input vector by using a number of INs. Intuitively, these INs have a similar function to that of the hidden units in the BP algorithm. That is to say, they augment the degrees of liberty necessary for transforming a NLS classification problem into a LS one.

The strategy for finding the INs needed to perform the NLS to LS transformation (which is based on the LS subset of maximum cardinality) has not been proven to give an optimal–minimal network topology. This strategy was shown to be NP-Complete when the dimension of the input vector is arbitrary. More research needs to be done in this area in order to develop some other search strategies.

We have proposed a new notion of linear separability for m classes. In this new solution, the m classes are LS by using a single hyperplane which is translated $m+1$ times with the same translation vector. In order to corroborate the effectiveness of this new notion of linear separability, we transform the original m class problem into the disjunction of $m!$ two-class problems.

We have also proposed a generalization of the NLS to LS transformation algorithm from 2 to m classes. This generalization is based on this new notion of linear separability. It is evident that this new model can be used to compute functions defined on a finite domain and thus, to approximate continuous functions of $\mathbb{R}^d$.

We have presented some results obtained by applying the RDP, BP, CC, RuleNeg, or Rulex neural networks to six classification problems. The percentage of test data correctly classified obtained with the five methods are comparable with each other. The ratio of the CC–RuleNeg–Rulex hidden units/RDP INs for the satellite image application varies considerably. As mentioned previously, the aim of the satellite application was to show that our method, without any optimization, allows us to obtain results, in terms of generalization, which are comparable to those obtained with other optimized methods based on the BP model. Our growing construction methods are parameterized by the choice of the selection strategy of the LS subsets in each step of construction. Thus, many different topologies can solve the same problem. The choice of a good selection strategy can allow us to optimize the topology. In any case, the general NLS to LS transformation algorithm (Table 1) allows us to obtain any topology which linearly separates the two subsets, provided that the right selection strategy of the LS subsets is given. We also present some comparison results between the RDP and two other growing methods. We used the same toy examples proposed by the authors of both methods.

The main disadvantages of RDP neural networks over other feedforward multilayer neural networks are:

- The fan-ins. The fan-ins could be a restriction for hardware implementations. Notice that in our model, the INs are connected to the previous ones only if their connection has a value different than zero. In this work, we did not try to optimize the topology in terms of number of connections. One way to do this optimization is by choosing, at each construction step, a hyperplane for linearly separating the selected subset of input patterns from the remaining patterns, containing the greatest number of zero weight components. In any case, the general NLS to LS transformation algorithm (Table 1) allows us to obtain any topology which linearly separates two subsets, provided that the right selection strategy of the LS subsets is given.
- The slightly lower generalization level (i.e. *percentage of test data correctly classified*). For any classification model, examples can always be found for which the model produces low generalization. If it is known how the model works, then it can be possible to explain the reason for this low generalization, and thus, intervene on the model parameters in order to improve this generalization. The structure of a RDP is flexible and transparent. The fact that, at each construction step of the RDP, there are several hyperplanes for linearly separating the selected subset of input patterns, from the remaining patterns, allows us to control and improve the generalization. Another solution to improve the level of generalization in a RDP is by interpolating the misclassified testing patterns with the addition of new INs to the topology of the RDP (Tajine and Elizondo, 1997). All these optimizations will allow us to control and improve the generalization of the RDP models.
- The computational time needed to find the LS subsets for building up the INs. As discussed before, we can make a compromise between size of the network topology and the computational time. In the worth case, topology size wise, we can find, with a linear approach, $n-1$ INs which can compute the RDP where n is the number of input patterns. In the worth case, computationally wise, we can find the smallest topology, with a NP-Complete approach. In all cases, the construction of the RDP, by the proposed algorithms, is guaranteed. It is not possible to compare the computational needs of the RDP neural network with other methods such as BP or CC because, contrary to the RDP, the convergence of the BP and CC methods is not guaranteed.

The main advantage of RDP neural networks over other feedforward multilayer neural networks are:

- Self determination of the number of INs needed to solve a given problem. This method adds INs to the network

one by one until the NLS problem becomes LS. Methods like BP require the user to define the number of hidden units to use to solve a problem.

- Guaranteed converge. Gradient descendent methods, such as the BP, have the disadvantage of being prone to getting stuck in local minima. Theorems 12, 13 (general NLS to LS algorithm), 14, and 15 (specializes NLS to LS algorithm), and 23 and 24 (m-class NLS to LS algorithm), respectively, guarantee the convergence of the 2 and m class RDP neural networks.
- Avoidance of catastrophic interference. Catastrophic interference refers to the complete loss of all the previously learned information. This can happen when a neural network model is trained with a subset of the total training data set, and new patterns from this data set are added. Due to the incremental learning nature of the RDP neural network, this problem will not arise with this learning method. The latter because once a new IN is added, its weights are frozen (Tajine and Elizondo, 1997).
- Using the RDP networks for rule extraction. As discussed before, the existing methods for extracting rules from neural networks have shown that this is a difficult task. This task can be simplified when using RDPs since their responses can always be explained in terms of the decision region which is defined by the hyperplanes corresponding to the INs composing the RDP. Actually, the decision region is a finite union of open polytopes as illustrated in Section 2 for the XOR problem.

Some of the problems we are studying at present include:

- Given a set S consisting of a finite union of open polytopes in $\mathbb{R}^d$, find X,Y of minimal cardinality (a sample), such that, there exists a RDP P, separating X and Y, and, S is a decision region defined by P. This result can be used for image compression, and also allows us to control the level of generalization of a RDP model.Given two finite subsets X,Y of $\mathbb{R}^d$:
- Introduce the separability class $\mathcal{CS}_Y(X)$ of X relatively to Y for a RDP, as a generalization of the notion of linear separability class for the SLPT, and characterize topologically and geometrically $\mathcal{CS}_Y(X)$ (Tajine et al., 1997). The characterization of $\mathcal{CS}_Y(X)$ of X relatively to Y for a RDP, will help to optimize the topology of the RDP.
- Find a lower bound for the height and width of the graph representing a RDP which separates X and Y by using some topological properties associated to X,Y.

Evidently, all these problems can also be explored for the general case of m classes.

Appendix A

Proof of Theorem 5:

The proof of this theorem is three-fold. Let X,Y be two finite subsets of $\mathbb{R}^d$:

1. first we prove that $X \| Y(P) \Leftrightarrow CH(X) \| CH(Y)(P)$.

   $\Leftarrow$) $X \subset CH(X)$, $Y \subset CH(Y)$ thus $CH(X) \| CH(Y)$ $(P) \Rightarrow X \| Y(P) \Rightarrow$) Assume that $X \| Y(\mathcal{P}(\mathbf{w}, t))$, then:
   $\forall \mathbf{x} \in X, \mathbf{w}^T\mathbf{x} + t > 0$ and $\forall \mathbf{y} \in Y, \mathbf{w}^T\mathbf{y} + t < 0$
   Let $\mathbf{x} \in CH(X)$, then, $\mathbf{x} = t_1\mathbf{x}_1 + \ldots + t_m\mathbf{x}_m$, where $t_1,\ldots,t_m \in [0,1]$ and $t_1 + \ldots + t_m = 1$.
   $\mathbf{w}^T\mathbf{x} + t = \mathbf{w}^T(t_1\mathbf{x}_1 + \ldots + t_m\mathbf{x}_m) + t = t_1(\mathbf{w}^T\mathbf{x}_1 + t) + \ldots + t_m(\mathbf{w}^T\mathbf{x}_m + t) > 0$,
   then $\forall \mathbf{x} \in CH(X), \mathbf{w}^T\mathbf{x} + t > 0$
   Let $\mathbf{y} \in CH(Y)$, then, $\mathbf{y} = s_1\mathbf{y}_1 + \ldots + s_n\mathbf{y}_n$, where $s_1, \ldots, s_n \in [0, 1]$ and $s_1 + \ldots + s_n = 1$.
   $\mathbf{w}^T\mathbf{y} + t = \mathbf{w}^T(s_1\mathbf{y}_1 + \ldots + s_n\mathbf{y}_n) + t = s_1(\mathbf{w}^T\mathbf{y}_1 + t) + \ldots + s_n(\mathbf{w}^T\mathbf{y}_n + t) < 0$.
   then $\forall \mathbf{y} \in CH(Y)$, $\mathbf{w}^T\mathbf{y} = T < 0$, Thus, $CH(X) \| CH(Y)(\mathcal{P}(\mathbf{w}, t))$.

2. next we prove that $-CH(X) = CH(-X)$ and $CH(X) \oplus CH(Y) = CH(X \oplus Y)$. Thus, $CH(X \ominus Y) = CH(X) \ominus CH(Y)$.

   Let $\mathbf{x} \in CH(X)$, then $\mathbf{x} = t_1\mathbf{x}_1 + \ldots + t_m\mathbf{x}_m$, where $t_1, \ldots, t_m \in [0, 1]$ and $t_1 + \ldots + t_m = 1$. Thus $-\mathbf{x} = t_1(-\mathbf{x}_1) + \ldots + t_m(-\mathbf{x}_m)$. So, $-CH(X) = CH(-X)$.
   Let $\mathbf{y} \in CH(Y)$, then $\mathbf{y} = s_1\mathbf{y}_1 + \ldots + s_n\mathbf{y}_n, s_1, \ldots, s_n \in [0, 1]$ and $s_1 + \ldots + s_n = 1$.
   Thus, $\mathbf{x} + \mathbf{y} = t_1\mathbf{x}_1 + \ldots + t_m\mathbf{x}_m + s_1\mathbf{y}_1 + \ldots + s_n\mathbf{y}_n$
   $= (s_1 + \ldots + s_n)(t_1\mathbf{x}_1 + \ldots + t_m\mathbf{x}_m) + (t_1 + \ldots + t_m)(s_1\mathbf{y}_1 + \ldots + s_n\mathbf{y}_n)$
   $= \Sigma_{1 \le i \le m}\Sigma_{1 \le j \le n} s_i t_j (x_i + \mathbf{y}_j)$
   then, $s_i t_j \ge 0$ and $\Sigma_{1 \le i \le m}\Sigma_{1 \le j \le n} s_i t_j = (\Sigma_{1 \le i \le m} s_i)(\Sigma_{1 \le j \le n} t_j) = 1$
   So, $(\mathbf{x} + \mathbf{y}) \in CH(X \oplus Y)$, then $CH(X) \oplus CH(Y) \subset CH(X \oplus Y)$.
   Let $\mathbf{z}_1, \mathbf{z}_2 \in CH(X) \oplus CH(Y)$ then $\mathbf{z}_1 = \mathbf{x} + \mathbf{y}$, $\mathbf{z}_2 = \mathbf{x}' + \mathbf{y}'$ for $\mathbf{x}, \mathbf{x}' \in X$ and $\mathbf{y}, \mathbf{y}' \in Y$
   Let $t \in [0, 1]$ then $t\mathbf{z}_1 + (1-t)\mathbf{z}_2 = (t\mathbf{x} + (1-t)\mathbf{x}') + (t\mathbf{y} + (1-t)\mathbf{y}')$.
   Thus, $t\mathbf{x} + (1-t)\mathbf{x}' \in CH(X)$ and $t\mathbf{y} + (1-t)\mathbf{y}' \in CH(Y)$
   then, the segment $[\mathbf{z}_1, \mathbf{z}_2] \subset CH(X) \oplus CH(Y)$
   Thus, $CH(X) \oplus CH(Y)$ is a convex and $CH(X) \oplus CH(Y) \subset CH(X \oplus Y)$. So, $CH(X) \oplus CH(Y) = CH(X \oplus Y)$.

3. last we prove that $CH(X) \| CH(Y) \Leftrightarrow CH(X) \cap CH(Y) = \varnothing$.

   $\Rightarrow$) if $CH(X) \| CH(Y)$ $(\mathcal{P}(\mathbf{w}, t))$, then

$\forall \mathbf{x} \in CH(X), \mathbf{w}^T\mathbf{x} + t > 0$ and $\forall \mathbf{y} \in CH(Y), \mathbf{w}^T\mathbf{y} + t < 0$
   thus, $CH(X) \cap CH(Y) = \varnothing$
   $\Leftarrow$) if $CH(X) \cap CH(Y) = \varnothing$ then $0 \notin CH(X) \ominus CH(Y) = CH(X \ominus Y)$.

Let $\mathbf{v}_1, \ldots, \mathbf{v}_k \in \mathbb{R}^d, b_1, \ldots, b_k \in \mathbb{R}$ such that $CH(X \ominus Y) = \{\mathbf{x} \in \mathbb{R}^n$ such that $\mathbf{v}_i^T\mathbf{x} \geq b_i$ for $\leq i \leq k\}$.
Then, $0 \notin CH(X \ominus Y) \Rightarrow \exists i$ such that $0 < b_i$, thus, $\forall \mathbf{z} \in CH(X \ominus Y)$, $\mathbf{v}_i^T\mathbf{z} \geq b_i > 0$, then $\forall \mathbf{x} \in CH(X)$, $\forall \mathbf{y} \in CH(Y)$, $\mathbf{v}_i^T\mathbf{x} > \mathbf{v}_i^T\mathbf{y}$.
Then, $\alpha = \mathrm{Max}_{\mathbf{y} \in Y}\mathbf{v}_i^T\mathbf{y} < \beta = \mathrm{Min}_{\mathbf{x} \in X}\mathbf{v}_i^T\mathbf{x}$.
So, $\forall \mathbf{x} \in CH(X), \mathbf{v}_i^T\mathbf{x} > \frac{\beta+\alpha}{2}$ and $\forall \mathbf{y} \in CH(Y)$, $\mathbf{v}_i^T\mathbf{y} < \frac{\beta+\alpha}{2}$
Thus, X and Y are LS by the hyperplane $\mathcal{P}(\mathbf{v}_i, -\frac{\beta+\alpha}{2})$
Thus, this theorem is a consequence of 1, 2 and 3. □

Proof of Proposition 6:

If $S = \{\mathbf{x}\}$ then $CH(\{\mathbf{x}\}) \cap CH(\varnothing) = \{\mathbf{x}\} \cap \varnothing = \varnothing$, thus $\{\mathbf{x}\} \| \varnothing$ ($\{\mathbf{x}\} \| \varnothing (\mathcal{P}(\mathbf{x}, 1))$).
Let $n \geq 2$ and assume that this proposition is true for all sets S of cardinality n. Let $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_{n+1}\}$ be a set of cardinality $n + 1$, then we have $\{\mathbf{x}_{n+1}\} \| (S \backslash \{\mathbf{x}_{n+1}\})$ or $\mathbf{x}_{n+1} \in CH(S \backslash \{\mathbf{x}_{n+1}\})$.
If $\mathbf{x}_{n+1} \in CH(S \backslash \{\mathbf{x}_{n+1}\})$, then there exists $\mathbf{y} \in (S \backslash \{\mathbf{x}_{n+1}\})$ (*assume that* $\mathbf{y} = \mathbf{x}_n$) such that $\{\mathbf{x}_n\} \| (S \backslash \{\mathbf{x}_n, \mathbf{x}_{n+1}\})$.
We will prove that $\mathbf{x}_n \notin CH(S \backslash \{\mathbf{x}_n\})$.
Assume that $\mathbf{x}_n \in CH(S \backslash \{\mathbf{x}_n\})$, then $\mathbf{x}_n = t_1\mathbf{x}_1 + \ldots + t_{n-1}\mathbf{x}_{n-1} + t_{n+1}\mathbf{x}_{n+1}$ where $t_1, \ldots, t_{n-1}, t_{n+1} \in [0, 1[$ and $t_1 + \ldots + t_{n-1} + t_{n+1} = 1$.
But $\mathbf{x}_{n+1} \in CH(S \backslash \{\mathbf{x}_{n+1}\})$, hence $\mathbf{x}_{n+1} = s_1\mathbf{x}_1 + \ldots + s_n\mathbf{x}_n$ for $s_1, \ldots, s_n \in [0, 1[$ and $s_1 + \ldots + s_n = 1$
Then $\mathbf{x}_n = \frac{1}{1 - t_{n+1}s_n}[(t_1 + t_{n+1}s_1)\mathbf{x}_1 + \ldots + (t_{n-1} + t_{n+1}s_{n-1})\mathbf{x}_{n-1}](t_{n+1}s_n \in [0, 1[)$, which is absurd because it implies that $\mathbf{x}_n \in CH(S \backslash \{\mathbf{x}_{n+1}, \mathbf{x}_n\})$. So, $\{\mathbf{x}_n\} \| (S \backslash \{\mathbf{x}_n\})$. □

Proof of Proposition 8:

- $ns(S) = \mathrm{Card}(\{\{A, S \backslash A\}; A \subset S$ *and* $A \| (S \backslash A)\})$, thus $ns(S) \leq 2^{\mathrm{Card}(S)-1}$. This proof is two-fold.

1. first we prove that if $ns(S) \leq 2^{\mathrm{Card}(S)-1}$, then S is a set of affinely independent points. Assume, by induction, that if $\mathrm{Card}(S) = n$ and $ns(S) = 2^{n-1}$, then S is a set of affinely independent points and let S' be a set of $n + 1$ points (i.e. $S' = \{\mathbf{x}_0, \ldots, \mathbf{x}_n\}$) such that $ns(S') = 2^n$, $\{\mathbf{x}_0\} \| (S' \backslash \{\mathbf{x}_0\})$ thus $\mathrm{Card}(S' \backslash \{\mathbf{x}\}) = n$ and hence $ns(S' \backslash \{\mathbf{x}_0\}) = 2^{(\mathrm{Card}(S') - 1) - 1}$ thus, $S' \backslash \{\mathbf{x}_0\}$ is a set of affinely independent points. Assume that S' is not a set of affinely independent points, then $\exists (\lambda_1, \ldots, \lambda_n) \in \mathbb{R}^d, \lambda_1 + \ldots + \lambda_n = 1$ and $\mathbf{x}_0 = \lambda_1\mathbf{x}_1 + \ldots + \lambda_n\mathbf{x}_n = \lambda_1\mathbf{x}_1 + \ldots + \lambda_q\mathbf{x}_q + \lambda_1\mathbf{x}_{q+1} + \ldots + \lambda_n\mathbf{x}_n$ where $\lambda_1, \ldots, \lambda_q \geq 0$ and $\lambda_{q+1}, \ldots, \lambda_n \leq 0$.

   Thus, $\{\mathbf{x}_0, \mathbf{x}_{q+1}, \ldots, \mathbf{x}_n\} \| \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_q\} (\mathcal{P}(\mathbf{u}, t))$ hence, $\mathbf{u}^T\mathbf{x}_0 + t, \mathbf{u}^T\mathbf{x}_{q+1} + t, \ldots, \mathbf{u}^T\mathbf{x}_n + t < 0$ and $\mathbf{u}^T\mathbf{x}_1 + t, \mathbf{u}^T\mathbf{x}_2 + t, \ldots, \mathbf{u}^T\mathbf{x}_q + t > 0$. So, $\mathbf{u}^T\mathbf{x}_0 + t = \mathbf{u}^T(\lambda_1\mathbf{x}_1 + \ldots + \lambda_q\mathbf{x}_q + \lambda_{q+1}\mathbf{x}_{q+1} + \ldots + \lambda_n\mathbf{x}_n) + t = \lambda_1\mathbf{u}^T\mathbf{x}_1 + \ldots + \lambda_q\mathbf{u}^T\mathbf{x}_q + \lambda_{q+1}\mathbf{u}^T\mathbf{x}_{q+1} + \ldots + \lambda_n\mathbf{u}^T\mathbf{x}_n + t > 0$ which is absurd, thus S' is a set of affinely independent points which implies that $\mathrm{Card}(S') \leq d + 1$.

2. last we prove that if S is a set of affinely independent points, then $ns(S) = 2^{\mathrm{Card}(S-1)}$.

   Let S be a set of affinely independent points and let $S' \subset S$, then we will prove that $S' \| (S \backslash S')$. Assume that $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ and $S' = \{\mathbf{x}_1, \ldots, \mathbf{x}_q\}$ with $q < n$ and assume that S' and $(S \backslash S')$ are NLS, $CH(S') \cap CH(S \backslash S') \neq \varnothing$. Thus, $\exists (t_1, \ldots, t_n) \in \mathbb{R}^d$ such that $t_1, \ldots, t_n \geq 0, t_1 + \ldots + t_q = t_{q+1} + \ldots + t_n = 1$ and $t_1\mathbf{x}_1 + \ldots + t_q\mathbf{x}_q = t_{q+1}\mathbf{x}_{q+1} + \ldots + t_n\mathbf{x}_n$, hence $\mathbf{x}_1 = -((t_2/t_1)\mathbf{x}_2 + \ldots + (t_q/t_1)\mathbf{x}_q) + (t_{q+1}/t_1)\mathbf{x}_{q+1} + \ldots + (t_n/t_1)\mathbf{x}_n$ (we can assume that $t_1 \neq 0$). Thus, $-((t_2/t_1) + \ldots + (t_q/t_1)) + (t_{q+1}/t_1) + \ldots + (t_n/t_1) = 1$, hence S is not an affinely independent set of points, which is absurd, so $S' \| (S \backslash S')$.

Thus, by 1 and 2 we can conclude that $ns(S) = 2^{\mathrm{Card}(S-1)}$ which implies that S is a set of affinely independent points.

- Assume that for all A such that $\mathrm{Card}(A) \leq n$ we have $ns(A) \geq \mathrm{Card}(A)$ and $ns(A) = \mathrm{Card}(A)$ implies that there exist a straight line D such that all the points of A are on D. Let $S = \{\mathbf{x}_0, \ldots, \mathbf{x}_n\}$ be a set of cardinality $n + 1$. If all the points of S are on a straight line D, the $ns(S) = n + 1$. If not all the points of S are on the same straight line, then by the Proposition 7, there exists $\mathbf{x} \in S$ such that $\{\mathbf{x}\} \| (S \backslash \{\mathbf{x}\}), (\mathbf{x} = \mathbf{x}_0)$. Two cases are possible:

1. all the points of $S \backslash \{\mathbf{x}_0\}$ are on the straight line $D = \{P_0 + t\mathbf{u} | t \in \mathbb{R}\}$ with $x_i = P_0 + t_i\mathbf{u}$ for $1 \leq i \leq n$ and $t_1 < t_2 < \ldots < t_n$, thus $\mathbf{x}_0 \in D$. So, $\{\mathbf{x}_1, \ldots, \mathbf{x}_i\} \| \{\mathbf{x}_0, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_n\}$ and $\{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{i-1}\} \| \{\mathbf{x}_i, \ldots, \mathbf{x}_n\}$ for $1 \leq i \leq n$, thus $ns(S) \geq 2n > n$ for $n \geq 2$.
2. not all the points of $S \backslash \{\mathbf{x}_0\}$ are on the same straight line, then $ns(S \backslash \{\mathbf{x}_0\}) \geq n + 1$, thus $ns(S) > n + 1$. □

Proof of Theorem 12:

Let $Z = \{z \in S_i | not(nmcc(\{z\}, S_i))\} = \{z_1, \ldots, z_k\}$ then two cases are possible:

1. $Z \neq S_i$ then there exists $j_1 < \ldots < j_k$ such that $\forall l \leq k, \forall z \in (S_i \backslash \{\mathbf{z}_l\}) \mathbf{z}_l(d + j_l) = \mathbf{z}(d + j_l)$.

   Let $S_i' = \Pi_{\{1, \ldots, d+i\} \{j_l, \ldots, j_k\}} S_i$; thus, $\forall \mathbf{x}, \mathbf{x}' \in (S_i \backslash Z_i) \forall l \leq k$, $\mathbf{x}(d + j_l) = \mathbf{x}'(d + j_l)$
   Let $\mathbf{x}' \in S_i'$ such that $\{\mathbf{x}'\} \| (S_i' \backslash \{\mathbf{x}'\}) (\mathcal{P}(\mathbf{w}, t))$, then let $\mathbf{w}' \in \mathbb{R}^{d+i}$ such that $\mathbf{w} = \Pi_{\{1, \ldots, d+i\} \backslash \{j_1, \ldots, j_k\}} \mathbf{w}'$ and for $1 \leq l \leq k$, $\mathbf{w}'(d + j_l) = a$ if $\mathbf{x}''(d + j_l) > \mathbf{z}_l(d + j_l)$ and $\mathbf{w}'(d + j_l) = -a$ if $\mathbf{x}''(d + j_l) < \mathbf{z}_l(d + j_l)$ where $a = \mathrm{Max}_{1 \leq l \leq k} (\Sigma_{h \notin \{j_1, \ldots, j_k\}} |\mathbf{w}'(h)\mathbf{z}_l(h)|)$ and $\mathbf{x}' = \Pi_{\{1, \ldots, d+i\} \backslash \{j_1, \ldots, j_k\}} \mathbf{x}''$ Let $t' = t + \Sigma_{1 \leq l \leq k} \mathbf{x}''(d + j_l)\ \mathbf{w}'(d + j_l)$.
   So, $nmcc_d(\{\mathbf{x}''\}, S_i)$ and $\{\mathbf{x}''\} \| (S_i \backslash \{\mathbf{x}''\}) (\mathcal{P}(\mathbf{w}', t'))$. □

Proof of Theorem 13:

Assume that $i < j$ and $\Pi_{\{1,\ldots,d\}}Z_i = \Pi_{\{1,\ldots,d\}}Z_j$ then, by construction, we have $not(nmcc_d(Z_j,S_j))$ because $\forall \mathbf{z}, \mathbf{z}'' \in Z_j$, $\forall \mathbf{x} \in (S_j \backslash Z_j)(\mathbf{z}(d+i+1) = \mathbf{z}(d+i+1)$ and $\mathbf{z}(d+i+1) \neq \mathbf{x}(d+i+1))$. So, if $n-1$ is the last value of i then $n-1 < 2^{\mathrm{Card}(X \cup Y)}$. Therefore, there exists $\mathbf{w}_n \in \mathbb{R}^{d+n}, t_n \in \mathbb{R}$ such that $X_n \| Y_n(\mathcal{P}(\mathbf{w}_n, t_n))$. Thus, the RDP $[(\mathbf{w}_0, t_0, a, b), \ldots, (\mathbf{w}_{n-1}, t_{n-1}, a, b), (\mathbf{w}_n, t_n, a, b)]$ linearly separates X and Y. □

Proof of Theorem 14:

We will prove that, there exists $\mathbf{x} \in X_i' \cup Y_i'$ such that $\{\mathbf{x}\} \| (S_i \backslash \{\mathbf{x}\})$.

Assume that $\forall \mathbf{x} \in X_i' \cup Y_i'$, $\{\mathbf{x}\}$ and $(S_i \backslash \{\mathbf{x}\})$ are NLS, then $X_i' \cup Y_i' \subset CH(S_i \backslash (X_i' \cup Y_i'))$. So, if $S_i = \{\mathbf{v}_1, \ldots, \mathbf{v}_k, \mathbf{v}_{k+1}, \ldots, \mathbf{v}_{k+m}, \mathbf{v}_{k+m+1}, \ldots, \mathbf{v}_{k+m+r}\}$ where $X_i' \cup Y_i' = \{\mathbf{v}_1, \ldots, \mathbf{v}_k\}, X_i \backslash X_i' = \{\mathbf{v}_{k+1}, \ldots, \mathbf{v}_{k+m}$ and $Y_i \backslash Y_i' = \{\mathbf{v}_{k+m+1}, \ldots, \mathbf{v}_{k+m+r}\}$.

Let $\mathbf{x} \in X_i' \cup Y_i'$, then $\mathbf{x} = t_1 \mathbf{v}_{k+1} + \ldots + t_{m+r}\mathbf{v}_{k+m+r} t_1, \ldots, t_{m+r} \geq 0$ and $t_1 + \ldots + t_{m+r} = 1$.

Let $j < m$ and $1 \leq l \leq i$ such that $\mathbf{v}_{k+j}(l) = a$ and $\mathbf{x}(l) = b (a < b)$. If $t_i > 0$ then $b = \mathbf{x}(l) < (t_1 + \ldots + t_{m+r})b = b$, which is absurd; thus, $\forall i \leq m$, $t_i = 0$. Let $j < r$ and $1 \leq l \leq i$ such that $\mathbf{v}_{k+m+j}(l) = b$ and $\mathbf{x}(l) = a (a < b)$. If $t_{m+i} > 0$ then $a = \mathbf{x}(l) > (t_{m+1} + \ldots + t_{m+r})a = a$, which is absurd; then, $\forall i \leq m$, $t_{m+i} = 0$, thus $\mathbf{x} \notin CH(S_i \backslash (X_i' \cup Y_i'))$.

So, there exist $\mathbf{x} \in X_i' \cup Y_i'$ such that $\{\mathbf{x}\} \| (S_i \backslash \{\mathbf{x}\})$. □

Proof of Theorem 15:

Let $n-1$ be the last value of i. Two cases are possible:

1. $X_n' \cup Y_n' \neq \emptyset$. Then there exists $\mathbf{w}_n \in \mathbb{R}^{d+n}, t_n \in \mathbb{R}$ such that $X_n \| Y_n(\mathcal{P}(\mathbf{w}_n, t_n))$.
2. $X_n' \cup Y_n' = \emptyset$. Then let $\mathbf{x} = (x_1, \ldots, x_d, \nu_1, \ldots, \nu_{n-1}) \in X_n$; in this case, the sequence $\nu_1, \ldots, \nu_{n-1}$ contains $k_X(n-1) + 1$ times a and $k_Y(n-1) - 1$ times b where $k_X(n-1)$ represents the number of LS subsets chosen in X up to step $n-1$, and $k_Y(n-1) = n - 1 - k_X(n-1)$.

   If $(y_1, \ldots, y_d, \mu_1, \ldots, \mu_{n-1}) \in Y_n$, then the sequence $\mu_1, \ldots, \mu_{n-1}$ contains $k_X(n-1) - 1$ times a and $k_Y(n-1) + 1$ times b.

   Let $\mathbf{w}_n = (0, \ldots, 0, 1, \ldots, 1) \in \mathbb{R}^{d+n}$ with d times 0 and n times 1, and let $t_n = -(k_X(n-1)a + k_Y(n-1)b)$. Thus, $\forall \mathbf{x} \in X_n, \mathbf{w}_n^{\mathrm{T}}\mathbf{x} + T_n = A - B < 0$, and $\forall \mathbf{y} \in Y_n, \mathbf{w}_n^{\mathrm{T}}\mathbf{y} + t_n = b - a > 0$. So, $X_n \| Y_n(\mathcal{P}(\mathbf{w}_n, t_n))$.

Thus, in both cases, the RDP $[(\mathbf{w}_0, t_0, a, b), \ldots, (\mathbf{w}_{n-1}, t_{n-1}, a, b), (\mathbf{w}_n, t_n, a, b)]$ linearly separates X and Y. □

Proof of Theorem 16:

- Let $\tilde{X} = Adj(X, 1), \tilde{Y} = -Adj(Y, 1)$, thus the **Max-Separability** problem for X,Y and M, is equivalent to find $\mathbf{w}' \in \mathbb{R}^{d+1}$ such that $\forall \mathbf{x} \in \tilde{X}, \mathbf{w}'^{\mathrm{T}}\mathbf{x} > 0$ and $Card(\{\mathbf{y} \in \tilde{Y} | \mathbf{w}'^{\mathrm{T}}\mathbf{y} > 0\}) \geq M$.

  Thus, the **Open Hemisphere** problem is a special case of the **Max-Separability** problem (i.e. $d = d' - 1, \tilde{X} = K, Y = \emptyset$ and $M = M'$).

  Moreover, the **Max-Separability** problem is reducible in polynomial time to the **Open Hemisphere** problem with $d' = d + 1, K = \tilde{X}_{n'} \cup \tilde{Y}$ and $M' = nm'_m$ where $\tilde{X}_{n'} = \{i\mathbf{x} | \mathbf{x} \in \tilde{X}$ and $1 \leq i \leq n'\}$. The **Open Hemisphere** problem is NP-complete (Johnson and Preparata, 1978), hence the **Max-Separability** problem is NP-complete.

- The reduction of the **Max-Separability** problem to the **Open Hemisphere** problem requires $O(nn'd)$ time. Thus, when the number d of dimensions is fixed, the use of the algorithm described in (Johnson and Preparata, 1978) solves the **Max-Separability** problem in $O((nn')^d \log(nn'))$. □

Proof of Proposition 19:

Let $\sigma \in \mathcal{S}_m$ represent a class, and let $\mathbf{w} \in \mathbb{R}^d, t \in \mathbb{R}$ such that $\forall i, \forall \mathbf{x} \in X_{\sigma(i)}, a_{i-1} < \mathbf{w}^{\mathrm{T}}\mathbf{x} + t < a_i$. Thus, $\forall i, \forall \mathbf{x} \in X_{\sigma(i)}, b_{i-1} < \frac{k}{h}\mathbf{w}^{\mathrm{T}}\mathbf{x} + (k/h)(t-a) + b < b_i$. □

Proof of Theorem 24:

We will prove that, there exists $\mathbf{x} \in X'^i_1 \cup \ldots \cup X'^i_m$ such that $\{\mathbf{x}\} \| (S_i \backslash \{\mathbf{x}\})$.

Assume that $\forall \mathbf{x} \in X'^i_1 \cup \ldots \cup X'^i_m$, $\{\mathbf{x}\}$ and $(S_i \backslash \{\mathbf{x}\})$ are NLS, then $X'^i_1 \cup \ldots \cup X'^i_m \subset CH(S_i \backslash (X'^i_1 \cup \ldots \cup X'^i_m))$.

So, if $S_i = \{\mathbf{v}_1, \ldots, \mathbf{v}_k, \mathbf{v}_{k+1}, \ldots, \mathbf{v}_{k+r_1}, \mathbf{v}_{k+r_1+1}, \ldots, \mathbf{v}_{k+r_1+r_2}, \ldots, \mathbf{v}_{k+r_1+\ldots+r_m}\}$ where $X'^i_1 \cup \ldots \cup X'^i_m = \{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$ and $X^i_j \backslash X'^i_j = \{\mathbf{v}_{k+r_1+\ldots+r_{j-1}+1}, \ldots, \mathbf{v}_{k+r_1+\ldots+r_j}\}$ for $1 \leq j \leq m$.

Let $\mathbf{x} \in X'^i_1 \cup \ldots \cup X'^i_m$, then $\mathbf{x} = t_1\mathbf{v}_{k+1} + \ldots + t_{r_1+\ldots+r_m}\mathbf{v}_{k+r_1+\ldots+r_m} t_1, \ldots, T_{r_1+\ldots+r_m} \geq 0$ and $t_1 + \ldots + t_{r_1+\ldots+r_m} = 1$.

Let $j < m$, $1 \leq l \leq r_j$ and e_l such that $\mathbf{v}_{k+r_1+\ldots+r_{j-1}+l}(e_l) = b_j$ and $\mathbf{v}_f(e_l) = b_{j+1}$ for $\mathbf{v}_f \notin X^i_j \backslash X'^i_j (b_j < b_{j+1}, \mathbf{x}(e_l) = b_{j+1})$. If $t_{r_1+\ldots+r_{j-1}+l} > 0$ then $b_{j+1} = \mathbf{x}(e_j) < (t_1 + \ldots + t_{k+r_1+\ldots+r_m})b_{j+1} = b_{j+1}$, which is absurd; thus, $\forall j \leq r_1 + \ldots + r_{m-1}, t_j = 0$.

Let $j < r_m$ and $1 \leq l \leq i$ such that $\mathbf{v}_{k+r_1+\ldots+r_{m-1}+j}(l) = b_m$ and $\mathbf{v}_f(l) = b_{m-1}$ for $\mathbf{v}_f \notin X^i_m \backslash X'^i_m (b_{m-1} < b_m, \mathbf{x}(l) = b_{m-1})$. If $t_{r_1+\ldots+r_{m-1}+j} > 0$ then $b_{m-1} = \mathbf{x}(l) > (t_{r_1+\ldots+r_{m-1}+1} + \ldots + t_{r_1+\ldots+r_{m-1}+r_m})b_{m-1} = b_{m-1}$, which is absurd; then, $\forall j \leq r_m, t_{r_1+\ldots+r_{m-1}+j} = 0$. Thus $\mathbf{x} \notin CH((S_i \backslash (X'^i_1 \cup \ldots \cup X'^i_m))$. So, there exists $\mathbf{x} \in X'^i_1 \cup \ldots \cup X'^i_m$ such that $\{\mathbf{x}\} \| (S_i \backslash \{\mathbf{x}\})$. □

Proof of Theorem 25:

Let $n-1$ be the last value of i.

Let $a_0 = b - (h/2), a_i = a_0 + ih$ for $1 \leq i \leq m$. Two cases are possible:

1. $X'^i_1 \cup \ldots \cup X'^{n-1}_m \neq \emptyset$. Then there exists $\mathbf{w}_n \in \mathbb{R}^{d+n}, t_n \in \mathbb{R}$ such that $X_1, \ldots, X_m$ are LS relatively to the arithmetic sequence $a_0, \ldots, a_m$.

2. $X'^{i}_{1} \cup \ldots \cup X'^{n-1}_{m} = \varnothing$. Then let $\mathbf{w}_n = (0, \ldots, 0, 1, \ldots, 1) \in \mathbb{R}^{d+n}$ with d times 0 and n times 1, and let $t_n = -(k_{X_1}(n-1)b_2 + \ldots + k_{X_{m-1}}(n-1)b_m + k_{X_m}(n-1)b_{m-1})$. Thus, $\forall j < m, \forall \mathbf{x} \in X_j^n$, $\mathbf{w}_n^{\mathrm{T}}\mathbf{x} + t_n = b_j - b_{j+1} = b + (j-1)h$, and $\forall \mathbf{x} \in X_m^n$, $\mathbf{w}_n^{\mathrm{T}}\mathbf{x} + t_n = b_m - b_{m-1} = b + (m-1)h$. Thus, $\forall j \leq m$, $\forall \mathbf{x} \in X_j^n, a_{j-1} < \mathbf{w}_n^{\mathrm{T}}\mathbf{x} + t_n < a_j$.

Thus, in both cases, the m-RDP $[(\mathbf{w}_0, t_0, a_0, h_0, b_{0,1}, b_{0,2}), \ldots, (\mathbf{w}_{n-1}, t_{n-1}, a_{n-1}, h_{n-1}, b_{n-1,1}, b_{n-1,2}), (\mathbf{w}_n, t_n, a, h, b_1, \ldots, b_m)]$ linearly separates $X_1, \ldots, X_m$.

References

Andrews, R., & Geva, S. (1995). Inserting and extracting knowledge from constrained error back-propagation networks. In *Proceedings of the 6th Australian Conference on Neural Networks*, NSW, Sydney, Australia.

Bose, N.K., & Garga, A.K. (1993). Neural network design using voronoi diagrams. *IEEE Transactions on Neural Networks, 4 (5)*, 778–787.

Campbell, C., & Perez Vicente, C. (1995). The target switch algorithm: a constructive learning procedure for feed-forward neural networks. *Neural Computation, 7 (6)*, 1245–1264.

Dasarathy, B.W. (1980). Nosing around the neighborhood: a new system structure and classification rule for recognition in partially exposed environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 2 (1)*, 67–71.

Elizondo, D.A. (1997). The recursive determinist perceptron (RDP) and topology reduction strategies for neural networks. PhD thesis, Université Louis Pasteur, Strasbourg, France.

Fahlman, S., & Lebiere, C. (1990). The cascade correlation architecture. In D. Touretzky (Ed.), *Advances in neural information processing systems* (Vol. 2, pp. 524–532). San Mateo, CA: Morgan Kaufmann .

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annual Eugenics, 7 (II)*, 179–188.

Gallant, S. I. (1990). Perceptron-based learning algorithms. *IEEE Transactions on Neural Networks, 1 (2)*, 179–191.

Gates, G.W. (1972). The reduced nearest neighbor rule. *IEEE Transactions on Information Theory*, 431–433.

Gorman, T. & Sejnowski, T. (1988). Analysis of hidden units in a layered network trained to classify sonar targets. *Neural Networks, 1(1)*.

Johnson, D.S., & Preparata, F.P. (1978). The densest hemisphere problem. *Theoretical Computer Science, 6*, 93–107.

Kim, J.H., & Park, S.K. (1995). The geometrical learning of binary neural networks. *IEEE Transactions on Neural Networks, 6 (1)*, 237–247.

Pop, E., & Diederich, J. (1994). Ruleneg: Rule-extraction from neural networks by step-wise negation, Technical report, Queensland University of Technology, Neurocomputing Research Center, Queensland, Australia.

Preparata, F.P., & Shamos, M. (1985). *Computational geometry. An introduction*. New York: Springer.

Rosenblatt, F. (1962). *Principles of neurodynamics*. Washington, DC: Spartan.

Rumelhart, D.E., McClelland, J.L., & The PDP Research Group (1986a). *Parallel distributed processing* (Vol. 1). Cambridge, MA: MIT Press.

Rumelhart, D.E., McClelland, J.L., & The PDP Research Group (1986b). *Parallel distributed processing* (Vol. 2). Cambridge, MA: MIT Press.

Stoer, J., & Witzgall, C. (1970). *Convexity and optimization in finite dimensions I*. Berlin: Springer.

Tajine, M., & Elizondo, D. (1996). Enhancing the perceptron neural network by using functional composition, Technical Report 96-07, Computer Science Department, Université Louis Pasteur, Strasbourg, France.

Tajine, M., & Elizondo, D. (1997). Geometrical properties of the recursive deterministic perceptron neural networks, Technical Report, Universite Louis Pastuer, 7, Rue Rene Descartes, F-67084 Strasbourg, France.

Tajine, M., Elizondo, D. Fiesler, E., & Korczak, J. (1997). The class of linear separability method. *The European Symposium on Artificial Neural Networks* (*ESNN97*), pp. 219–224: Brussels: D-facto Publications.

Thrun, S.B., Bala, J., Bloendorn, E., Bratko, I., Cestnik, B., Cheng, J., Jong, K.D., Dzeroski, S., Fahlman, S.E., Fisher, D., Hamann, R., Kaufman, K., Keller, S., Kononenko, I., Kreuziger, J., Michalski, R., Mitchell, T., Pachowicz, P., Reich, Y., Vafaie, H., de Welde, W.V., Wenzel, W., Wnek, J., & Zhang, J. (1991). The monks problems, a performance comparison of different learning algorithms, Technical Report CMU-CS-91-197, Carnegie Mellon University.