Polytope ARTMAP: Pattern Classification Without Vigilance Based on General Geometry Categories

Dinani Gomes Amorim, Manuel Fernández Delgado, and Senén Barro Ameneiro, *Associate Member, IEEE*

***Abstract*—This paper proposes polytope ARTMAP (PTAM), an adaptive resonance theory (ART) network for classification tasks which does not use the vigilance parameter. This feature is due to the geometry of categories in PTAM, which are irregular polytopes whose borders approximate the borders among the output predictions. During training, the categories expand only towards the input pattern without category overlap. The category expansion in PTAM is naturally limited by the other categories, and not by the category size, so the vigilance is not necessary. PTAM works in a fully automatic way for pattern classification tasks, without any parameter tuning, so it is easier to employ for nonexpert users than other classifiers. PTAM achieves lower error than the leading ART networks on a complete collection of benchmark data sets, except for noisy data, without any parameter optimization.**

***Index Terms*—Adaptive resonance theory (ART) neural networks, general geometry categories, polytope category representation regions (CRRs), vigilance, parameter tuning.**

I. Introduction

Adaptive resonance theory (ART) constitutes a model for the development of artificial neural networks which is inspired by the studies of Grossberg and Carpenter [1]. These networks feature online, incremental learning, with self-organized (ART1, ART2, ART3, fuzzy ART) and supervised learning (fuzzy ARTMAP (FAM) [2], Gaussian ARTMAP (GAM) [3], distributed ARTMAP (DAM) [4], ellipsoid ARTMAP (EAM) [5], FasArt [6], and FAM with relevance (FAMR) [7], among others). These models have been widely used in many application fields, including robotics [8], data mining [9], information fusion [10], data clustering [11], multichannel pattern recognition [12], image classification [13], etc. Default ARTMAP [14] compiles the basic structure of an ARTMAP network.

Some drawbacks of ART networks have been identified in the literature. The proliferation of internal categories is perhaps the most important one [15], [16]. It has been associated with the presence of noise [17] and with the inefficiency of the category geometry [3], which requires an excessive number of categories in order to cover the input space. Category proliferation has been also associated [18] with overtraining, and cross validation [19] has been suggested as a solution. DAM [4] proposes distributed learning as another solution to the category proliferation problem. Parrado-Hernández *et al.* [20] quantitatively analyzed the influence of distributed learning in category proliferation, concluding that the reduction in the number of categories created by DAM with respect to FAM depends on the geometry of the output predictions in the data set. Specifically, DAM creates less categories than FAM in data sets with nonrectangular category geometries, while the difference between them is low in data sets with rectangular geometry.

The previous analysis suggests that category geometry is an important factor in the performance and the number of categories created by an ART network. Several ART models have been proposed with nonrectangular category geometry, given by the category choice function (CCF). GAM [3] uses CCF with hyperellipsoidal symmetry. Hypersphere ARTMAP (HAM) [21] and EAM [5] use hypersherical and hyperellipsoidal internal category representation regions (CRRs) [22], respectively. Despite of using nonhyperbox CRRs, the category geometry in HAM and EAM[1] is also preestablished (spherical or ellipsoidal) by the CCF. However, if the geometric structure of the input data (e.g., the borders among the output predictions) is not similar to the CRR geometry, the network may need more categories in order to learn the data. Thus, networks with predefined CRR geometry are more suited to data sets with similar geometric properties, e.g., rectangular geometries for FAM-DAM, or ellipsoidal geometries for GAM, HAM, and EAM, and their performance may decrease otherwise.

This paper goes one step beyond, and proposes to use categories whose geometry is not preestablished, but defined by the training patterns. Our approach, called polytope ARTMAP (PTAM) [23], uses categories which are irregular polytopes. This geometry is more flexible than the predefined category geometries in order to approximate the borders among the desired output predictions in the data set. Since the CRR geometry in PTAM is not predefined, each category can cover only the regions in the input space populated by patterns with its own output prediction, so category overlap is not necessary. The vigilance parameter can be removed, so PTAM does not need any tuning parameter.

Simplex ARTMAP [24], a previous version of this work, approximates general geometry by means of nonoverlapping simplex categories, whose CCF is a sum of Gaussian functions, and it does not use the vigilance parameter. However, the Gaussian spread replaces vigilance as tuning parameter. On the

Manuscript received November 16, 2005; revised June 9, 2006 and December 5, 2006; accepted January 2, 2007. This work was supported in part by the Spanish CICyT under the Project TIC2003-09400-C04-03 and in part by the Xunta de Galicia under the Project PGIDIT04SIN206003PR.

The authors are with the Department of Electronics and Computer Science, University of Santiago de Compostela, Santiago de Compostela 15706, Spain (e-mail: dinani@usc.es; delga@dec.usc.es; senen@dec.usc.es).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNN.2007.894036

[1] In EAM, the hyperellipsoid ratio of lengths is also predefined by the nonadaptive parameter μ.

1045-9227/$25.00 © 2007 IEEE

other hand, the number of Gaussian functions required to cover the simplex volume quickly increases with the dimension of the input space, which hinders its application to high-dimensional data sets. These limitations suggested to us to replace simplex categories defined by Gaussian choice functions by polytope categories, defined by hyperplanes whose vertices are selected training patterns.

This paper is organized as follows. Section II explains the main ideas of PTAM. Its detailed description is covered by Section III. Sections IV and V compare the results obtained by PTAM with other classifiers, including the leading ART networks. Section VI discusses these results and Section VII summarizes the main conclusions and future research lines.

II. Outline of PTAM

An internal category in the ART networks is usually defined by a weight vector which represents the input patterns codified by the category. The response of each internal category to an input pattern is given by its CCF, which depends both on the input pattern and on the category weight vector, and defines the geometry of the CRR. The CCF is constant inside the CRR, although some ART variants (FasArt [6] and adaptive fuzzy classifier (AFC) [25]) feature categories with nonconstant CCF. The network creates categories that approximate the borders among the output predictions in the data set. This approximation is more accurate, and requires less categories, if the geometries of the categories and the output predictions are similar. For example, FAM and DAM probably need more categories than EAM or HAM in order to achieve equal performance on a data set with circular geometries. However, the categories in PTAM are irregular polytopes delimited by hyperplanes which compose a piecewise linear approximation to the border among the output predictions. The category vertices are weight vectors selected among previous training patterns in order to fit the border among the output predictions. Each output prediction can be associated to one or several polytopes. Irregular polytope categories are more flexible than categories with predefined geometry to fit complex borders among the output predictions, so they might achieve lower error. If these categories were more efficient to discriminate among predictions, a lower number of categories could even be created. On the other hand, the number of weight vectors is different for each category in PTAM, and it is adaptive during training, as opposite to the other ART networks, whose categories are defined by a fixed number of weight vectors: category center and deviations in GAM, category center in EAM, vector of ranges in each dimension in FAM, etc.

The calculation of the CCF for the polytope categories is complex, specially for nonconvex geometries. For mathematical simplicity, we represent each polytope as divided in several adjacent simplexes. The polytope CCF is calculated using a choice function for each simplex. The CCF is 1 if the input pattern falls inside one of its simplexes. Otherwise, the CCF is <1 and it decreases with the distance between the pattern and the polytope.

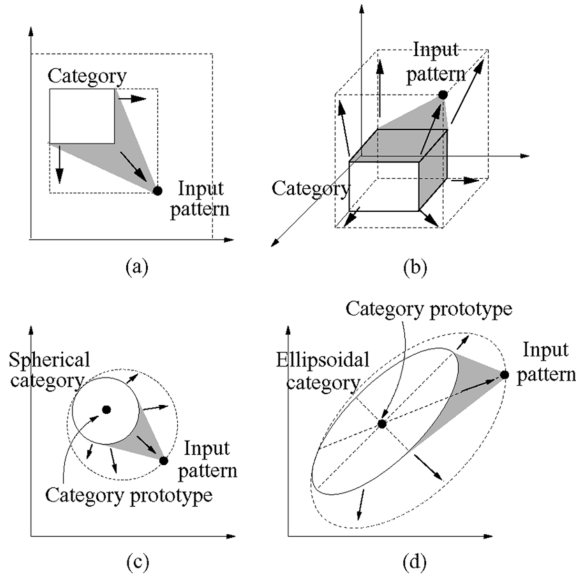


Fig. 1. Fast learning with category expansion with (a) FAM in $\mathbb{R}^2$, (b) FAM in $\mathbb{R}^3$, (c) HAM in $\mathbb{R}^2$, and (d) EAM in $\mathbb{R}^2$. The dashed line is the new category after learning. This category includes regions not in the direction (shadowed in the figure) from the old category to the input pattern.

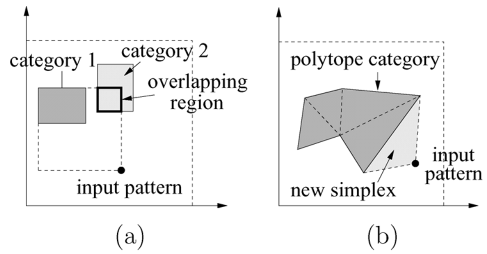


Fig. 2. (a) FAM learning in $\mathbb{R}^2$. When category 1 learns the input pattern, while keeping its hyperbox shape, it overlaps with category 2, which has a different associated prediction. (b) Polytope expansion in $\mathbb{R}^2$ creating a new simplex (triangle in $\mathbb{R}^2$).

In the ART networks, and in the "fast learning" case (the most usual), the resonant category C [with prediction $P(C)$] expands towards the training pattern $\mathbf{I}$ [with desired prediction $P(C)$]. The predefined geometry forces C not only to expand towards $\mathbf{I}$, but also in other directions (Fig. 1). Thus, C may cover not only the region between it and $\mathbf{I}$, which is supposed to be associated to prediction $P(C)$, but also other regions associated to different predictions. When a training pattern is presented to the network with desired prediction $P \neq P(C)$ and it falls in these regions, C is selected as the winner, but match tracking is invoked and another category C' with $P(C') = P$ covers that region (or a new category is created). C is not updated, so C and C' overlap. Therefore, the predefined category geometry somehow leads to category overlap [Fig. 2(a)], because categories invade regions with different predictions. The fuzzy min–max network (FMMN) [26], [27] is a remarkable exception among the ART-based networks, because it uses categories with predefined geometry (hyperbox) which do not overlap if they have different associated predictions. However, FMMN uses geometrical techniques in order to test the category overlap and, in this case, the expanded category is contracted. Therefore, when a category in FMMN tries to expand towards an input pattern, the expansion may be not possible if the hyperbox shape leads to cover regions populated by other predictions.

In the ART networks, when a test pattern falls inside an overlap region, the small categories (the most specific ones) are candidates for the resonance in the first place, due to its CCF

TABLE I
NOMENCLATURE USED IN THIS PAPER

n	Dimension of the input space
$\mathbf{I}$	Input pattern, $\mathbf{I} \in [0, 1]^n$, not complement-coded
P_d	Desired prediction for the input pattern
N_c	Number of categories
C_i	i-th Category, $i = 1, \ldots, N_c$
$P(C_i)$	Associated prediction of category C_i
N_i^s	Number of simplexes in category C_i
S_{ij}	j-th Simplex of category C_i, $j = 1, \ldots, N_i^s$
$\mathbf{w}_{ijl}$	l-th Weight vector (vertex) of simplex S_{ij} in category C_i, $l = 1, \ldots, n+1$
h_{ijk}	k-th Hyperplane of simplex S_{ij} in category C_i, $k = 1, \ldots, n+1$
$\mathbf{w}_{ijkl}$	l-th Weight vector of hyperplane h_{ijk} of simplex S_{ij} in category C_i, $l = 1, \ldots, n$
$T_i(\mathbf{I})$	Category choice function (CCF) of category C_i
$T_{ij}(\mathbf{I})$	Choice function of simplex S_{ij} in category $C_i, j = 1, \ldots, N_i^s$
r_i	Reset state of category C_i ($r_i = 1$ if C_i is reset, $r_i = 0$ otherwise)
$\mathbf{w}_i$	Weight vector of single-vector category C_i ($N_i^s = 0$)

being higher than the CCF of the bigger categories. Thus, the category size is used to select one among several overlapping categories. Also, there is a maximum category size, which limits the category expansion. Since categories can overlap, and they can cover regions with a different desired prediction, if category expansion were not limited, e.g., by this maximum size, the network might create only one category for each output prediction, covering the smallest volume which includes all the training patterns with this prediction. This fact might reduce the discrimination ability of the network, e.g., if each prediction has several nonadjacent regions, so the regions of the different predictions are mixed. Therefore, the ART networks avoid the expansion if the size of the expanded category surpasses the maximum size, defined by the vigilance parameter.[2] Unfortunately, there is no automatic way to compute the upper limit on the category size for a given data set, so the vigilance must be determined by trial and error, often using cross validation. In the case of FMMN, the overlap test (OT) limits the category expansion, but a maximum category size, given by a parameter similar to the vigilance, is also used. Both OT and maximum category size seem to have the same objective, i.e., to limit the category expansion, so they are somehow redundant. Therefore, FMMN could work without vigilance, using only the OT to limit the category expansion.

We can conclude that predefined category geometries are closely related with other typical features of ART networks, such as category overlap, category size, and vigilance. PTAM categories have no predefined geometry, although they are internally managed as a connected set of adjacent simplexes. Since the simplex covers the smallest volume defined by its vertices, the polytope category can expand only towards the input pattern—and not in other directions—by adding a new simplex with vertices in the polytope and in the input pattern [Fig. 2(b)]. The polytope categories can be built in such a way that they cover only regions in the input space populated by patterns with its equal output prediction (if there are patterns with a different prediction between the category and the input pattern, the category is adjusted as described later). If category overlap is forbidden, the polytope categories can still expand, as opposed to the other ART networks, because they cover only the regions of the input space populated by patterns with its own output prediction. Therefore, the internal categories of PTAM are not allowed to overlap. Consequently, the category size is not required to select one among several overlapping categories, so the vigilance parameter is not necessary. Thus, the expansion of a category is limited in a natural way by the other categories, and not by the category size, which is a somehow artificial criterion.

PTAM replaces the vigilance test in the ART networks by the OT (Section III-B), similarly to FMMN: If a category overlaps with other categories after the expansion towards the training pattern, the category is reset and contracted, and a new category is selected to codify the training pattern. Due to this test, a category cannot cover regions populated by input patterns with a different prediction. The test limits the category expansion without tuning parameters. Overlap among categories is tested by geometric techniques, which are described in Section III-B and in the Appendix III.

The OT itself is not enough to avoid category overlap. A category C with prediction $P(C)$ can cover a region populated by patterns with desired prediction $P_d \neq P(C)$ if no pattern with prediction P_d fell inside that region in the past; so, the categories must be able to correct previous wrong expansions. If a training pattern $\mathbf{I}$ with desired prediction P_d were presented to the network in the future, $\mathbf{I}$ would fall inside the CRR of C. In this case, C could be corrected removing the simplex which contains $\mathbf{I}$. The simplex vertices which do not belong to any other simplex would be used as new input patterns. This is done in the category adjustment step (Section III-C).

If the active category C passes the OT after the expansion towards $\mathbf{I}$, the prediction test (PT, Section III-C) determines if its desired prediction P_d is equal to $P(C)$. The training pattern $\mathbf{I}$ is assigned to the category which passes both OT and PT (Section III-D). If no category passes both tests, PTAM creates a new one (Section III-E). Since the polytope categories have at least one simplex, defined by $n+1$ vertices, n training patterns with desired prediction P_d are required to create a new single-simplex category. If the new category overlaps with other categories, or there are less than n weight vectors with prediction

[2] Although the baseline vigilance value $\bar{\rho} = 0$ is often used for classification tasks, the vigilance is raised to positive values when math tracking is invoked, so there is a maximum category size until the presentation of the following training pattern.

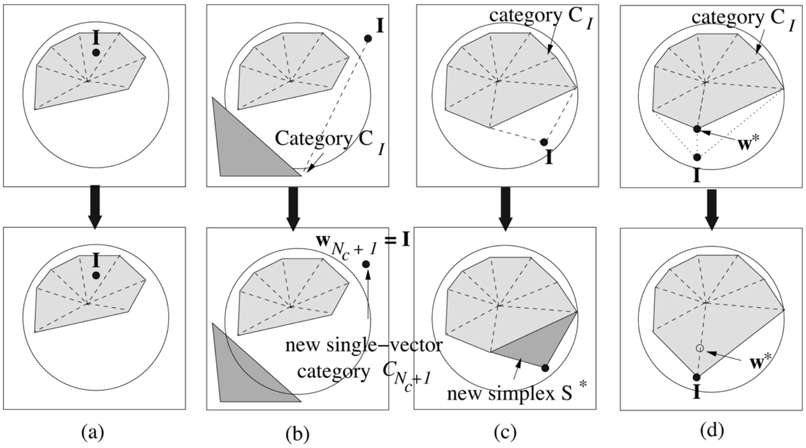


Fig. 3. Polytope category learning examples for the "circle-in-the-square" (CIS) problem in $\mathbb{R}^2$. (a) No category expansion. The input pattern falls inside C_I, which is not expanded. (b) Creation of a new single-vector category. Less than $n = 2$ weight vectors are connectable from $\mathbf{I}$ in C_I: C_I is not expanded and $\mathbf{I}$ creates a new single-vector category C_{N_c+1} with $\mathbf{w}_{N_c+1} = \mathbf{I}$. (c) Creation of a new simplex. Exactly n weight vectors are connectable from $\mathbf{I}$, and C_I expands towards $\mathbf{I}$ creating a new simplex S^*. (d) Vector replacement. The number of connectable weight vectors from $\mathbf{I}$ is $3 (> n = 2)$: vector $\mathbf{w}^*$ is replaced by $\mathbf{I}$ without volume loss for category C_I.

P_d, a new single-vector category is created with the training pattern $\mathbf{I}$.

III. PTAM Algorithm

Sections III-A–III-G describe the main steps of the training and testing phases of PTAM. Table I summarizes the nomenclature used in this paper.

A. CCF

Since each polytope category C_i is composed by a set of simplexes, its CCF $T_i(\mathbf{I})$ is defined as the maximum of the choice functions $T_{ij}(\mathbf{I})$ of its simplexes $S_{ij}, j = 1, \ldots, N_i^s$:

$$T_i(\mathbf{I}) = \max_{j=1,\ldots,N_i^s} \{T_{ij}(\mathbf{I})\}, \qquad i = 1, \ldots, N_c. \tag{1}$$

$T_{ij}(\mathbf{I})$ is defined in such a way that $T_{iJ}(\mathbf{I}) = 1$ if $\mathbf{I}$ falls inside the simplex S_{iJ}: in this case, $T_i(\mathbf{I}) = T_{iJ}(\mathbf{I}) = 1$ because $\mathbf{I}$ falls inside the CRR of C_i. Otherwise, $0 < T_{ij}(\mathbf{I}) < 1$, and it decreases with the distance $d(\mathbf{I}, S_{ij})$ between the input pattern and the simplex. Thus, $T_i(\mathbf{I}) = T_{iJ}(\mathbf{I})$, where S_{iJ} is the nearest simplex from $\mathbf{I}$ in C_i. We separately describe both cases.

1) Input Pattern Inside a Simplex: The border of simplex S_{ij} is given by $n+1$ hyperplanes $\{h_{ij1}, \ldots, h_{ij(n+1)}\}$. The hyperplane h_{ijk} is defined by n weight vectors $\{\mathbf{w}_{ijk1}, \ldots, \mathbf{w}_{ijkn}\}$ (simplex vertices), and it can be described by the following:

$$g_{ijk}(\mathbf{I}) \equiv \operatorname{sgn}(\phi_{ijk}(\widehat{\mathbf{w}}_{ijk}))\phi_{ijk}(\mathbf{I}) = 0 \tag{2}$$

where $\operatorname{sgn}(x) = 1$ if $x > 0$ and $\operatorname{sgn}(0) = 0$ and $\operatorname{sgn}(x) = -1$ if $x < 0$. The vector $\hat{\mathbf{w}}_{ijk}$ is the weight vector of simplex S_{ij} which does not belong to hyperplane h_{ijk}, and $\phi_{ijk}(\mathbf{x})$ is given by

$$\phi_{ijk}(\mathbf{x}) = \begin{vmatrix} x_1 - w_{ijk1,1} & \ldots & x_n - w_{ijk1,n} \\ w_{ijk2,1} - w_{ijk1,1} & \ldots & w_{ijk2,n} - w_{ijk1,n} \\ \ldots & & \\ w_{ijkn,1} - w_{ijk1,1} & \ldots & w_{ijkn,n} - w_{ijk1,n} \end{vmatrix}. \tag{3}$$

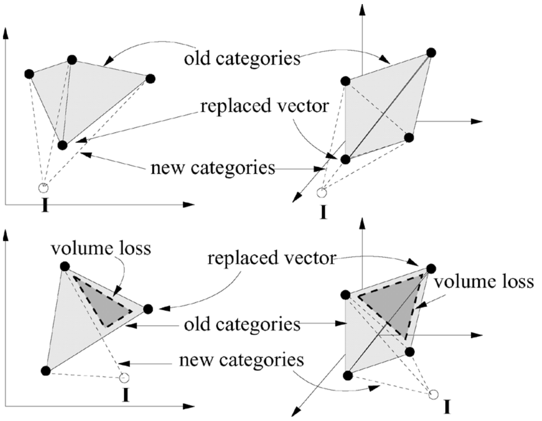
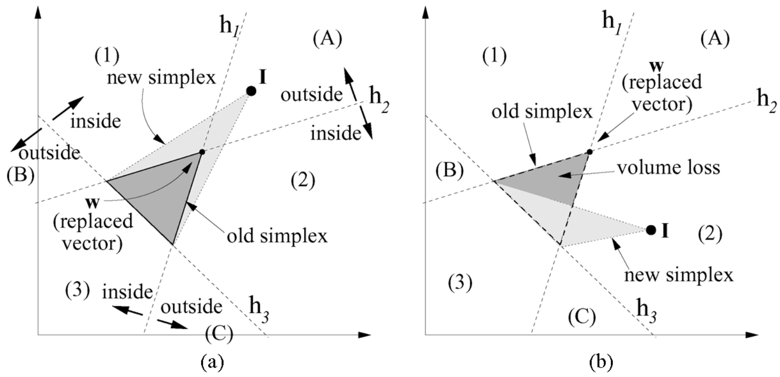


Fig. 4. Examples of weight vector replacement for simplexes in $\mathbb{R}^2$ (left panels) and $\mathbb{R}^3$ (right panels), without volume loss (upper panels) and with volume loss (lower panels) of the simplex.

In (3), $w_{ijkl,m}$ is the mth component of $\mathbf{w}_{ijkl}$ (see Appendix I). Equation (2) splits the input space $[0,1]^n$ in two regions, one for which $g_{ijk}(\mathbf{I}) > 0$ and another defined by $g_{ijk}(\mathbf{I}) < 0$. The factor $\operatorname{sgn}(\phi_{ijk}(\hat{\mathbf{w}}_{ijk}))$ in (2) guarantees that $g_{ijk}(\mathbf{I}) > 0$ in the hyperplane side that is inside the simplex, and $g_{ijk}(\mathbf{I}) < 0$ outside the simplex. An input pattern $\mathbf{I}$ falls inside the simplex S_{ij} only if it falls in the inner side of the $n+1$ hyperplanes $h_{ij1}, \ldots, h_{ij(n+1)}$ of S_{ij}. This means that $T_{ij}(\mathbf{I}) = 1$ if and only if $g_{ijk}(\mathbf{I}) > 0, \quad \forall k = 1, \ldots, n+1$.

2) Input Pattern Outside a Simplex: The choice function $T_{ij}(\mathbf{I})$ of simplex S_{ij} must satisfy $0 < T_{ij}(\mathbf{I}) < 1$ outside the simplex and it must be decreasing with the distance between the pattern and the simplex. We define $T_{ij}(\mathbf{I}) = e^{-d(\mathbf{I},S_{ij})/\gamma}$, where $\gamma \equiv \sqrt{n}/10$, so $e^{-10} < T_{ij}(\mathbf{I}) < 1$ (note that $e^{-10} \simeq 0$ and $0 \le d(\mathbf{I}, S_{ij}) < \sqrt{n}$, being $\sqrt{n}$ the diagonal length of hyperbox $[0,1]^n$). We use an approximated value of $d(\mathbf{I}, S_{ij})$ since its accurate calculation is computationally demanding for $n > 2$. Let us consider the hypersphere centered in the centroid $\mathbf{c}_{ij}$ of the simplex vertices $\mathbf{c}_{ij} = \sum_{l=1}^{n+1} \mathbf{w}_{ijl}/(n+1)$, whose radius R_{ij} is the maximum distance between this centroid and the simplex



(a) (b)

Fig. 5. Example of vector replacement condition in $\mathbb{R}^2$. (a) Patterns in regions A–C can replace weight vectors in the old simplex, but (b) patterns in regions 1–3 cannot do it, because the replacement reduces the simplex volume.

vertices $R_{ij} = \max\{\|\mathbf{w}_{ijl} - \mathbf{c}_{ij}\|\}_{l=1}^{n+1}$. We consider the following two cases.

1) If the input pattern falls inside this hypersphere ($\|\mathbf{I} - \mathbf{c}_{ij}\| < R_{ij}$), $d(\mathbf{I}, S_{ij})$ is approximated by the distance between $\mathbf{I}$ and its nearest hyperplane h_{ijk} in the simplex ($\mathbf{I}$ falls outside S_{ij}, so $g_{ijk}(\mathbf{I}) < 0$)

$$d(\mathbf{I}, S_{ij}) \simeq \min_{k \in \mathcal{S}} \frac{|g_{ijk}(\mathbf{I})|}{\|\mathbf{w}^*_{ijk}\|}, \qquad \|\mathbf{I} - \mathbf{c}_{ij}\| < R_{ij} \tag{4}$$

where $\mathbf{w}^*_{ijk}$ is the direction vector of hyperplane h_{ijk} (see Appendix II), and $|g_{ijk}(\mathbf{I})|/\|\mathbf{w}^*_{ijk}\|$ is the distance between $\mathbf{I}$ and h_{ijk}. The set $\mathcal{S}$ includes the indices k of hyperplanes h_{ijk} in S_{ij} for which $g_{ijk}(\mathbf{I}) < 0$.

2) If $\mathbf{I}$ falls outside the hypersphere, then the distance between the pattern and the simplex is approximated by the minimum distance between $\mathbf{I}$ and a vertex of S_{ij}

$$d(\mathbf{I}, S_{ij}) \simeq \min_{l=1,\ldots,n+1} \{\|\mathbf{I} - \mathbf{w}_{ijl}\|\}, \qquad \|\mathbf{I} - \mathbf{c}_{ij}\| \geq R_{ij}. \tag{5}$$

Finally, the activation function $T_{ij}(\mathbf{I})$ of simplex S_{ij} is given by

$$T_{ij}(\mathbf{I}) = \begin{cases} 1, & g_{ijk}(\mathbf{I}) > 0, \qquad k = 1, \ldots, n+1 \\ e^{-d(\mathbf{I}, S_{ij})/\gamma}, & \text{otherwise} \end{cases} \tag{6}$$

B. OT

The competition among polytope categories in PTAM selects the category C_I with the highest CCF, $T_I(\mathbf{I}) = \max\{T_i(\mathbf{I})\}_{i=1}^{N_c}$, which is not reset ($r_I = 0$). If all the categories are reset ($r_i = 1, i = 1, \ldots, N_c$), a new category is created (Section III-E). Otherwise, the OT determines if C_I overlaps with the other categories when it expands towards $\mathbf{I}$. If there is an overlap, C_I does not pass the OT and it is reset ($r_I = 1$). In this case, the nonreset category with the highest CCF is selected by the competitive process. If C_I passes the OT, the associated prediction $P(C_I)$ for C_I is tested in Section III-C.

There are several possible situations in the OT, depending on the relative position of $\mathbf{I}$ and C_I, which are illustrated in Fig. 3.

1) $\mathbf{I}$ falls inside the CRR of C_I ($T_I(\mathbf{I}) = 1$). In this case, C_I already covers $\mathbf{I}$, so it does not expand and the overlap with other categories is not possible. Thus, C_I passes the OT [Fig. 3(a)].

2) $\mathbf{I}$ falls outside the CRR of C_I ($T_I(\mathbf{I}) < 1$). Then, the category C_I can learn $\mathbf{I}$ expanding towards $\mathbf{I}$, either including $\mathbf{I}$ as a weight vector or replacing a weight vector of C_I by $\mathbf{I}$. In both cases, new segments between $\mathbf{I}$ and vertices of C_I must be created, which cannot overlap with existing categories. Therefore, the set $\mathcal{A}_I(\mathbf{I})$ of weight vectors in category C_I which are connectable from $\mathbf{I}$ without the overlap must be created. Function $O_{\text{ls}}(\mathbf{I}, \mathbf{w}, S_{ij})$ (line-simplex overlap), defined in (23) (Appendix III-A), determines if the line segment $\overrightarrow{\mathbf{Iw}}$ overlaps with simplex S_{ij} ($O_{\text{ls}}(\mathbf{I}, \mathbf{w}, S_{ij}) = 1$) or if it does not overlap ($O_{\text{ls}}(\mathbf{I}, \mathbf{w}, S_{ij}) = 0$). The set $\mathcal{A}_I(\mathbf{I})$ can be defined as $\mathcal{A}_I(\mathbf{I}) = \{\mathbf{w} \in C_I : O_{\text{ls}}(\mathbf{I}, \mathbf{w}, S_{ij}) = 0, \forall S_{ij}\}$. Depending on $|\mathcal{A}_I(\mathbf{I})|$ (cardinality of set $\mathcal{A}_I(\mathbf{I})$), there are the following three cases.

   a) If $0 \leq |\mathcal{A}_I(\mathbf{I})| < n$ [Fig. 3(b)], there are not enough connectable vertices from $\mathbf{I}$ in C_I to create a new simplex between them, because segments from $\mathbf{I}$ to C_I overlap with other categories. The category C_I does not pass the OT because it cannot expand towards $\mathbf{I}$ without the overlap.

   b) If $|\mathcal{A}_I(\mathbf{I})| = n$ [Fig. 3(c)], a new simplex S^* belonging to C_I can be created using $\mathbf{I}$ and the n weight vectors of $\mathcal{A}_I(\mathbf{I})$ (Section III-B1). If S^* overlaps with other categories, C_I is removed and it does not pass the OT. Otherwise, C_I passes the OT.

   c) If $|\mathcal{A}_I(\mathbf{I})| > n$ [Fig. 3(d)], some weight vector in $\mathcal{A}_I$ may be replaced by $\mathbf{I}$ without volume loss for the category (Section III-B2). If the modified simplexes after vector replacement do not overlap with other categories, C_I passes the OT.

Sections III-B1 and III-B2 give more details about cases in Fig. 3(c) and (d).

1) Category Expansion: Creation of a New Simplex: When $|\mathcal{A}_I(\mathbf{I})| = n$, the OT determines if category C_I can expand towards $\mathbf{I}$ creating a new simplex $S^* = \mathcal{A}_I \cup \{\mathbf{I}\}$ among them. Overlap among S^* and the other categories is tested by using the function $O_{\text{sc}}(S^*, C_i)$—overlap between simplex and category—defined in (25) (Appendix III-B) which is $O_{\text{sc}}(S^*, C_i) = 1$ if simplex S^* and category C_i overlap and $O_{\text{sc}}(S^*, C_i) = 0$, otherwise. If the former case holds for some C_i, S^* is removed and C_I does not pass the OT. Otherwise, C_I passes the OT. The category expansion increases the amount of information stored by the network—number of weight vectors and simplexes—similarly to the creation of new categories in classical ART networks.

The new simplex could be very acute if its hyperplanes are nearly parallel. In this case, the simplex volume is rather small, and even when it covers regions populated by patterns with the right prediction, it does not significantly increase the category volume. Thus, categories with acute simplexes are not efficient to cover the input space. On the other hand, if the acute simplex covers a region with a different prediction, its small volume makes it very improbable that future training patterns with that prediction fall inside this simplex, so the category adjustment step (Section III-C) cannot remove it. Therefore, acute simplexes make the expansion of other categories difficult and contribute to category proliferation. In these cases, PTAM does not create acute simplexes, and the winner category is not expanded.

The volume of a simplex in $\mathbb{R}^n$ can be computed by using the Cayley–Menger determinant [28], but its calculation is very complex. PTAM avoids the creation of acute simplexes assuming that they have some small angle between their hyperplanes. Specifically, PTAM creates a new simplex S_{ij} with weight vectors $\{\mathbf{w}_{ij1}, \dots, \mathbf{w}_{ij(n+1)}\}$ only if the angle θ_{klm} between vectors $\mathbf{w}_{ijl} - \mathbf{w}_{ijk}$ and $\mathbf{w}_{ijm} - \mathbf{w}_{ijk}$ is above a threshold value $\theta_{\min}$. The angle θ_{klm} is calculated using the inner product of vectors $\mathbf{w}_{ijl} - \mathbf{w}_{ijk}$ and $\mathbf{w}_{ijm} - \mathbf{w}_{ijk}$

$$\theta_{klm} = \cos^{-1}\left[\frac{(\mathbf{w}_{ijl} - \mathbf{w}_{ijk})(\mathbf{w}_{ijm} - \mathbf{w}_{ijk})}{\|\mathbf{w}_{ijl} - \mathbf{w}_{ijk}\|\|\mathbf{w}_{ijm} - \mathbf{w}_{ijk}\|}\right], \quad k, l, m = 1, \dots, n+1; \qquad k \neq l \neq m. \tag{7}$$

The value of $\theta_{\min}$ must be low in order to remove only the acute simplexes. From graphical considerations about acute simplexes in $\mathbb{R}^2$, we assumed that $\theta_{\min} = 10$ degree is a reasonable threshold value.

2) Category Expansion: Weight Vector Replacement: If the number of connectable weight vectors in category C_I from $\mathbf{I}$ is higher than n ($|\mathcal{A}_I(\mathbf{I})| > n$), the OT determines if some weight vector in C_I can be replaced by $\mathbf{I}$ (Fig. 4). The category C_I does not pass OT if it overlaps with other categories after the vector replacement. The replacement of weight vectors does not increase the amount of information learnt by the network—number of weight vectors and simplexes.

Fig. 4 shows that, depending on the relative position of a pattern and a simplex, the replacement of a weight vector may reduce (lower panels) or not reduce (upper panels) the volume of one or several simplexes. Fig. 5 shows in $\mathbb{R}^2$ that a simplex loses volume when the input pattern falls in regions 1–3 [Fig. 5(b)], but not when it falls in regions A–C [Fig. 5(a)]. If $\mathbf{I}$ falls in the outer (out of the simplex) side of the n hyperplanes which cross at $\mathbf{w}$, $\mathbf{I}$ can replace $\mathbf{w}$ without simplex volume loss. Otherwise, the simplexes defined by $\mathbf{w}$ lose volume [Fig. 5(b): $\mathbf{I}$ falls in the inner side of h_2 and h_3]. Let $H_{Ij}(\mathbf{w}) = \{k = 1, \dots, n+1 : \mathbf{w} \in h_{Ijk}\}$ be the set of hyperplanes in S_{Ij} which contain the vector $\mathbf{w}$. $\mathbf{I}$ can replace $\mathbf{w}$ in S_{Ij} if

$$\prod_{k \in H_{Ij}(\mathbf{w})} \Phi(g_{Ijk}(\mathbf{I})) > 0. \tag{8}$$

TABLE II
LIST OF DATA SETS USED IN THE EXPERIMENTAL WORK
(SEE SECTION IV FOR DETAILS)

No.	Name	Predictions	Observations
Two-dimensional data sets			
1	CIS	2	#patterns/prediction $\propto$ area
2	Chess	2	50% patterns/prediction
3	T3	5	20% patterns/prediction
4	T4	5	20% patterns/prediction
5	T5	6	#patterns/prediction $\propto$ area
6	T6	3	#patterns/prediction $\propto$ area
7	T7	3	50%-30%-20% patterns
Two-dimensional data sets with noise			
8	CIS-noise	2	noise level 0.01:0.01:0.05
9	T5-noise	6	noise level 0.01:0.01:0.03
Two-dimensional data sets with category overlap			
10	4G1	4	$d = 0.134$ (low overlap)
11	4G2	4	$d = 0.120$ (middle overlap)
12	4G3	4	$d = 0.084$ (high overlap)
2D data set with irregular geometry			
13	Form	2	50%-50% patterns
Real multi-dimensional data sets			
14	PID	2	768 patterns, 8 inputs
15	Abalone	3	4177 patterns, 8 inputs

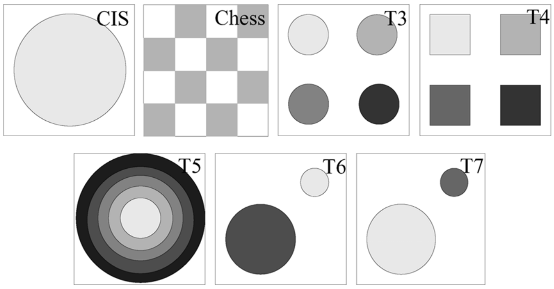


Fig. 6. The 2-D data sets CIS, chess, T3, T4, T5, T6, and T7 used in the experimental work. In T6, the number of patterns of each category is proportional to its size (75.8% outside the circles, 19.7% in the big circle, and 4.5% in the small circle). In T7, the populations are 50%-30%-20% [20].

The function $g_{ijk}(\mathbf{I})$ is defined in (2), and the function $\Phi(x)$ is defined by $\Phi(x) = 1$ if $x \leq 0$ and $\Phi(x) = 0$, otherwise. If $\mathbf{I}$ falls in the inner side of some hyperplane $h_{Ijk} \in H_{Ij}(\mathbf{w})$, then $g_{Ijk}(\mathbf{I}) > 0$ and $\Phi(g_{Ijk}(\mathbf{I})) = 0$, so (8) is not fulfilled. If $g_{Ijk}(\mathbf{I}) < 0, \forall k \in H_{Ij}(\mathbf{w})$, then $\mathbf{I}$ verifies (8). The candidate for replacement is the nearest weight vector $\mathbf{w}^*$ from $\mathbf{I}$ in $\mathcal{A}_I(\mathbf{I})$ that satisfies (8) for each simplex $S_{Ij} \in C_I$ containing $\mathbf{w}^*$. This vector $\mathbf{w}^*$ is replaced by $\mathbf{I}$ only if these simplexes do not overlap with the other categories after the vector replacement. This condition means that $O_{\text{sc}}(S_{Ij} \setminus \{\mathbf{w}^*\} \cup \{\mathbf{I}\}, C_i) = 0$ [function O_{sc} is defined in (25) in Appendix III-B], $\forall S_{Ij} : \mathbf{w}^* \in S_{Ij}$, $\forall C_i, i \neq I$. If no weight vector in $\mathcal{A}_I(\mathbf{I})$ satisfies this condition, C_I does not pass the OT.

C. PT and Category Adjustment

If C_I passes the OT, the PT compares its associated prediction $P(C_I)$ with the desired prediction P_d for the training pattern $\mathbf{I}$. If $P(C_I) = P_d$, C_I passes the PT and the resonance step in Section III-D is executed. Otherwise, $P(C_I) \neq P_d$, C_I does not pass the PT, and it is reset ($r_I = 1$). If $T_I(\mathbf{I}) = 1$, the input pattern falls inside the CRR of C_I, so the region covered by C_I is populated by patterns with $P_d \neq P(C_I)$. In this case, the simplex S_{IJ} which verifies $T_{IJ}(\mathbf{I}) = 1$ is removed in order to adjust

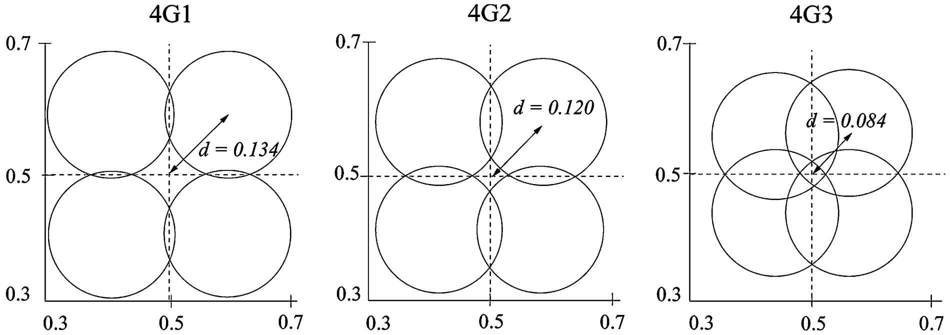


Fig. 7. Circles surrounding the Gaussian distributions, with radius 2σ ($\sigma = 0.05$ is the Gaussian spread) for the three overlap levels. The distance d between the center of each circle and the unit square center (0.5, 0.5) is $d = 0.134$ in 4G1, $d = 0.120$ in 4G2, and $d = 0.084$ in 4G3.

category C_I (category adjustment). The competitive process selects another nonreset category and the OT is performed again (Section III-B).

If $\mathbf{w}_{IJl}$ $(1 \leq l \leq n+1)$ is a weight vector of simplex S_{IJ} which does not belong to any other simplex in C_I, it is presented as a new training pattern, after the current one $\mathbf{I}$, in order to be assigned to other categories. If another simplex S_{pq} covers $\mathbf{w}_{IJl}$ during the classification of $\mathbf{I}$, S_{pq} may be removed when $\mathbf{w}_{IJl}$ is presented again, and this fact could lead to an infinite loop. In order to avoid these loops, PTAM creates a new single-vector category C^* with $\mathbf{w}_{IJl}$ when simplex S_{IJ} is removed. Thus, the OT ensures that no simplex S_{pq} covers C^*. When $\mathbf{w}_{IJl}$ is presented again as a new input pattern, C^* is removed, so that $\mathbf{w}_{IJl}$ cannot remove any simplex in the second presentation and infinite loops are not possible.

D. Resonance

If C_I passes both the PT and OT, then it has the right prediction and it expanded towards $\mathbf{I}$ without overlap, either replacing a new vector or creating a new simplex. However, in order to avoid simplex proliferation, the new simplex is only created if C_I is not the first winner in the competition, i.e., if $T_I(\mathbf{I}) \neq \max\{T_i(\mathbf{I})\}_{i=1}^{N_c}$. Otherwise, the expansion of C_I is not necessary, because if the training pattern were presented again, it would resonate with C_I (which is the first winner). If all the training patterns with the same desired prediction were successively presented, PTAM would only create one single-simplex category with that prediction, which is undesirable. However, the training patterns are usually shuffled, so patterns with different predictions are mixed and this situation is not possible.

E. Creation of a New Category

If no category passes both the OT and PT, PTAM selects the n-nearest connectable weight vectors from $\mathbf{I}$ belonging to categories with prediction P_d. The function O_{ls} [(23) in Appendix III-A] determines if a weight vector is connectable from $\mathbf{I}$. If there are n or more connectable weight vectors from $\mathbf{I}$ with prediction P_d, PTAM creates a new simplex S^* with $\mathbf{I}$ and its n-nearest connectable vectors. If S^* is not acute (Section III-B1) and it does not overlap with other categories, PTAM creates a new category C_{N_c+1} with only one simplex (S^*) and associated prediction P_d ($N_{N_c+1}^s = 1$, $S_{(N_c+1)1} = S^*$, and $P(C_{N_c+1}) = P_d$). If S^* is acute, or if it overlaps with other simplexes, then PTAM removes S^* and creates a new single-vector category C_{N_c+1} with zero simplexes ($N_{N_c+1}^s = 0$), weight vector $\mathbf{w}_{N_c+1} = \mathbf{I}$, and prediction $P(C_{N_c+1}) = P_d$. If there are less than n connectable weight vectors with prediction P_d, $\mathbf{I}$ is also added as a new single-vector category.

PTAM needs at least $n+1$ training patterns with the same prediction to create a polytope category. Otherwise, PTAM creates one single-vector category for each training pattern. During the testing phase, the CCF of each single-vector category C_i ($N_i^s = 0$) depends on the Euclidean distance between its weight vector $\mathbf{w}_i$ and $\mathbf{I}$ [see (9)]. Thus, $\mathbf{I}$ is assigned to its nearest single-vector category, i.e., PTAM behaves as a nearest neighbor classifier. In this case, the number of available training patterns is not enough to correctly learn the data set, which is severely affected by the "curse of dimensionality" problem [29].

F. PTAM Training Phase

The training phase of PTAM can be summarized in the following steps.

1) *Presentation of a new input pattern* $\mathbf{I}$.
2) *Calculation of CCFs*: Calculate $T_i(\mathbf{I}), i = 1, \ldots, N_c$ [see (1) and (6)] for all the categories.
3) *Competition*: Select the category C_I with $r_I = 0$ and the highest CCF: $T_I(\mathbf{I}) = \max\{T_i(\mathbf{I})\}_{i=1}^{N_c}$. If all the categories are reset, then go to step 8).
4) *OT*: Expand C_I towards $\mathbf{I}$ either creating a new simplex S^*, only if S^* is not an acute simplex (Section III-B1), or replacing some weight vector $\mathbf{w}^*$ in C_I by $\mathbf{I}$. If C_I overlaps with other categories after expansion, reset C_I $(r_I = 1)$, return it to its previous state, and go to step 3).
5) *PT*: If $P(C_I) \neq P_d$ (the associated prediction of C_I is different from the desired prediction), reset C_I $(r_I = 1)$. Otherwise, go to step 7).
6) *Category adjustment*: If $T_I(\mathbf{I}) = 1$ ($\mathbf{I}$ falls inside the CRR of category C_I), then remove the simplex S_{IJ} for which $T_{IJ}(\mathbf{I}) = 1$. Include in the set of input patterns the vectors $\mathbf{w}_{IJ1}, \ldots, \mathbf{w}_{IJ(n+1)}$ of S_{IJ} which do not belong to any other simplex in C_I. Go to step 3).
7) *Resonance*: Assign the input pattern to C_I. If C_I expands towards $\mathbf{I}$ creating a new simplex S^*, and $T_I(\mathbf{I}) = \max\{T_i(\mathbf{I})\}_{i=1}^{N_c}$ (C_I is the first winner), remove S^*. Go to step 1).

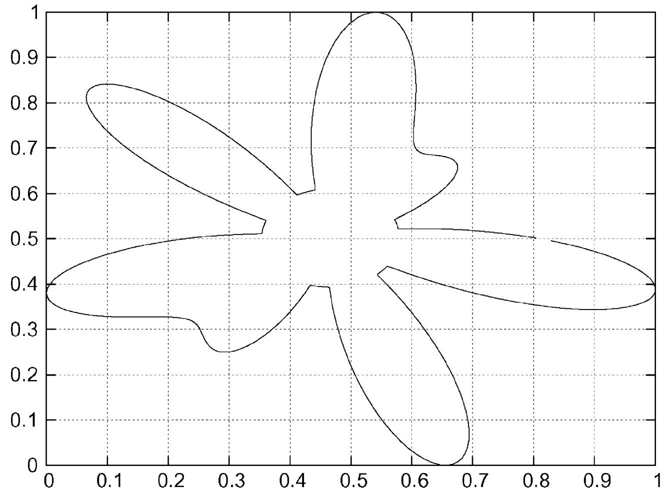


Fig. 8. Border among the two predictions in data set form.

8) *Creation of a new category*: Create a new simplex S^* with $\mathbf{I}$ and its n-nearest connectable weight vectors with prediction P_d. If S^* is not acute and it does not overlap with other categories, create a new single-simplex category C_{N_c+1} with S^* as its only simplex. If there are less than n connectable weight vectors from $\mathbf{I}$, or if S^* is an acute simplex, or if S^* overlaps with other categories, remove S^* and create a new single-vector category C_{N_c+1} with $\mathbf{w}_{N_c+1} = \mathbf{I}$. Go to step 1).

G. PTAM Testing Phase

The output of PTAM for each input pattern during the testing phase is the associated prediction of the category C_I with the highest $T_i(\mathbf{I})$ [see (1) and (6)]. The CCF of a single-vector category C_i ($N_i^s = 0$) is calculated using the Euclidean distance between its weight vector $\mathbf{w}_i$ and $\mathbf{I}$

$$T_i(\mathbf{I}) = \begin{cases} \max_{j=1,\ldots,N_i^s}\{T_{ij}(\mathbf{I})\}, & N_i^s > 0 \\ e^{-\|\mathbf{I}-\mathbf{w}_i\|/\gamma}, & N_i^s = 0 \end{cases} \qquad (9)$$

where $\gamma = \sqrt{n}/10$ (Section III-A2). The competition selects the category C_I with the highest CCF

$$T_I(\mathbf{I}) = \max_{i=1,\ldots,N_c}\{T_i(\mathbf{I})\}. \qquad (10)$$

The output of PTAM is the prediction $P(C_I)$ associated to the winner category C_I.

IV. Experimental Setting

PTAM is tested and compared with other ART and non-ART classifiers on a series of benchmark data sets (Table II). Several data sets are 2-D, in order to show graphically the polytope categories created by PTAM. Data sets with noise and category overlap have been also used in order to test the behavior of PTAM in these situations. As well, PTAM was tested with real, high-dimensional data sets.

TABLE III
Parameter Values of the Different Algorithms

FAM	$\alpha = 0.01, \beta = 1$ (learning rate), $\rho_{ab} = 1$, Weber Law and Choice-By-Difference
GAM	$\gamma = 0.2$, $\gamma = 0.8$ (PID and Abalone)
DAM	$\alpha = 0.01, \beta = 1$ (learning rate), $p = 10$ (contrast enhancement exponent)
EAM	$\mu = 0.5, D = \sqrt{n}/\mu, \beta = 1, \alpha = 0.1$
FasArt	$\bar{\rho}_b = 1, \beta = 1, \gamma_a = \gamma_b = 10$, $\gamma_a = \gamma_b = 1$ (PID and Abalone)
Common	$\bar{\rho} = 0.00 : 0.02 : 0.98$; 1-5 training epochs
SVM	$C = 0.01, 0.1, 1, 100, 1000$, Gaussian kernel with spread $\in \{0.1 : 0.1 : 1, 2.5, 5, 7.5, 10, 12, 15, 17, 20\}$

TABLE IV
Selected Values of the Tuning Parameters (ρ in the ART Networks, C and Spread (σ) in the SVM) on the Different Validation Sets. CIS-n and T5-n Stand for CIS-Noise and T5-Noise, Respectively

	FAM	GAM	DAM	EAM	FasArt	SVM	
Data set	$\bar{\rho}$	ρ	$\bar{\rho}$	ρ	ρ	C	σ
CIS	0.88	0.04	0.9	0.84	0.68	100	0.1
Chess	0.46	0.06	0.7	0.94	0.92	1000	0.1
T3	0.2	0.10	0.6	0.94	0.9	1000	0.1
T4	0.3	0.12	0.76	0.94	0.82	1000	0.1
T5	0.92	0.02	0.94	0.96	0.94	100	0.1
T6	0.54	0.04	0.88	0.92	0.84	1000	0.1
T7	0.48	0.1	0.78	0.82	0.86	1000	0.1
CIS-n 0.01	0.9	0.1	0.9	0.94	0.8	1000	0.9
0.02	0.9	0.06	0.9	0.92	0.88	100	0.9
0.03	0.92	0.02	0.9	0.9	0.9	10	0.7
0.04	0.9	0.02	0.9	0.9	0.92	10	0.9
0.05	0.92	0.02	0.9	0.9	0.92	1	0.5
T5-n 0.01	0.9	0.36	0.95	0.96	0.88	1000	0.7
0.02	0.88	0.02	0.95	0.02	0.88	10	0.4
0.03	0.92	0.02	0.95	0.02	0.32	10	0.2
Form	0.36	0.12	0.88	–	–	–	–
	0.92	–	0.94	–	–	–	–
4G1	0.9	0.02	0.75	0.96	0.54	10	0.9
4G2	0.94	0.02	0.65	0.8	0.3	10	0.9
4G3	0.94	0.02	0.1	0.9	0.02	1	0.7
PID	0.8	0.2	0.6	0.6	0.6	1000	10
Abalone	0.95	0.2	0.8	0.4	0.0	1000	2.5

A. The 2-D Data Sets

The CIS, chess (also known as "generalized XOR"), and T3–T7 (sets 1–7 in Table II) are 2-D artificial data sets (Fig. 6), with rectangular (chess and T4) and circular (CIS, T3, and T5-T7) geometries. The CIS [30] data set was often used in the ART literature. The results achieved with CIS are averaged over seven circle sizes ranging from 10% to 70% of the unit square area. Data sets T3-T7 were used in [20] as benchmarks for FasArt, FAM, and DAM. Chess is another benchmark data set frequently used in the literature [25], [31]. The number of patterns for each prediction is proportional to its area for data sets CIS, T5, and T6 (Table II), while predictions in the data sets chess, T3, and T4 have the same number of patterns.

B. Noisy 2-D Data Sets

Data sets CIS-noise and T5-noise (sets 8 and 9 in Table II) are generated by adding Gaussian noise to the CIS and T5 data sets, respectively. Several noise levels were used, given by the standard deviation of the Gaussian distribution, with values in

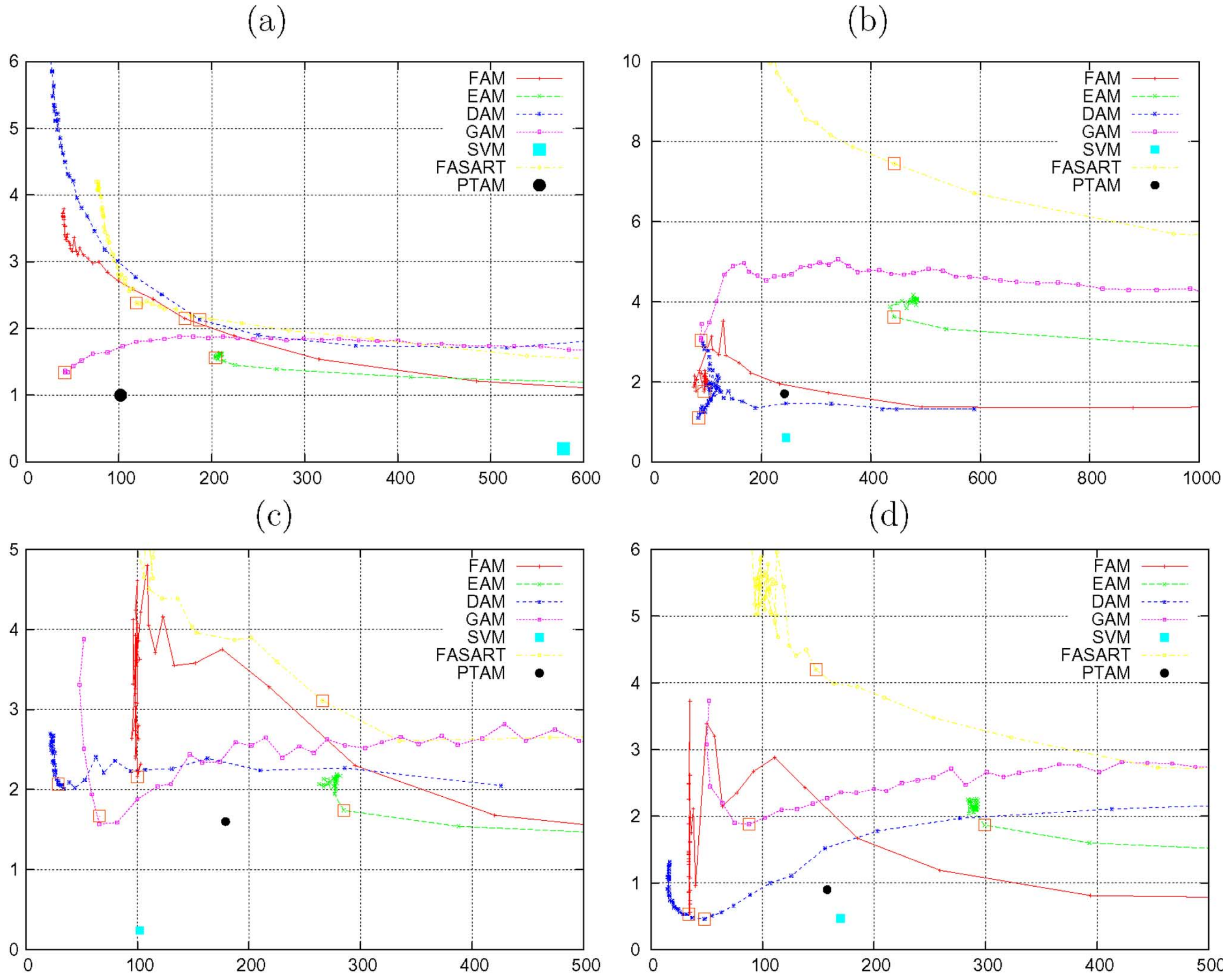


Fig. 9. Error (in percent) against the number of categories on the validation sets in (a) CIS, (b) chess, (c) T3, and (d) T4 varying vigilance. The selected operating point, with the best tradeoff between the error and the number of categories, is marked with an empty square.

the range $0.01:0.01:0.05$ and $0.01:0.01:0.03$ for CIS-noise and T5-noise, respectively.

C. The 2-D Data Sets With Prediction Overlap

The data sets 4G1, 4G2, and 4G3 (labeled 10–12 in Table II) are used in the literature [32] to evaluate classification algorithms when predictions overlap. Each data set has four output predictions, given by Gaussian probability distributions with standard deviation $\sigma = 0.05$. The overlap between them is decreasing with the distance d between its centers and the center of the unit square (Fig. 7).

D. Data Set With Irregular Geometry

Since the previous data sets have circular or rectangular geometries, we generated another 2-D data set, called form (set 13 in Table II) with irregular geometry and, therefore, not specially suited to circular or rectangular category geometries. Form has two predictions associated to the input patterns falling inside and outside of the curve in Fig. 8, which is an example of a mask used for the analysis of irregular textured regions in computer vision [33]. Specifically, this curve is described by the following parametric equations:

$$x = 0.5 + r(\theta)\cos\theta \quad y = 0.5 + r(\theta)\sin\theta, \qquad 0 \le \theta < 2\pi \tag{11}$$

$$r(\theta) = \max\left\{ r_{\min}, r_0 + \sum_{n=0}^{N} r_0 f(n)\cos[(2n+1)\theta + \psi(n)] \right\}. \tag{12}$$

Here, $0 \le r_0 \le 0.5$ is the mean radius of the curve and $0 \le f(n) \le 1$ and $0 \le \psi(n) \le 2\pi$ are pseudorandom numbers. The curve is a circle for $N = 0$ and its irregularity increases with N. We used $r_0 = 0.5$ and $N = 3$. In order to make the input pattern labeling easier, a minimum radius $r_{\min} = 0.08$ was required. The coordinates (x, y) where scaled to have minimum 0 and maximum 1.

E. Real, High-Dimensional Data Sets

Pima Indians diabetes (PID) and abalone (both available from the University of California at Irvine (UCI) Machine Learning

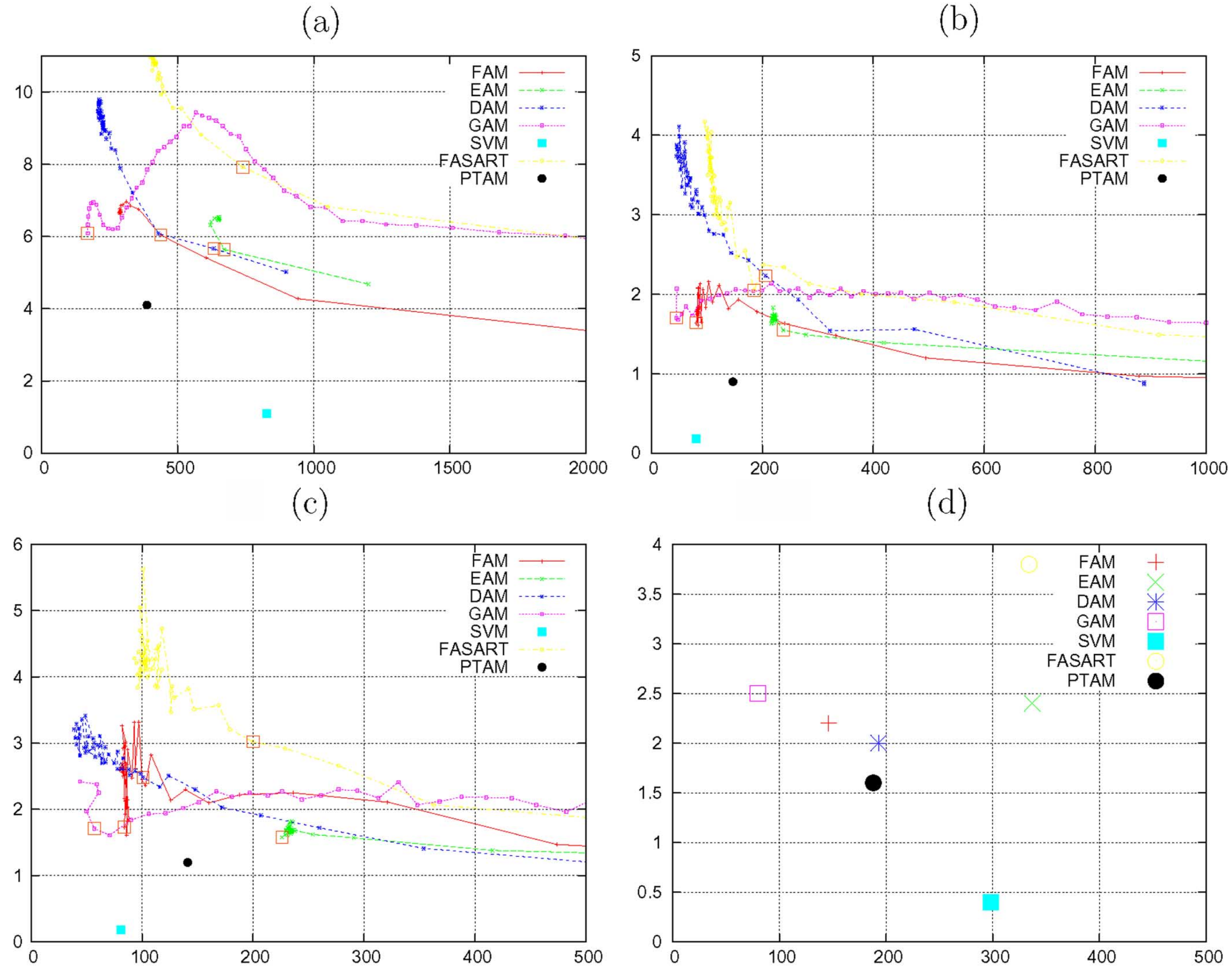


Fig. 10. Error (in percent) against the number of categories in (a) T5, (b) T6, and (c) T7 varying vigilance. The selected operating point is marked with an empty square. (d) Average test results of each classifier.

Repository [34]), are real 8-D data sets (labeled 14 and 15 in Table II) with two and three output predictions, respectively, and they are reported as relatively difficult in the literature. FAM achieved 38% error [25] and safe-μARTMAP [16] 32% error[3] in PID. In abalone, 36% error was reported with multilayer perceptron (MLP) [35] and 50% with FAM (367 categories) [32]. One input is discrete (three values) and the remaining seven inputs are continuous. The output is an integer number in the range $\{1 \ldots 29\}$. For classification tasks, the input patterns must be grouped in three output predictions, associated to outputs 1–8, 9–10, and 11–29. The input patterns in both data sets are preprocessed in such a way that each component falls in the range $[0, 1]$ before feeding to the ART networks. The preprocessing phase associated to the support vector machine (SVM) is the usual mean removal and covariance equalization.

V. Results

We developed experiments comparing PTAM with the most popular ART networks on these data sets, including FAM [2], GAM [3], DAM [4], EAM [5], and FasArt [6]. We also reported the results achieved by SVM [36], which is a reference for classification tasks. The parameter values of the different algorithms are reported in Table III. FAM used the Weber law and choice-by-difference choice functions: The results were very similar, but slightly better using the Weber law. For the ART classifiers, including PTAM, five training epochs were run on data sets 1–7 and one training epoch was run on data sets 8–14, in order to reduce the computational cost of the simulations. FasArt used $\gamma_a = \gamma_b = 1$ in data sets PID and abalone, because when using the recommended value $\gamma_a = 10$ the categories are too narrow and they do not cover the whole input space. We used the SVM implementation provided by the Torch Machine Learning Library [37] with polynomial, sigmoid, dot-product, and Gaussian kernels, which usually provided the best results.

[3] In [16], the authors use 576 training and 192 testing patterns. We use 192 training, 384 validation, and 192 testing patterns for all the neural networks. We are only interested in the comparison between them, so the small size of the training set is not expected to reduce the significance of our experiments.

TABLE V

TEST ERROR (ϵ, IN PERCENT) AND #C [NUMBER OF WEIGHT VECTORS (#W) FOR PTAM AND NUMBER OF SUPPORT VECTORS (#SV) FOR SVM] ACHIEVED BY EACH CLASSIFIER IN 2-D DATA SETS. THE BEST ϵ AND #C ACHIEVED BY AN ART AND NON-ART CLASSIFIER ARE IN BOLD

	PTAM		FAM		GAM		DAM		EAM		FasArt		SVM	
	ϵ	#W	ϵ	#C	ϵ	#C	ϵ	#C	ϵ	#C	ϵ	#C	ϵ	#SV
CIS	**1.0**	102	1.3	188	1.3	**43**	1.3	238	1.3	195	1.8	350	**0.2**	578
Chess	1.7	242	1.8	95	3.0	90	**1.1**	**89**	3.4	443	4.7	443	**0.6**	245
T3	**1.6**	179	2.2	100	**1.6**	67	2.1	**42**	1.7	285	3.1	266	**0.2**	102
T4	1.0	158	0.5	**34**	1.9	89	**0.4**	48	1.9	300	4.1	149	0.5	170
T5	**4.1**	387	5.8	439	6.1	**169**	5.6	630	5.6	670	7.8	739	**1.1**	827
T6	**0.9**	147	1.6	81	1.7	**45**	1.6	193	1.5	239	2.1	185	**0.2**	81
T7	**1.2**	141	1.9	85	1.8	**57**	1.9	113	1.7	226	3.1	207	**0.2**	81
Avg.	**1.6**	188	2.2	146	2.5	**80**	2.0	193	2.4	337	3.8	334	**0.4**	298
Std.	**1.1**	100	1.7	137	1.7	**44**	1.7	206	1.5	167	2.0	205	**0.3**	291

For each tuning parameter, the value with the best average performance over the validation sets was selected for each classifier and data set (Table IV).

Some authors [25], [16] report the results achieved by the ART networks with zero baseline vigilance ($\bar{\rho} = 0$). However, the error and number of categories achieved by an ART network strongly depend on the vigilance value, as Figs. 9 and 10 show. We used cross validation in order to determine the best $\bar{\rho}$ for each data set. For data sets 1–12, 20 triplets, each one composed of one training, one validation, and one test set, were created, containing 10 000 randomly generated patterns in each set. For each ART classifier (FAM, GAM, DAM, EAM, and FasArt), we tried $\bar{\rho}$ values in the range $0.00 : 0.02 : 0.98$. For each vigilance value, 20 versions of the classifiers were trained, one for each training set, and they were tested on its corresponding validation set. Figs. 9 and 10 represent the percentage of error rate against the number of categories #C [number of weight vectors (#W) for PTAM or support vectors (#SV) for SVM], with the different vigilance values and for each data set, averaged over the 20 validation sets. The vigilance value ρ which provides the best tradeoff between the error and #C on the validation set (operating point, marked with an empty square in Figs. 9 and 10) was selected for the test phase. For example, the FAM operating point for data set CIS [Fig. 9(a)] has 2.15% error, 171 categories, and $\bar{\rho} = 0.88$. The best results were usually achieved with high vigilance, as reported in Table IV. GAM is a noticeable exception, because its error and #C increase with the vigilance, so its selected value is often $\rho \simeq 0$.

The 20 classifiers trained with the selected vigilance value were tested on the 20 test sets. The average error rate and #C over the test sets are reported in Table V and represented in Fig. 10(d). The same experimental methodology is used with the noisy 2-D data sets CIS-noise and T5-noise, and with the overlap data sets 4G1, 4G2, and 4G3.

Fig. 11(a) and (c) shows the CRRs created by PTAM for data sets chess and T5, respectively, which are examples of rectangular and circular geometries respectively. Fig. 11(b) and (d) shows the weight vectors and the borders among predictions created by PTAM, which achieve quite accurately the desired borders. The category borders (line segments in $\mathbb{R}^2$) in Fig. 11(a) and (c) compose a piecewise linear approximation to the borders among the output predictions. The lower left corner of Fig. 11(a) shows that sometimes polytope categories may not cover the whole input space. However, the regions not covered are assigned to the right output prediction, as Fig. 11(b) shows.

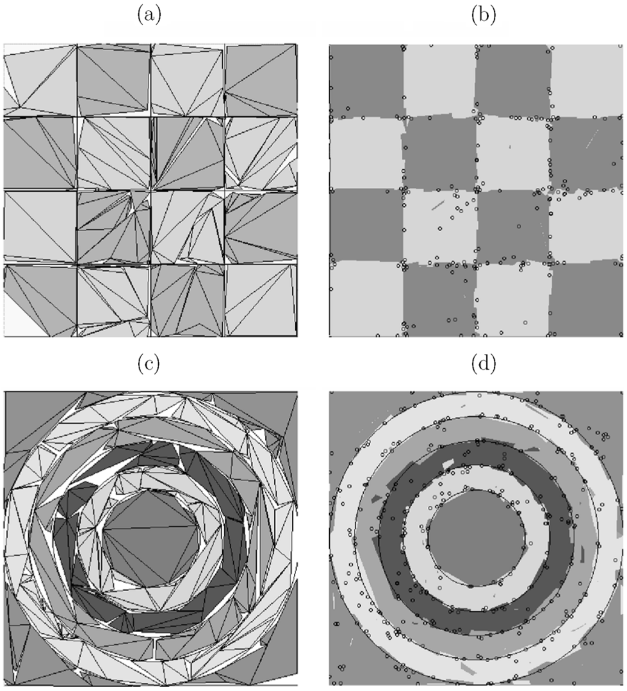


Fig. 11. Examples of categories created by (a) PTAM and (b) classification regions for data set chess. (c) and (d) Idem for data set T5.

A. The 2-D Data Sets

Table V reports the test results of the seven classifiers for 2-D data sets 1–7. SVM achieves the lowest error for all the data sets except in T4. However, SVM generates in average more SVs than categories in PTAM, FAM, GAM, and DAM. PTAM achieves the lowest error among the ART networks in five of seven data sets (CIS, T3, T5, T6, and T7). Its error (4.1%) is clearly lower than the other networks in data set T5 (above 5.5%), which is the most complex one. PTAM outperforms the circular ART networks (GAM and EAM) both in rectangular and circular data sets (GAM achieves the equal error in T3).

TABLE VI
ERROR (ϵ, IN PERCENT) AND #C ACHIEVED IN [20] BY FAM, DAM, AND FASART IN CIS AND T3–T7

	FAM		DAM		FasArt	
	ϵ	#C	ϵ	#C	ϵ	#C
CIS	6.3	23.2	11.1	12.3	3.7	63.4
T3	11.5	53.1	12.1	33.4	7.3	122.3
T4	3.8	31.9	4.3	30.7	8.2	139.2
T5	15.5	124.8	19.1	125.4	4.0	278.7
T6	5.2	21.2	21.1	8.6	3.6	62.2
T7	5.4	20.7	4.7	12.9	3.7	57.7

TABLE VII
NUMBER OF CATEGORIES (N_c), SIMPLEXES (N_s), AND PARAMETERS N_p DIVIDED BY THE DIMENSION n OF THE INPUT SPACE ($n = 2$ FOR ALL THE DATA SETS IN THE TABLE) STORED BY PTAM, FAM, AND DAM

	PTAM			FAM		DAM	
Data set	N_s	N_w	N_p/n	N_c	N_p/n	N_c	N_p/n
CIS	78	102	**258**	188	**376**	238	**1071**
Chess	183	242	**608**	95	**190**	89	**401**
T3	134	179	**447**	100	**200**	42	**189**
T4	122	158	**402**	34	**68**	48	**216**
T5	282	387	**951**	439	**878**	630	**2835**
T6	114	147	**375**	81	**162**	193	**869**
T7	101	141	**343**	85	**170**	113	**509**
Average	145	194	**483**	146	**292**	193	**870**
Std.	69	95	233	137	274	206	926

Also, PTAM achieves the second lowest error in the rectangular data set chess, outperforming FAM. PTAM creates an intermediate number of vectors, less than DAM, EAM, and FasArt, but more than GAM and FAM.

DAM achieves the lowest error in data sets chess and T4, which have rectangular geometry. FAM achieves an error very similar to, although slightly higher than, DAM in all the data sets. They achieve errors above PTAM, GAM, and EAM in circular data set T3. FAM creates less categories than DAM, except in chess and T3. GAM and EAM achieve similar errors, which are high compared to PTAM, DAM, and FAM, specially in the rectangular data sets. However, EAM creates the highest number of categories and GAM the lowest one. Finally, FasArt achieves the highest error both in circular and rectangular data sets, with a high number of categories.

Results in data sets CIS and T3–T7 are different from [20] (Table VI), where the reported errors are higher, and the number of categories is lower, than in our experiments. Some possible reasons are as follows: 1) the differences in the number of training and test patterns (2000 patterns in [20]) and in the category populations for some data sets and 2) we optimize the vigilance value in the ART networks ($\bar{\rho} \neq 0$ in our simulations except for GAM) while FAM, DAM, and FasArt use $\bar{\rho} = 0$ in [20]. Hence, we achieve lower error and higher #C with FasArt, FAM, and DAM. This may be the cause of the good results achieved by FAM and DAM in our simulations, compared to FasArt, with respect to the cited paper.

It is interesting to compare the number of parameters (N_p) stored by the different networks. In the case of PTAM, $N_p = (n+2)N_s + nN_w$, where N_s and N_w are, respectively, the number of simplexes and weight vectors (see Appendix IV). Since each category in PTAM requires several vectors, it usually stores more parameters than the ART networks. This is expectable, since the polytope categories of PTAM are more flexible than hyperbox categories. Table VII reports N_p, divided by n (dimension of the input space), for PTAM, FAM, and DAM, in the seven 2-D data sets. In average, PTAM stores 65% more parameters than FAM, for which $N_p = 2nN_c$ (N_c is the number of categories). On the other hand, the number of parameters of DAM is given by $N_p = (4n+1)N_c$, and it stores 80% more parameters than PTAM (DAM stores less parameters than PTAM only in data sets chess, T3, and T4).

B. Data Sets With Noise

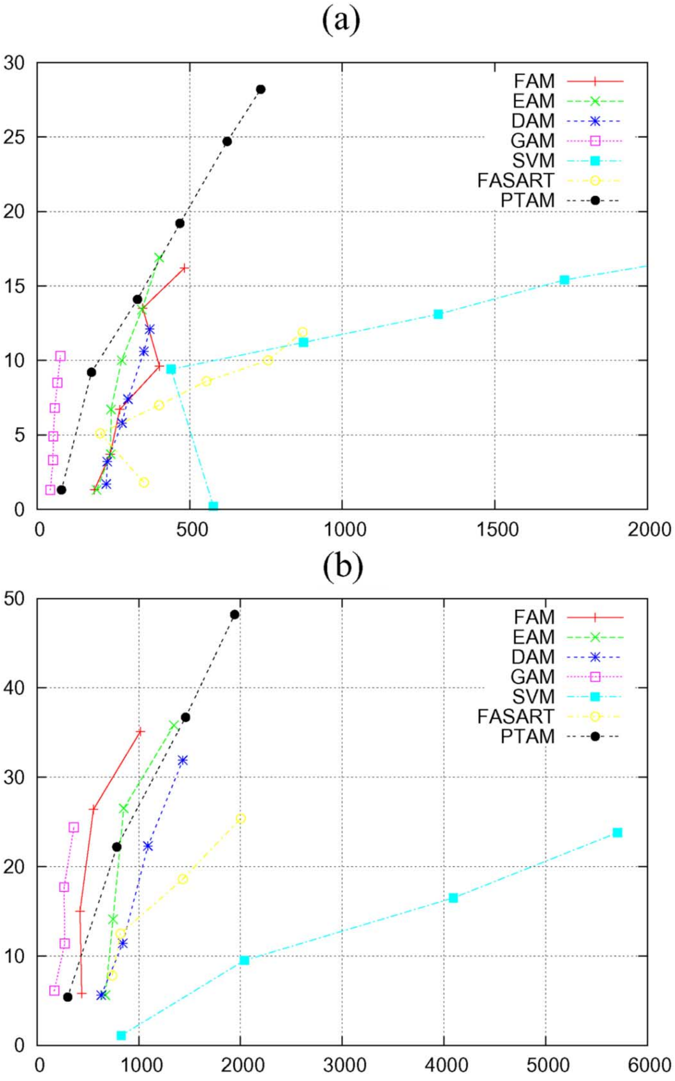


Fig. 12. Average error (in percent) against the number of categories in (a) CIS-noise and (b) T5-noise. Each point corresponds to a different noise level (0.00 : 0.01 : 0.05 in CIS-noise and 0.00 : 0.01 : 0.03 in T5-noise).

All the classifiers increase their error and #C with the noise level, as reported in Table VIII and Fig. 12 in data sets CIS-noise and T5-noise. The table shows also the Bayes error [38] in both data sets. GAM achieves the lowest error (10.3% with the highest noise level) in CIS-noise for all the noise levels. In T5-noise, GAM achieves slightly higher error than SVM (23.8%, equal to the Bayes error). However, SVM uses a huge number of SVs, while the #C of GAM increases very slowly

TABLE VIII
TEST RESULTS FOR 2-D DATA SETS WITH NOISE. THE BEST ERROR (ϵ, IN PERCENT) AND #C ACHIEVED BY AN ART AND NON-ART CLASSIFIER ARE IN BOLD

		CIS-noise					T5-noise		
Method		0.01	0.02	0.03	0.04	0.05	0.01	0.02	0.03
Bayes	ϵ	1.7	3.3	5.0	6.5	8.0	8.2	16.1	23.8
PTAM	ϵ	9.2	14.1	19.2	24.7	28.3	21.9	36.6	46.8
	#W	178	329	468	623	733	759	1421	1890
FAM	ϵ	3.7	6.7	9.6	13.5	16.2	15.0	26.4	35.1
	#C	240	271	401	346	483	421	554	1014
GAM	ϵ	**3.3**	**4.9**	**6.8**	**8.5**	**10.3**	**11.4**	**17.7**	**24.4**
	#C	**52**	**53**	**58**	**67**	**76**	**272**	**265**	**362**
DAM	ϵ	3.2	5.8	7.4	10.6	12.1	11.5	22.3	31.9
	#C	230	278	298	350	369	842	1090	1432
EAM	ϵ	3.7	6.7	10.0	13.4	16.9	14.1	26.5	35.8
	#C	241	242	277	343	400	746	851	1344
FasArt	ϵ	5.1	7.0	8.6	10.0	11.9	12.5	18.6	25.4
	#C	207	400	555	757	870	819	1433	2002
SVM	ϵ	9.4	11.2	13.1	15.4	16.7	**9.5**	**16.5**	**23.8**
	#SV	439	873	1315	1728	2105	2039	4094	5707

with the noise level. Remarkably, GAM creates less categories in CIS-noise with noise level 0.05 than the other classifiers without noise. FAM, DAM, and EAM report similar number of categories for each noise level, although DAM achieves slightly lower error than FAM and EAM. FasArt achieves similar error to DAM in CIS-noise with high noise level, but with more categories. PTAM achieves higher error than the other ART networks for the different noise levels, and it creates rather high number of categories, but lower than FasArt and SVM.

C. Data Sets With Prediction Overlap

In the overlap category data sets 4G1, 4G2, and 4G3, PTAM also reports poor performance (Table IX). The first row in this table reports the Bayes errors for each data set. GAM shows the best global behavior, because it achieves the Bayes error on each data set with a very low number of categories. FasArt and SVM also achieve the Bayes error, but their complexity (number of categories or SVs) is higher—specially SVM. DAM achieves higher error than GAM with a similar complexity. EAM achieves higher error than DAM, with much more categories. FAM achieves clearly higher error than EAM and DAM, with an intermediate number of categories. PTAM achieves higher error than the other classifiers, and it only creates less vectors than FasArt and SVM.

D. Irregular Geometry Data Set

We also compared PTAM with the best rectangular ART networks (DAM and FAM), and with the best circular ART network (GAM), in the irregular data set form. Fig. 13(a) shows an example of the polytope categories and Fig. 13(b) the borders among predictions created by PTAM for this data set. FAM, DAM, and GAM were trained and tested using the same cross-validation methodology as in the other 2-D data sets. Fig. 14 shows the validation results achieved by FAM, DAM, and GAM (varying $\bar{\rho}$ in the range $0.00:0.02:0.98$), and PTAM, and Table X reports the test results. PTAM achieves lower test error (2.3%) than the best rectangular and circular ART networks. GAM achieves higher error than PTAM for all the vigilance values, and the test error is twice the error of PTAM. FAM and DAM need about 1000 categories to achieve the same error as PTAM (Fig. 14). We selected two operating points for DAM and other two points for FAM (marked with empty squares in the figure). DAM achieves higher error (2.7%) than PTAM using $\bar{\rho} = 0.94$, and more categories (572). Using $\bar{\rho} = 0.88$, DAM also achieves higher error (3.9%) than PTAM, with slightly less categories (270). FAM achieves higher error than PTAM for the two selected vigilance values.

TABLE IX
TEST ERROR (ϵ, IN PERCENT) AND #C IN DATA SETS WITH PREDICTION OVERLAP. THE BEST ϵ AND #C ACHIEVED BY AN ART AND NON-ART CLASSIFIER ARE IN BOLD

	4G1		4G2		4G3	
	ϵ	#C	ϵ	#C	ϵ	#C
Bayes	5.7	–	8.7	–	21.8	–
PTAM	25.6	632	34.0	905	52.5	1869
FAM	15.0	212	19.3	372	39.2	801
GAM	**5.8**	**41**	**8.8**	**51**	22.3	**123**
DAM	10.3	52	14.5	70	32.5	161
EAM	11.9	425	17.2	582	35.2	1937
FasArt	**5.8**	703	**8.8**	1000	**21.9**	2250
SVM	**5.8**	1551	**8.7**	2352	**21.9**	5173

E. Real, High-Dimensional Data Sets

High-dimensional data sets PID and abalone have a reduced number of patterns, so the usual experimental procedure [16] is Kfold cross validation with $k = 4$. In order to optimize the vigilance parameter, we created 20 training, 20 validations, and

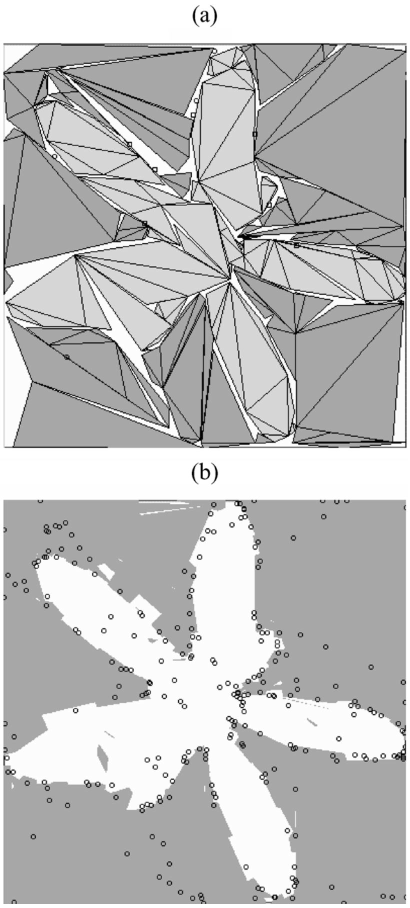


Fig. 13. Example of (a) categories and (b) classification regions and weight vectors created by PTAM in form.

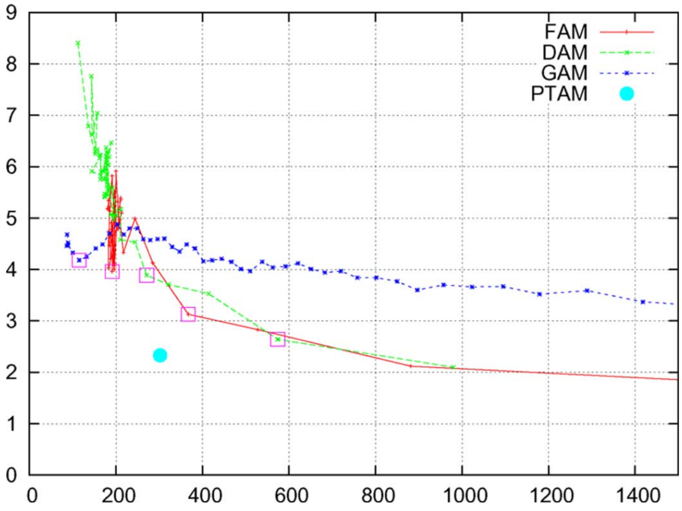


Fig. 14. Average validation error (in percent) and #C achieved by FAM, DAM, and GAM (varying vigilance) and by PTAM in data set form.

TABLE X

TEST ERROR (IN PERCENT) AND #C ACHIEVED BY PTAM AND THE BEST RECTANGULAR AND CIRCULAR ART NETWORKS (FOR WHICH THE VALUE OF $\bar{\rho}$ IS REPORTED) IN THE FORM DATA SET. THE BEST ERROR AND #C ARE IN BOLD

	PTAM	GAM	DAM		FAM	
$\bar{\rho}$	–	0.12	0.88	0.94	0.36	0.92
Error (%)	**2.3**	4.3	3.9	2.7	4.1	3.3
#C	302	**115**	270	572	190	368

20 test sets, containing 25%, 50%, and 25% of the input patterns, respectively, randomly selected from the original data set. Specifically, we used 192 training, 384 validations, and 192 test patterns in PID, and 1044 training, 2088 validations, and 1044 test patterns in abalone. Baseline vigilance values $\bar{\rho}$ on the set $\{0.0, 0.2, 0.4, 0.6, 0.8, 0.95\}$ were tried, and the value with the best average tradeoff between the error and #C over the 20 validation sets was selected (Table IV).

Table XI reports the average results of each classifier with its best vigilance value over the 20 test sets. SVM is clearly the best classifier in PID (24%) and abalone (34%), with 52% and 73% of the training patterns as SVs, respectively. In the PID data set, this error was also achieved by AFC [25] using 576 training and 192 test patterns, and with zero vigilance. FasArt, DAM, and FAM achieve the lowest error among the ART networks (about 30%), but FAM creates twice the number of categories (60) than FasArt (29) and DAM (25). GAM and EAM achieve higher error (34%–37%), with few categories. PTAM achieves the highest error (43%) and number of categories (74% of the training patterns). In the abalone data set, FAM and DAM achieve the lowest error (43%), but FAM creates the highest #C (370), and DAM creates only 131 categories. FasArt and EAM achieve higher error than FAM and DAM, with less categories. PTAM achieves 50% error, only below GAM (57%), but it is similar to the one reported by FAM (50%) in [32]. Also, PTAM creates the lowest number of categories (27) among the ART networks.

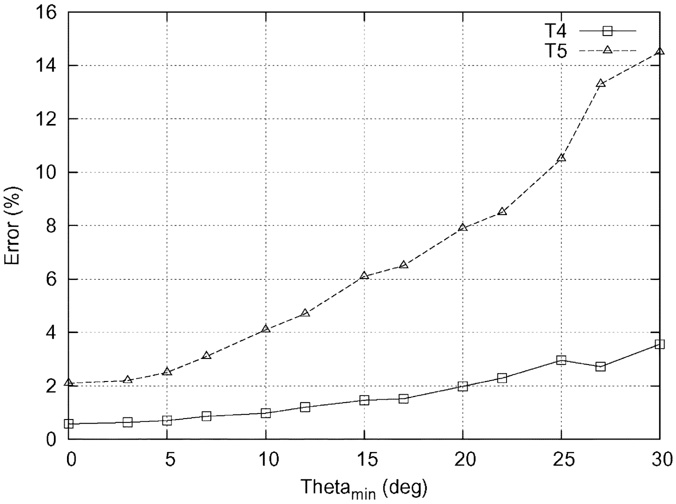
VI. DISCUSSION

The average test results (error and #C) obtained by the different classifiers in the 2-D data sets 1–7 are reported in the two bottom rows of Table V and represented in Fig. 10(d). SVM achieves the lowest global error (0.4%), with a high number of SVs (298). PTAM achieves the lowest average error among the ART networks (1.6%), and the lowest error in five of seven data sets. In the other two data sets, PTAM does not achieve much higher error than the best ART network. Only GAM and FAM create less categories (44 and 137, respectively) than PTAM (188). DAM achieves the second lowest error (2%), with #C (193) similar to PTAM. However, DAM only achieves lower error than PTAM in the rectangular data sets chess and T4. FAM achieves higher error than DAM, and less categories. EAM achieves equal error as GAM, and it creates the highest #C. FasArt achieves the highest error (3.8%) and #C. PTAM also achieves lower error than the best rectangular and circular ART networks (DAM and GAM), with a low number of categories, on the irregular geometry data set form (Table X).

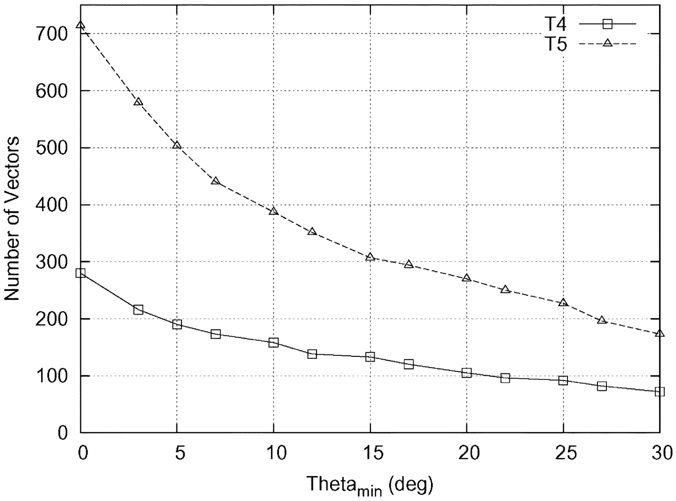
PTAM creates a new simplex during resonance only if the resonant category is not the most active one (Section III-D). We developed experiments in which a new simplex is created every time an input pattern falls outside the resonant category. In these experiments, the average test error and #W achieved by PTAM in data sets 1–7 are 1.4% and 262 weight vectors. The error reduces by 0.2, but the number of weight vectors #W increases

TABLE XI
TEST ERROR (ϵ, IN PERCENT) AND #C IN HIGH-DIMENSIONAL DATA SETS PID AND ABALONE. THE BEST ϵ AND #C ACHIEVED BY AN ART AND NON-ART CLASSIFIER ARE IN BOLD

		PTAM	FAM	GAM	DAM	EAM	FasArt	SVM
PID	ϵ	43.6	**31.1**	37.4	**30.9**	34.3	**30.8**	**24.5**
	#C	142	63	**19**	25	22	29	100
Abalone	ϵ	50.3	**42.9**	57.4	**43.3**	47.6	44.1	**34.2**
	#C	**27**	369	60	131	174	341	764



(a)



(b)

Fig. 15. (a) Error and (b) number of vectors achieved by PTAM against $\theta_{\min}$ in data sets T4 and T5.

by 39%. Thus, the creation of new simplexes in all the cases does not significantly reduce the error rate, but it contributes to category proliferation. This result justifies the creation of a new simplex only if the resonant category is not the most active one.

We also did experiments related to the parameter $\theta_{\min}$ used to discard acute simplexes (Section III-B1). Fig. 15 shows the typical behavior of error and number of vectors achieved by PTAM varying $\theta_{\min}$ in T4 and T5 (they were selected as representants of rectangular and circular geometries, but we found the same behavior in the other data sets). High $\theta_{\min}$ values increase the error and reduce the number of vectors, because too many simplexes are discarded, so PTAM is unable to fit the borders among the output predictions. With low $\theta_{\min}$ values, PTAM discards few simplexes, so its learning ability is higher, the error is low, and the number of vectors is high. If $\theta_{\min} = 0$, no simplex is discarded, so the number of simplexes and vectors boosts [left end of Fig. 15(b)]. Therefore, a low $\theta_{\min}$ value must be used to discard only the acute simplexes. Based on the experiments developed with several data sets, we selected the value $\theta_{\min} = 10$ degree as acceptable. Given that the minimum angle to consider a simplex as acute does not seem to depend on the data set, this value should be valid for any data set, so it does not need to be optimized via cross validation.

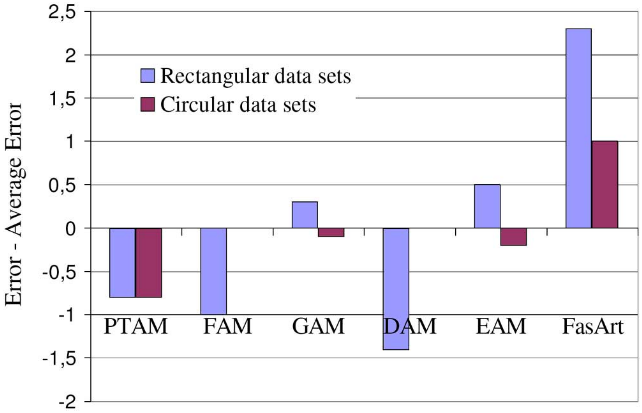


Fig. 16. Difference between the error (in percent) achieved by each ART classifier and the average error for each data set, grouped in rectangular and circular data sets.

It is interesting to evaluate the behavior of each classifier depending on the data set geometry. Table V reports that PTAM is the ART network with the lowest standard deviation in the error (1.1) compared to EAM (1.5), FAM, GAM, and DAM (1.7), and FasArt (2.0). Thus, the results of the ART networks are more dependent on the data set than the results of PTAM. We grouped the results of each classifier for rectangular and circular data sets. For each one, we calculated the average error over all the ART networks. Then, we calculated the difference between the error achieved by each classifier and the average error (this difference is negative if the error is lower than the average, and it is positive otherwise). Fig. 16 shows the average differences for each classifier over the circular (CIS, T3, and T5–T7) and over the rectangular data sets (chess and T4). FAM and DAM have high negative difference for the rectangular data sets and zero difference for the circular ones. The circular classifiers (GAM and EAM) have negative (positive) differences for circular (rectangular) data sets. FasArt has high positive difference for rectangular and circular data sets. Thus, the behavior of both circular and rectangular classifiers depend on the data set geometry, and their average results are better (negative differences) for data sets with their equal geometry. PTAM is the only network with high negative difference for both geometries of data sets, because its irregular polytope categories are not specially suited to any particular geometry.

The good results of PTAM are possible without vigilance and, consequently, without any parameter tuning. This feature makes PTAM easier to use than the other ART networks, because this tuning often requires cross-validation trials, which may have an important cost in processing time. On the other hand, these trials require validation sets, so that the number of available training patterns is reduced, which may difficult learning. This might be a drawback in real data sets, where the number of available input patterns is often limited by the cost of each data acquisition.

The results on data sets CIS-noise and T5-noise (Table VIII), and 4G1, 4G2, and 4G3 (Table IX) show that PTAM achieves higher error than the other ART networks. PTAM also creates a high number of categories, although it is lower than FasArt and, specially, SVM. PTAM approximates the borders among the output predictions by strictly following the information contained in the training set, both in the category expansion and adjustment steps. In the presence of noise or prediction overlap, the input patterns belonging to different categories are mixed and they break simplexes in the category adjustment step, which leads to the creation of noisy single-vector categories. Future work will use statistical information in the category expansion and adjustment steps in order to make PTAM robust to noise and category overlap. The behavior of PTAM for real, high-dimensional data sets PID and abalone (Table XI) is worse than the other ART networks, with the highest error in data set PID, and a variable number of categories.

A. Computational Complexity

The computational cost of each iteration (processing of a training pattern) in PTAM is the following. The cost of functions $\phi_{ijk}(\mathbf{I})$ and $g_{ijk}(\mathbf{I})$ [see (3) and (2), respectively] is $\mathcal{O}(n^3)$, because they use n-order determinants [39]. In order to evaluate if the input pattern falls inside the simplex, $n+1$ values of $g_{ijk}(\mathbf{I})$ must be calculated, which is $\mathcal{O}(n^4)$. If the pattern falls outside the simplex, but inside the hypersphere [see (4)], $\|\mathbf{w}^*_{ijk}\|$ must be calculated for the $n+1$ hyperplanes in S_{ij}. Since n determinants D_{ijkl} [see (17)] must be calculated, $\|\mathbf{w}^*_{ijk}\|$ is $\mathcal{O}(n^4)$, so $d(\mathbf{I}, S_{ij})$ [see (4)] is $\mathcal{O}(n^5)$. If the pattern falls inside the hypersphere, $d(\mathbf{I}, S_{ij})$ [see (5)] is $\mathcal{O}(n^2)$. Therefore, $T_{ij}(\mathbf{I})$ and $T_i(\mathbf{I})$ are $\mathcal{O}(n^5)$. Since the system of n linear equations in I_{sh} [see (22)] is $\mathcal{O}(n^2)$, the function O_{ls} [see (23)] is $\mathcal{O}(n^3)$, because it requires to calculate I_{sh} for the $n+1$ hyperplanes of a simplex. The calculation of set $\mathcal{A}_I(\mathbf{I})$ (Section III-B) is $\mathcal{O}(n^4)$, because $(n+N^s_I)\sum_{i=1}^{N_c} N^s_i$ values of O_{ls} must be calculated.[4] The function O_{ss} [see (24)] for the OT is $\mathcal{O}(n^5)$, because $(n+1)^2$ values of O_{ls} must be calculated, so the creation of a new simplex is $\mathcal{O}(n^5)$. The removal of acute simplexes is $\mathcal{O}(n^4)$ [see (7)], because it requires to calculate n^3 angles, which are $\mathcal{O}(n)$ each one. The vector replacement step is $\mathcal{O}(n^4)$, because $n + N^s_I$ values of $g_{Ijk}(\mathbf{I})$ (which is $\mathcal{O}(n^3)$) must be calculated, but the OT for the modified simplex is also $\mathcal{O}(n^5)$. The costs of the PT, category adjustment, and resonance steps do not depend on n. The creation of a new category requires $\sum_{i=1}^{N_c}(n+N^s_i)$ calculations of O_{ls}, which is $\mathcal{O}(n^4)$, in order to select the connectable vectors from $\mathbf{I}$, and the OT for the new simplex, which is $\mathcal{O}(n^5)$.

Overall, the cost of PTAM for each iteration is $\mathcal{O}(n^5)$. While this high cost makes the use of PTAM with high-dimensional data sets difficult, for n low, its speed is comparable to the other ART networks, which require cross-validation trials. Specifically, the current implementation of PTAM takes about 40 s. for the CIS data set (10 000 training and 10 000 test patterns) on a general purpose personal computer (PC) Pentium IV with 512-MB random access memory (RAM), and it runs much faster than DAM (560 s), three times slower than GAM and EAM (15 s), and eight times slower than FAM and FasArt (5 s).

VII. Conclusion and Future Work

This paper analyzes the utility of general, irregular geometries for the internal categories of the ART networks. The categories in PTAM are n-dimensional polytopes whose outlines are a piecewise linear approximation to the borders among the output predictions. During the learning phase, the polytope categories are expanded towards the input pattern, covering only the regions of the input space populated by training patterns with equal category prediction. They can expand without overlap, as opposite to the other ART networks, so the category expansion is naturally limited by the other categories, instead of the category size. The vigilance parameter is not required, so PTAM has no tuning parameter.

Each polytope category is represented internally as a set of simplexes, whose vertices are training patterns. The CCF is 1 inside the polytope and decreasing with the distance to the polytope outside it. The category with the highest CCF tries to expand towards the input pattern avoiding category overlap, either creating a new simplex or replacing a vertex by the input pattern. A simplex is removed when a training pattern with a different prediction falls inside it. When no category with the desired prediction can expand towards the input pattern without overlap, PTAM creates a new single-vector category.

Experimental results on a complete collection of 2-D data sets show that PTAM achieves lower error than the leading ART networks, with a similar number of categories, and they achieve higher error than SVM. The irregular geometry of categories in PTAM provides less dependence on the data set geometry (circular, rectangular) than the other ART networks. Hence, PTAM is clearly better than the best rectangular and circular ART networks on a data set with irregular geometry. We would like to emphasize that PTAM achieves these results without the vigilance parameter. The absence of tuning parameters in PTAM reports the following advantages. 1) The nonexpert user is not concerned about the internal operation of PTAM, which is easier to use. Although ART networks can use zero vigilance, it may not be the best value for a given data set, so it must be tuned, e.g.,

[4] Each polytope category C_i with $N^s_i > 0$ is defined by $(n+1)+(N^s_i-1) = n + N^s_i$ weight vectors.

using cross validation. 2) The input patterns devoted to cross validation sets cannot be used for training. This fact may be important for real data sets, which often have a reduced number of input patterns. PTAM is rather sensible to noisy outliers or prediction overlap and it does not outperform the leading ART networks on real, high-dimensional data sets, which suggests that its ability to model complex borders among categories reduces with the dimensionality of the input space.

The main objective of future research is to simplify PTAM in order to decrease its computational cost and to get an efficient implementation. We have analyzed the behavior of PTAM when category overlap is allowed [40], and the conclusion is that categories with irregular geometry do not work better if they overlap. In our future work, we intend to avoid the idea that categories cover regions where predictions overlap, without simplex breaking. Another objective is to adapt the category overlap to the distribution of training patterns, in order to reduce the sensitivity to noise and prediction overlap. We are also working to replace simplexes by hyperplanes in the definition of the polytopes, in order to overcome some limitations in the category expansion associated to simplexes. Distributed learning with polytope geometries is another interesting issue for future research. After these improvements, we will evaluate the performance of PTAM using additional real-world data sets.

Appendix I
Equation of the Hyperplane Defined by $N+1$ Vectors

Let h_{ijk} be the hyperplane in $\mathbb{R}^n$ defined by weight vectors $\mathbf{w}_{ijk1},\ldots,\mathbf{w}_{ijkn}$, and $g_{ijk}(\mathbf{x})=0$ the equation of h_{ijk}. From elementary geometry, we know that every point $\mathbf{x}\in h_{ijk}$ is a linear combination of vectors in the set $\{\mathbf{w}_{ijk1}\}\cup\{\mathbf{w}_{ijkl}-\mathbf{w}_{ijk1}, l=2,\ldots,n\}$, so $\exists\lambda_2,\ldots,\lambda_n\in\mathbb{R}$ which verify

$$\mathbf{x}=\mathbf{w}_{ijk1}+\sum_{l=2}^{n}\lambda_l(\mathbf{w}_{ijkl}-\mathbf{w}_{ijk1}). \qquad (13)$$

Thus, the set of vectors $\{\mathbf{x}-\mathbf{w}_{ijk1},\mathbf{w}_{ijk2}-\mathbf{w}_{ijk1},\ldots,\mathbf{w}_{ijkn}-\mathbf{w}_{ijk1}\}$ is linearly dependent, and they compose a singular matrix

$$\begin{vmatrix} x_1-w_{ijk1,1} & \cdots & x_n-w_{ijk1,n}\\ w_{ijk2,1}-w_{ijk1,1} & \cdots & w_{ijk2,n}-w_{ijk1,n}\\ \cdots & \cdots & \cdots\\ w_{ijkn,1}-w_{ijk1,1} & \cdots & w_{ijkn,n}-w_{ijk1,n}\end{vmatrix}=0 \qquad (14)$$

where $w_{ijkl,m}$ is the mth component of $\mathbf{w}_{ijkl}$. Equation (14) is the definition of $\phi_{ijk}(\mathbf{x})$ in (3) (Section III-A1). On the other hand, $g_{ijk}(\mathbf{x})>0$ must imply that $\mathbf{x}$ falls in the side of h_{ijk} inside the simplex S_{ij}, and $g_{ijk}(\mathbf{x})<0$ implies that $\mathbf{x}$ falls outside S_{ij}. Let $\hat{\mathbf{w}}_{ijk}$ be the vertex of S_{ij} which does not belong to h_{ijk}, and $\operatorname{sgn}(x)$ the sign function (Section III-A1). Then, it must be $\operatorname{sgn}(\phi_{ijk}(\mathbf{x}))=\operatorname{sgn}(\phi_{ijk}(\hat{\mathbf{w}}_{ijk}))$ if $\mathbf{x}$ falls inside the simplex, and $\operatorname{sgn}(\phi_{ijk}(\mathbf{x}))\neq\operatorname{sgn}(\phi_{ijk}(\hat{\mathbf{w}}_{ijk}))$ when $\mathbf{x}$ falls outside the simplex. In order to meet this condition, the equation of hyperplane h_{ijk} can be written as $\operatorname{sgn}(\phi_{ijk}(\hat{\mathbf{w}}_{ijk}))\phi_{ijk}(\mathbf{x})=0$ [see (2)], so that $g_{ijk}(\mathbf{x})\equiv\operatorname{sgn}(\phi_{ijk}(\hat{\mathbf{w}}_{ijk}))\phi_{ijk}(\mathbf{x})$.

Appendix II
Calculation of the Direction Vector of a Hyperplane

Let $g_{ijk}(\mathbf{x})=0$ [see (2)] be the equation of hyperplane h_{ijk}. In order to calculate its direction vector $\mathbf{w}^*_{ijk}$, this equation must be transformed into the following form:

$$\mathbf{w}^*_{ijk}\mathbf{x}+b_{ijk}=0. \qquad (15)$$

Equation $g_{ijk}(\mathbf{x})=0$ implies $\phi_{ijk}(\mathbf{x})=0$ [if $\phi_{ijk}(\mathbf{x})\neq 0$, $\operatorname{sgn}(\phi_{ijk}(\mathbf{x}))\neq 0$, and $g_{ijk}(\mathbf{x})\neq 0$ by (3)]. Developing the determinant $\phi_{ijk}(\mathbf{x})$ in (3) by the first row and equaling to (15) yields

$$\begin{aligned}\phi_{ijk}(\mathbf{x})&=\sum_{l=1}^{n}(-1)^{l+1}(x_l-w_{ijk1,l})D_{ijkl}\\ &=\sum_{l=1}^{n}(-1)^{l+1}D_{ijkl}x_l-\sum_{l=1}^{n}(-1)^{l+1}w_{ijk1,l}D_{ijkl}\\ &=\mathbf{w}^*_{ijk}\mathbf{x}+b_{ijk}\end{aligned} \qquad (16)$$

where D_{ijkl} is the determinant of matrix in (3) after removing row 1 and column l. If we denote $\delta^{pq}_{ijk}\equiv w_{ijkp,q}-w_{ijk1,q}$, then

$$D_{ijkl}=\begin{vmatrix}\delta^{21}_{ijk} & \cdots & \delta^{2(l-1)}_{ijk} & \delta^{2(l+1)}_{ijk} & \cdots & \delta^{2n}_{ijk}\\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots\\ \delta^{n1}_{ijk} & \cdots & \delta^{n(l-1)}_{ijk} & \delta^{n(l+1)}_{ijk} & \cdots & \delta^{nn}_{ijk}\end{vmatrix}. \qquad (17)$$

Equaling the two sides of the last equality and denoting by $\mathbf{e}_l$ the unitary vector ($\|\mathbf{e}_l\|=1$) in dimension l yields

$$\mathbf{w}^*_{ijk}=\sum_{l=1}^{n}(-1)^{l+1}D_{ijkl}\mathbf{e}_l\ \Rightarrow\ \|\mathbf{w}^*_{ijk}\|=\sqrt{\sum_{l=1}^{n}D^2_{ijkl}}. \qquad (18)$$

The norm of the hyperplane direction vector $\mathbf{w}^*_{ijk}$ is used to calculate the distance between an input pattern $\mathbf{I}$ and the hyperplane h_{ijk} [see (4)].

Appendix III
Geometric Implementation of the OT

The overlap between a simplex and a line segment is tested by using geometric techniques. In order to test if a weight vector is "connectable" from the input pattern, the line segment between them cannot overlap with other simplexes. On the other hand, in order to create a new simplex, or to replace a weight vector by the input pattern, the new or modified simplex cannot overlap with other categories. In the following, we will describe the two situations.

A. Overlap Between a Line Segment and a Simplex

The line segment $\overrightarrow{\mathbf{w}_1\mathbf{w}_2}$ and simplex S_{ij} in category C_i overlap only if the former intersects with some hyperplane h_{ijk} of the latter (note that weight vectors can only be simplex vertices, so they cannot fall inside a simplex). The intersection between the segment $\overrightarrow{\mathbf{w}_1\mathbf{w}_2}$ and the hyperplane h_{ijk} defined by vectors $\{\mathbf{w}_{ijk1},\ldots,\mathbf{w}_{ijkn}\}$ can be tested solving for $\alpha^s,\alpha^h_2,\ldots,\alpha^h_n$ the following vector equation (obtained

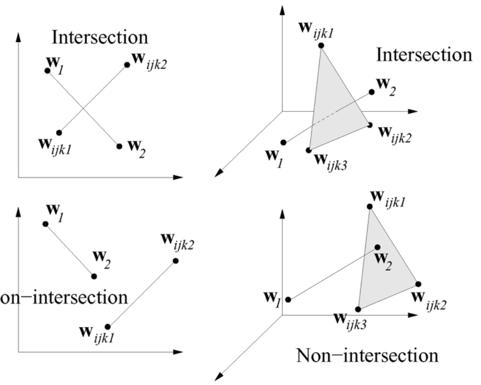


Fig. 17. Examples in $\mathbb{R}^2$ and $\mathbb{R}^3$ of intersection and nonintersection between the segment $\mathbf{w}_1 - \mathbf{w}_2$ and hyperplanes $(\mathbf{w}_{ijk1}, \mathbf{w}_{ijk2})$ in $\mathbb{R}^2$ and $(\mathbf{w}_{ijk1}, \mathbf{w}_{ijk2}, \mathbf{w}_{ijk3})$ in $\mathbb{R}^3$.

from the parametric equations of the straight line and the hyperplane):

$$\mathbf{w}_1 + \alpha^s(\mathbf{w}_2 - \mathbf{w}_1) = \mathbf{w}_{ijk1} + \sum_{l=2}^{n} \alpha_l^h(\mathbf{w}_{ijkl} - \mathbf{w}_{ijk1}). \quad (19)$$

The solution $(\alpha^s, \alpha_2^h, \ldots, \alpha_n^h)$ of this system of n equations is a valid intersection point only if it falls inside the bounded hyperplane area whose border is defined by the n vectors $\mathbf{w}_{ijk1}, \ldots, \mathbf{w}_{ijkn}$ of h_{ijk}, and if it falls in the segment between $\mathbf{w}_1$ and $\mathbf{w}_2$ (Fig. 17). This imposes the following conditions to the solution of (19):

$$0 < \alpha^s < 1 \quad (20)$$
$$0 < \alpha_p^h + \alpha_q^h < 1 \qquad \forall p, q = 2, \ldots, n, \qquad p \neq q. \quad (21)$$

If $\exists \alpha^s, \alpha_2^h, \ldots, \alpha_n^h \in \mathbb{R}$ satisfying (19)–(21), the segment $\overrightarrow{\mathbf{w}_1\mathbf{w}_2}$ and the hyperplane h_{ijk} intersect. We can define the function I_{sh} (intersection segment–hyperplane) in the following way:

$$I_{\text{sh}}(\mathbf{w}_1, \mathbf{w}_2, h_{ijk}) = \begin{cases} 1, & \text{(19)–(21) have a solution} \\ 0, & \text{otherwise} \end{cases}. \quad (22)$$

The function $I_{\text{sh}}(\mathbf{w}_1, \mathbf{w}_2, h_{ijk}) = 1$ if line segment and hyperplane intersect and $I_{\text{sh}}(\mathbf{w}_1, \mathbf{w}_2, h_{ijk}) = 0$, otherwise. Finally, the overlap between line segment $\overrightarrow{\mathbf{w}_1, \mathbf{w}_2}$ and the simplex S_{ij} is given by

$$O_{\text{ls}}(\mathbf{w}_1, \mathbf{w}_2, S_{ij}) = \begin{cases} 1, & \exists h_{ijk} \in S_{ij}\colon\ I_{\text{sh}}(\mathbf{w}_1, \mathbf{w}_2, h_{ijk}) = 1 \\ 0, & \text{otherwise} \end{cases}. \quad (23)$$

B. Overlap Between a Simplex and a Category

If category C_j is a single-vector category $(N_j^s = 0)$ defined by the vector $\mathbf{w}_j$, the simplex S_{ik} and C_j overlap if $T_{ik}(\mathbf{w}_j) = 1$. If C_j is a polytope category $(N_j^s > 0)$, they overlap only if there exists a simplex $S_{jl} \in C_j$ that overlaps with S_{ik}. Let function O_{ss} (overlap between two simplexes) be defined by $O_{\text{ss}}(S_{ik}, S_{jl}) = 1$ if simplexes S_{ik} and S_{jl} overlap and $O_{\text{ss}}(S_{ik}, S_{jl}) = 0$, otherwise. Since the vertices of a simplex cannot be inside another simplex, the simplexes S_{ik} and S_{jl} overlap only if a line segment of S_{ik} and a hyperplane of S_{jl} intersect. Specifically, if $\exists p, q \in \{1, \ldots, n+1\}, p < q$, such as $O_{\text{ls}}(\mathbf{w}_{ikp}, \mathbf{w}_{ikq}, S_{jl}) = 1$ [O_{ls} is defined in (23)], then the line segment $\overrightarrow{\mathbf{w}_{ikp}, \mathbf{w}_{ikq}}$ and the simplex S_{jl} intersect, so $O_{\text{ss}}(S_{ik}, S_{jl}) = 1$. Note that if a line segment of S_{jl} and a hyperplane of S_{ik} intersect, then some line segment of S_{ik} intersects with a hyperplane of S_{jl}, which is the previous case. The overlap function $O_{\text{ss}}(S_{ik}, S_{jl})$ between the simplexes S_{ik} and S_{jl} is given by

$$O_{\text{ss}}(S_{ik}, S_{jl}) = \begin{cases} 1, & \exists p, q = 1, \ldots, n+1, p < q\colon \\ & O_{\text{ls}}(\mathbf{w}_{ikp}, \mathbf{w}_{ikq}, S_{jl}) = 1 \\ 0, & \text{otherwise} \end{cases}. \quad (24)$$

Finally, overlap between the simplex S_{ik} and the category C_j is given by the function $O_{\text{sc}}(S_{ik}, C_j)$, defined by

$$O_{\text{sc}}(S_{ik}, C_j) = \begin{cases} \psi(T_{ik}(\mathbf{w}_j)), & N_j^s = 0 \\ \max_{l=1,\ldots,N_j^s}\{O_{\text{ss}}(S_{ik}, S_{jl})\}, & N_j^s > 0 \end{cases} \quad (25)$$

where the function $\psi(x)$ is given by

$$\psi(x) = \begin{cases} 1 & x = 1 \\ 0 & x < 1 \end{cases}. \quad (26)$$

Thus, if C_j is a single-vector category, it overlaps S_{ik} if $T_{ik}(\mathbf{w}_j) = 1$, so $\psi(T_{ik}(\mathbf{w}_j)) = 1$. Otherwise, if C_j is not a single-vector category $(N_j^s \geq 1)$, it overlaps S_{ik} if some simplex in C_j overlaps S_{ik}.

Appendix IV
Number of Parameters in PTAM

Since each category in PTAM has a variable number of simplexes, its number of weight vectors (vertices) is also variable. If PTAM has N_c categories, N_s simplexes, and N_w weight vectors, the number N_p of parameters which it stores is as follows.

- For each category, the indexes of their simplexes: Given that category C_i has N_i^s simplexes, the number of parameters to store for the N_c categories is $\sum_{i=1}^{N_c} N_i^s = N_s$, because each simplex belongs to exactly one category and there are N_s simplexes.
- For each simplex, the indexes of their weight vectors: Since each simplex has $n+1$ weight vectors, and there are N_s simplexes, the number of parameters is $(n+1)N_s$ (note that $(n+1)N_s > N_w$, since vectors may be shared between simplexes).
- For each weight vector, its n components: Since there are N_w weight vectors, the number of parameters is nN_w.

Finally, the total number of parameters stored by PTAM is

$$N_p^{\text{PTAM}} = (n+2)N_s + nN_w. \quad (27)$$

Comparatively, the number of parameters used by FAM is $N_p^{\text{FAM}} = 2nN_c$ (each internal category is defined by a $2n$-dimensional weight vector, due to complement coding). Finally, each internal category in DAM [4, p. 804] is defined by $4n$ weights $\{\tau_{ij}, \tau_{ji}; i, j = 1, \ldots, 2n\}$ and an instance counter c_j, so the number of parameters in DAM is $N_p^{\text{DAM}} = (4n+1)N_c$.

ACKNOWLEDGMENT

The authors would like to thank Dr. Anagnostopoulos for allowing the use of GAM and EAM simulators, the Supercomputing Center of Galicia (CESGA), and Dr. A. Formella for his useful suggestions about computational geometry.

REFERENCES

[1] G. A. Carpenter and S. Grossberg, "The ART of adaptive pattern recognition by a self-organizing neural network," *Computer*, vol. 21, no. 3, pp. 77–88, 1988.

[2] G. A. Carpenter, S. Grossberg, N. Markuzon, J. H. Reynolds, and D. Rosen, "Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps," *IEEE Trans. Neural Netw.*, vol. 3, no. 5, pp. 698–713, Sep. 1992.

[3] J. Williamson, "Gaussian ARTMAP: A neural network for fast incremental learning of noisy multidimensional maps," *Neural Netw.*, vol. 9, no. 5, pp. 881–897, 1996.

[4] G. A. Carpenter, B. L. Milenova, and B. W. Noeske, "Distributed ARTMAP: A neural network for fast distributed supervised learning," *Neural Netw.*, vol. 11, no. 5, pp. 793–813, 1998.

[5] G. C. Anagnostopoulos and M. Georgiopoulos, "Ellipsoid ART and Ellipsoid ARTMAP for incremental clustering and classification," in *Proc. Int. Joint Conf. Neural Netw.*, 2001, vol. 2, pp. 1221–1226.

[6] J. Cano-Izquierdo, Y. Dimitriadis, E. Gómez-Sánchez, and J. López-Coronado, "Learning from noisy information in FasArt and FasBack neuro-fuzzy systems," *Neural Netw.*, vol. 14, no. 4/5, pp. 407–425, 2001.

[7] R. Andonie and L. Sasu, "Fuzzy ARTMAP with input relevances," *IEEE Trans. Neural Netw.*, vol. 17, no. 4, pp. 929–941, Jul. 2006.

[8] R. Araujo, "Prune-able fuzzy ART neural architecture for robot map learning and navigation in dynamic environments," *IEEE Trans. Neural Netw.*, vol. 17, no. 5, pp. 1235–1249, Sep. 2006.

[9] O. Parsons and G. A. Carpenter, "ARTMAP neural networks for information fusion and data mining: Map production and target recognition methodologies," *Neural Netw.*, vol. 16, no. 7, pp. 1075–1089, 2003.

[10] G. A. Carpenter, S. Martens, and O. J. Ogas, "Self-organizing information fusion and hierarchical knowledge discovery: A new framework using ARTMAP neural networks," *Neural Netw.*, vol. 18, no. 3, pp. 287–295, 2005.

[11] M. F. Yeh and S. S. Chian, "GreyART network for data clustering," *Neurocomput.*, vol. 67, pp. 313–320, 2005.

[12] M. Fernández-Delgado and S. Barro, "A multi-channel ART-based neural network," *IEEE Trans. Neural Netw.*, vol. 9, no. 1, pp. 139–150, Jan. 1998.

[13] B. Vigdor and B. Lerner, "Accurate and fast off and online Fuzzy ARTMAP-based image classification with application to genetic abnormality diagnosis," *IEEE Trans. Neural Netw.*, vol. 17, no. 5, pp. 1288–1300, Sep. 2006.

[14] G. A. Carpenter, "Default ARTMAP," in *Proc. Int. Joint Conf. Neural Netw.*, 2003, pp. 1396–1401.

[15] M. Georgiopoulos, A. Koufakou, G. C. Anagnostopoulos, and T. Kasparis, "Overtraining in Fuzzy ARTMAP: Myth or reality?," in *Proc. Int. Joint Conf. Neural Netw.*, 2001, vol. 2, pp. 1186–1190.

[16] E. Gómez-Sánchez, Y. Dimitriadis, J. M. Cano-Izquierdo, and J. López-Coronado, "Safe μARTMAP: A new solution for reducing category proliferation in Fuzzy ARTMAP," in *Proc. Int. Joint Conf. Neural Netw.*, 2001, vol. 2, pp. 1197–1202.

[17] A. Marriott and R. F. Harrison, "A modified Fuzzy ARTMAP architecture for the approximation of noisy mappings," *Neural Netw.*, vol. 8, no. 4, pp. 619–641, 1995.

[18] S. Verzi, G. L. Heileman, M. Georgiopoulos, and M. J. Healy, "Boosting the performance of ARTMAP," in *Proc. Int. Joint Conf. Neural Netw.*, 1998, pp. 396–401.

[19] A. Koufakou, M. Georgiopoulos, G. Anagnostopoulos, and T. Kasparis, "Cross-validation in Fuzzy ARTMAP for large databases," *Neural Netw.*, vol. 14, no. 9, pp. 1279–1291, 2001.

[20] E. Parrado-Hernández, E. Gómez-Sánchez, and Y. Dimitriadis, "Study of distributed learning as a solution to category proliferation in Fuzzy ARTMAP based neural systems," *Neural Netw.*, vol. 16, no. 7, pp. 1039–1057, 2003.

[21] G. C. Anagnostopoulos and M. Georgiopoulos, "Hypersphere ART and ARTMAP for unsupervised and supervised incremental learning," in *Proc. Int. Joint Conf. Neural Netw.*, 2000, vol. 6, pp. 59–64.

[22] G. C. Anagnostopoulos and M. Georgiopoulos, "Category regions as new geometrical concepts in Fuzzy-ART and Fuzzy-ARTMAP," *Neural Netw.*, vol. 15, no. 10, pp. 1205–1221, 2002.

[23] D. Gomes, M. Fernández-Delgado, and S. Barro, "A vigilance-free ART network with general geometry internal categories," in *Proc. Int. Joint Conf. Neural Netw.*, 2005, pp. 463–468.

[24] ——, "Simplex ARTMAP: Building general geometry borders among predictions with simplex-shaped classes," in *Proc. IASTED-ASC*, 2004, vol. 1, pp. 232–237.

[25] E. P. Sapozhnikova and W. Rosenstiel, "AFC: ART-based Fuzzy Classifier," in *Proc. Int. Conf. Knowl.-Based Intell. Inf. Eng. Syst. (KES)*, 2003, vol. 2, pp. 30–36.

[26] P. K. Simpson, "Fuzzy min-max neural networks—Part 1: Classification," *IEEE Trans. Neural Netw.*, vol. 3, no. 5, pp. 776–786, Sep. 1992.

[27] A. Nandedkar and P. Biswas, "A fuzzy min-max neural network classifier with compensatory neuron architecture," *IEEE Trans. Neural Netw.*, vol. 18, no. 1, pp. 42–54, Jan. 2007.

[28] P. Gritzmann and V. Klee, "On the Complexity of Some Basic Problems in Computational Convexity II. Volume and Mixed Volumes," in *Polytopes: Abstract, Convex and Computational*. Norwell, MA: Kluwer, 1994.

[29] R. Bellman, *Adaptive Control Processes: A Guided Tour*. Princeton, NJ: Princeton Univ. Press, 1961.

[30] G. Wilensky, "Analysis of neural network issues: Scaling, enhanced nodal processing, comparison with standard classification," *DARPA Neural Netw. Program Rev.*, pp. 29–30, 1990.

[31] E. Gómez-Sánchez, Y. Dimitriadis, J. M. Cano-Izquierdo, and J. López-Coronado, "MicroARTMAP: Use of mutual information for category reduction in Fuzzy ARTMAP," in *Proc. Int. Joint Conf. Neural Netw.*, 2000, pp. 647–652.

[32] G. C. Anagnostopoulos and M. Georgiopoulos, "Putting the utility of match tracking in fuzzy ARTMAP training to the test," in *Proc. Int. Conf. Knowl.-Based Intell. Inf. Eng. Syst. (KES)*, 2003, vol. 2, pp. 1–6.

[33] P. García-Sevilla and M. Petrou, "Analysis of irregularly shaped texture regions," *Comput. Vis. Image Understand.*, vol. 84, pp. 62–76, 2001.

[34] C. Blake and C. Merz, "UCI Repository of machine learning databases," Univ. California at Irvine, Irvine, CA, 1998 [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html

[35] D. Clark, Z. Schreter, and A. Adams, "A quantitative comparison of dystal and backpropagation," in *Proc. Austr. Conf. Neural Netw. (ACNN)*, 1996.

[36] V. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.

[37] R. Collobert, S. Bengio, and J. Mariéthoz, "Torch: A modular machine learning software library," IDIAP, Tech. Rep., 2003 [Online]. Available: http://www.idiap.ch/~bengio/publications/pdf/rr02-46.pdf

[38] A. Webb, *Statistical Pattern Recognition*, 2nd ed. New York: Wiley, 2002.

[39] E. Kaltofen and G. Villard, "On the complexity of computing determinants," *Comput. Complexity*, vol. 13, no. 3–4, pp. 91–130, 2004.

[40] D. Gomes, M. Fernández-Delgado, and S. Barro, "Internal categories with irregular geometry and overlapping in ART networks," in *Lecture Notes in Artificial Intelligence*. Berlin, Germany: Springer-Verlag, 2006, vol. 4177, pp. 291–300.

Dinani Gomes Amorim received the B.Sc. degree in computer science from the University Católica of Pernambuco, Recife, Brazil, in 1988. She is currently working toward the Ph.D. degree at the Department of Electronics and Computer Science, University of Santiago de Compostela, Santiago de Compostela, Spain.

Her research interests focus on artificial neural networks (modeling and applications), learning algorithms, classification and clustering, and signal and knowledge processing.

Manuel Fernández Delgado was born in A Coruña, Spain, in 1971. He received the B.S. degree in physics and the Ph.D. degree in computer science from the University of Santiago de Compostela, Santiago de Compostela, Spain, in 1994 and 1999, respectively.

Currently, he is an Assistant Professor of Computer Science at the University of Santiago de Compostela. His research interests include pattern recognition, neural computation, and intelligent monitoring of physiological signals.

Senén Barro Ameneiro (A'02) was born in A Coruña, Spain, in 1962. He received the B.S. and Ph.D. degrees (with honors) in physics from the University of Santiago de Compostela, Santiago de Compostela, Spain, in 1985 and 1988, respectively.

Currently, he is a Professor of Computer Science and Rector of the University of Santiago de Compostela, Santiago de Compostela, Spain. Before joining the university in 1989, he was an Associate Professor at the Faculty of Informatics, University of A Coruña, A Coruña, Spain. His research focuses on signal and knowledge processing, mainly fuzzy systems, mobile robotics, and artificial neural networks (applications and biological modeling). He is the editor of five books and author of over 200 scientific papers in these fields.