Ellipsoid ART and ARTMAP for Incremental Clustering and Classification

Georgios C. Anagnostopoulos[1], Michael Georgiopoulos[2]

University of Central Florida
School of Electrical Engineering & Computer Science
4000 Central Florida Blvd.
Engineering Building, Room 407
Orlando, Florida United States 32816
Orlando, Florida, US
[1]anagnostop@email.com, [2]michaelg@mail.ucf.edu

Abstract

We introduce Ellipsoid-ART (EA) and Ellipsoid-ARTMAP (EAM) as a generalization of Hyper-sphere ART (HA) and Hypersphere-ARTMAP (HAM) respectively. As was the case with HA/HAM, these novel architectures are based on ideas rooted in Fuzzy-ART (FA) and Fuzzy-ARTMAP (FAM). While FA/FAM aggregate input data using hyper-rectangles, EA/EAM utilize hyper-ellipsoids for the same purpose. Due to their learning rules, EA and EAM share virtually all properties and characteristics of their FA/FAM counterparts. Preliminary experimentation implies that EA and EAM are to be viewed as good alternatives to FA and FAM for data clustering and classification tasks respectively.

1 Introduction

Fuzzy-ART (FA) [1] and Fuzzy-ARTMAP (FAM) [2] are two neural network architectures based on the *adaptive resonance theory* that addresses Grossberg's *stability-plasticity dilemma* [3]. While FA can be used for clustering of data, FAM is capable of forming associations between clusters of two different spaces and, as a special case, can be used as a classifier too. In this text, FAM will refer to the FAM classifier network. An important feature of FA/FAM is the ability to undergo both batch (off-line) and incremental (on-line) learning. In off-line learning, a set of training patterns is repeatedly presented until the termination of the networks' training phase. On the other hand, during on-line learning, the networks' structure is being altered as necessary to explain the existence of new patterns as they become available. Under *fast learning* ([1], [2]), a particularly interesting property of these networks is that they complete their learning in a finite number of steps, meaning that all training patterns will have been perfectly learned after a finite number of *list presentations* (epochs). This is in contrast, for example, to feed-forward neural networks, which use the Backprop algorithm for their training and only asymptotically reach a stable state.

In order to perform their learning task (clustering for FA and classification for the FAM classifier), both architectures group their input data into clusters (*FA categories* or simply *categories*). FA forms its categories from unlabeled input patterns via an unsupervised learning scheme, while in FAM categories are formed in a supervised manner and consist of input patterns bearing the same class label. Note that, in FAM many categories might describe a single class of patterns and therefore share a common class label. FA and FAM can be thought of as networks that during training perform compression of their inputs by substituting single patterns with clusters. The forming of clusters is achieved via a self-organizing scheme; FA/FAM perform their tasks without optimizing a specific objective function. Both of them process real-valued, vector-valued data; both cannot cope with data featuring missing attribute values. Also, FA and FAM work especially well, when the data is binary-valued. Furthermore, both of the networks require a preprocessing stage, where either input pattern normalization or complement coding is used to prevent category proliferation. While input data normalization causes a loss of vector-length information, complement coding normalizes input vectors and preserves their amplitude information. Another interesting aspect of the two architectures is that due to the internal structure, it is easy to explain the networks' outputs, such as why a particular pattern was selected by a category. This is not the case, for example, with feed-forward neural architectures,

0-7803-7044-9/01/$10.00 ©2001 IEEE

where it is significantly difficult to explain why an input **x** generates a certain output **y**. A key element in the learning process of the two architectures is a atypical pattern detection mechanism that is capable of identifying patterns, whose presence is not explained by the already developed (via learning) categories within the networks. Thus, during training phase, a pattern that does not fit the characteristics of already existing categories will initiate the creation of a new category. Aside from FA/FAM's advantages, criticism [4] has been voiced on the lack of a smoothing operation during learning that would ameliorate the effects of noise present in the data. The architectures use hard-competitive learning to form their categories; hence, over-fitting becomes an issue. The interested reader might find properties of learning for FA and FAM in [5] and [6]. We assume that the reader already possesses a rudimentary background in FA and FAM.

FA and FAM utilize hyper-rectangles for category representation, which works especially well for patterns, whose attributes take quantized values (for example, patterns with binary valued-features). Note, that if *M* is the dimensionality of the input space, then each FA category requires *2M* memory units (floating point numbers, for example). When it comes to clustering problems, depending on the distribution of input space patterns, hyper-rectangles are not always the ideal shape to represent clusters [7]. Furthermore, regarding classification tasks, due to the fact that both algorithms utilize city-block (L_1) distances, it can be shown that decision boundaries created by FAM are piece-wise linear.

Based on the aforementioned facts and FA/FAM's sensitivity to noise, *Gaussian-ART* (GA) and *Gaussian-ARTMAP* (GAM) were introduced in [7]. Although GA/GAM's categories do not have a geometric representation in the same fashion as FA/FAM categories do, they still correspond to hyper-ellipsoidal regions embedded in the input space, which signify the set of patterns that constantly update the related category. These regions can also be thought of summarizing the data they include in some loose sense and they accomplish to form non-linear (in general) decision boundaries. Note, that each category in GA/GAM utilizes *2M* memory units as categories in FA/FAM do. Despite several similarities between GA/GAM and FA/FAM, the former ones do not feature an appealing property of the latter ones, which is to complete the off-line training phase in a finite number of list presentations under fast learning conditions (learning rate γ equals 1). In other words, there is no fast learning law for GA/GAM. *Hypersphere-ART* (HA) [8] and *Hypersphere-ARTMAP* (HAM) [8] were the first neural network architectures to employ shapes (hyper-spheres) other than hyper-rectangles for category description, while maintaining the major (if not all) properties of FA and FAM. In HA/HAM, hyper-spheres require *M+1* memory units per category and are also capable of forming more complex decision boundaries than FA/FAM.

2 Ellipsoid-ART & Ellipsoid-ARTMAP

In this paper we present EA and EAM, which are a successful attempt of using hyper-ellipsoids as category representations with *2M+1* memory units per category, while simultaneously retaining virtually all of the properties and characteristics of FA and FAM respectively. The new architectures essentially extend the ideas first presented in HA/HAM.

To guarantee the inheritance of FA/FAM properties and to avoid over-parameterization of the resulting architectures, during the training phase EA categories are updated in such a manner so that they comply to the following two constraints: i) the hyper-ellipsoids maintain a constant ratio μ between the lengths of their major axis and their remaining minor axes; minor axes are of equal length. ii) The hyper-ellipsoids also maintain constant the direction of their major axis once it is set. Despite the above limitations, EA categories can have arbitrary orientations in the input space in order to capture the characteristics of the data. An EA category *j* corresponds to each committed node in the EA module's F_2 layer and is described by its *template* vector $\mathbf{w}_j=[\mathbf{m}_j\ \mathbf{d}_j\ R_j]$, where $\mathbf{m}_j$ is the *center* of the hyper-ellipsoid, $\mathbf{d}_j$ is called the category's *direction vector*, which coincides with the direction of the hyper-ellipsoid's major axis, and R_j is called the category's *radius*, which equals half of the major axis' length. Uncommitted nodes in the F_2 layer do not correspond to any category, since they represent the "blank memory" of the system, and their template vector $\mathbf{w}_j$ is undefined. A category's size $s(\mathbf{w}_j)$ is defined as the full length of the major axis and equals $2R_j$. A comparison between a 2-dimensional FA and EA category is given in Figure 1.

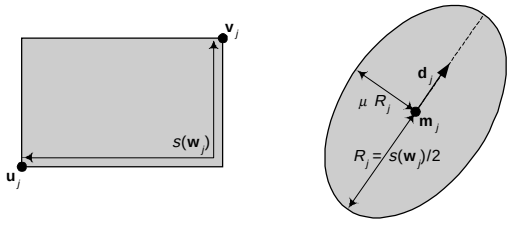

Figure 1: 2-dimensional FA and EA categories with template elements.

EA/EAM geometry revolves around the use of weighted Euclidian distances, rather than the city-block (L_1) distance of FA/FAM. Distances of patterns from an EA category *j* depend on the category's shape matrix $\mathbf{C}_j$ defined as

$$\mathbf{C}_j = \mathbf{C}_j^T = \frac{1}{\mu^2}\left[\mathbf{I} - (1-\mu^2)\mathbf{d}_j\mathbf{d}_j{}^T\right] \quad (1)$$

In the above expression, μ is the ratio of the hyper-ellipsoid's major axis length over the length of each other minor axis. Also, all vectors are arranged in columns and the *T*-exponent signifies the transpose of the quantity it is applied upon. Note, that for EA categories that encode a single pattern, $\mathbf{d}_j$ is defined to be equal to the zero vector $\mathbf{0}$. In general, the distance of an input pattern $\mathbf{x}$ from an EA category *j* is given as

$$dis(\mathbf{x} \mid \mathbf{w}_j) = \max\left\{\left\|\mathbf{x}-\mathbf{m}_j\right\|_{\mathbf{C}_j}, R_j\right\} - R_j \quad (2)$$

where we define

$$\left\|\mathbf{x}-\mathbf{m}_j\right\|_{\mathbf{C}_j} = \sqrt{(\mathbf{x}-\mathbf{m}_j)^T\,\mathbf{C}_j\,(\mathbf{x}-\mathbf{m}_j)} \quad (3)$$

The quantity displayed in Equation 3 is the *Mahalanobis distance* of the pattern from the category's center. Instead of explicitly forming the shape matrix $\mathbf{C}_j$, using Equation 1 we can calculate the distance of a pattern $\mathbf{x}$ from the center $\mathbf{m}_j$ of a category *j* with shape matrix $\mathbf{C}_j$ according to

$$\left\|\mathbf{x}-\mathbf{m}_j\right\|_{\mathbf{C}_j} = \frac{1}{\mu}\sqrt{\left\|\mathbf{x}-\mathbf{m}_j\right\|_2^2 - (1-\mu^2)\left[\mathbf{d}_j{}^T(\mathbf{x}-\mathbf{m}_j)\right]^2} \quad (4)$$

In Equation 4, $\|\cdot\|_2$ denotes the usual Euclidian (L_2) norm of its vector argument. Based on Equation 2, we define the *representation region* of an EA category *j* as the set of all points of the input space satisfying

$$dis(\mathbf{x} \mid \mathbf{w}_j) = 0 \quad \Rightarrow \quad \left\|\mathbf{x}-\mathbf{m}_j\right\|_{\mathbf{C}_j} \le R_j \quad (5)$$

In Figure 1 the shaded areas correspond to the representation regions of the categories involved. The shape of the representation regions in EA/EAM depends on the order, according to which patterns are being presented during training, and the EA/EAM network parameters. Apart from the vigilance parameter $\rho \in [0,1]$ the choice parameter $a>0$ and the learning rate $\gamma \in (0,1]$, EA/EAM feature 3 additional network parameters. The first one is $D>0$ and assumes the role of *M* (input domain dimensionality) used in FA/FAM. Another one is the ratio of lengths $\mu \in (0,1]$ of a category's minor axis with respect to its major axis. The last one is $\omega \ge 1$, which corresponds to the parameter $w_u \ge 1$ of FA/FAM. As a reminder, templates of uncommitted nodes in FA/FAM are initialized to $\mathbf{w}=w_u\mathbf{1}$, where $\mathbf{1}$ is the all-ones vector. In the case of the EAM classifier, all these parameters will refer to the ones of its ART_a module, since the ones of ART_b are of no practical interest. The ART_a module corresponds to the input domain and ART_b to the output domain of EAM. When EAM is used as a classifier the output domain coincides with the set of class labels pertinent to the classification problem.

As was the case with HA/HAM [8], the definition of the *category match function* (CMF) $\rho(\mathbf{w}_j|\mathbf{x})$ and the *category choice function* (CCF) $T(\mathbf{w}_j|\mathbf{x})$ (also known as *bottom-up input* or *activation function*) for a committed node *j* is based on the homologous expressions valid for FA categories:

$$\rho(\mathbf{w}_j \mid \mathbf{x}) = \frac{D - s(\mathbf{w}_j) - dis(\mathbf{x},\mathbf{w}_j)}{D} \overset{\text{Eq. 2}}{\Rightarrow} \rho(\mathbf{w}_j \mid \mathbf{x}) = 1 - \frac{R_j + \max\left\{R_j, \left\|\mathbf{x}-\mathbf{m}_j\right\|_{\mathbf{C}_j}\right\}}{D} \quad (6)$$

$$T(\mathbf{w}_j \mid \mathbf{x}) = \frac{D - s(\mathbf{w}_j) - dis(\mathbf{x},\mathbf{w}_j)}{D - s(\mathbf{w}_j) + a} \overset{\text{Eq. 2}}{\Rightarrow} T(\mathbf{w}_j \mid \mathbf{x}) = \frac{D - R_j - \max\left\{R_j, \left\|\mathbf{x}-\mathbf{m}_j\right\|_{\mathbf{C}_j}\right\}}{D - 2R_j + a} \quad (7)$$

Uncommitted nodes in EA/EAM feature a constant CMF and CCF value for all patterns of the input space as shown below

$$\rho(\mathbf{x} \mid \mathbf{w}_j) = 1 \quad (8)$$

$$T(\mathbf{x} \mid \mathbf{w}_j) = \frac{D}{2D\omega + a} \quad (9)$$

As a reminder, the CMF value of a category is used for its comparison to the vigilance parameter, when performing the *vigilance test* (VT). Also, CCF values are utilized to determine a winning (committed or uncommitted) node during node competition in the F_2 layer for an input pattern. To guarantee that the CMF and the CCF values are non-negative for all categories and for all possible input patterns, *D* should be selected at least equal to the maximum possible Euclidian distance between patterns of the input space (*input space Euclidian diameter*) in consideration divided by the ratio parameter in use, that is,

$$D \ge \frac{1}{\mu}\max_{p,q}\left\|\mathbf{x}_p - \mathbf{x}_q\right\|_2 \quad (10)$$

From Equations 6 through 9 it can be shown that using a value of $D=D_1>0$ and a value of $a=a_1>0$ during the training phase of EA/EAM is equivalent to using $D=D_2>0$,

$a=a_2=a_1D_2/D_1$ and to scaling (multiplying) all input data by a factor of D_2/D_1.

EA/EAM training and phases are identical to the ones of FA/FAM. Upon presentation of a new pattern all nodes in the F_2 layer compete in terms of CCF values. The node of maximum CCF value is declared winner and its CMF value is compared to the value of ρ (VT is being performed). The pattern chooses this node, if the VT is passed. Otherwise, the winning node is being reset and is excluded from future node competitions until the next input pattern is presented; after the reset, the node competition will be repeated. During training, if the winning node that passes the VT is uncommitted, the node becomes committed through appropriate initialization. Otherwise, if the node is already committed, then it might get appropriately updated; if the pattern falls inside the node's representation region, no node update takes place. Moreover, in the case of EAM, if the chosen node corresponds to a different class label than the just-presented pattern, *match tracking* (MT) [2] goes into effect.

EA categories encode training patterns by updating their templates. During training, EA categories can only grow in size and therefore can never be destroyed. The learning rules of EA/EAM resemble the ones of HA/HAM and are depicted below. The special case of $\gamma=1$ corresponds to *fast learning*. Also, $\mathbf{x}_{(2)}$ denotes the second pattern to be encoded in category j.

$$R_j^{new} = R_j^{old} + \frac{\gamma}{2}\left(\max\left\{ R_j^{old}, \left\| \mathbf{x} - \mathbf{m}_j^{old} \right\|_{C_j^{old}} \right\} - R_j^{old} \right) \quad (11)$$

$$\mathbf{m}_j^{new} = \mathbf{m}_j^{old} + \frac{\gamma}{2}\left(1 - \frac{\min\left\{ R_j^{old}, \left\| \mathbf{x} - \mathbf{m}_j^{old} \right\|_{C_j^{old}} \right\}}{\left\| \mathbf{x} - \mathbf{m}_j^{old} \right\|_{C_j^{old}}} \right) \left(\mathbf{x} - \mathbf{m}_j^{old} \right) \quad (12)$$

$$\mathbf{d}_j = \frac{\mathbf{x}_{(2)} - \mathbf{m}_j}{\left\| \mathbf{x}_{(2)} - \mathbf{m}_j \right\|_2} \qquad \mathbf{x}_{(2)} \neq \mathbf{m}_j \quad (13)$$

Equation 11 and 12 imply that, when a training pattern is already located inside the representation region of category j, no updates will take place for this category. Figure 2 provides a 2-dimensional illustration of a comparison between FA and EA category updates. Assuming that a category already encodes a minimum of 2 patterns, due to the learning rules in Equations 11 through 13, the category's new representation region after an update can be shown to be the minimum hyper-volume hyper-ellipsoid that simultaneously contains both the old representation region and the new pattern to be encoded. Notice, that once the EA category's direction vector $\mathbf{d}_j$ has been set, it remains constant during future updates. Notice, also that the boundaries of the two ellipsoids E^{old} and E^{new} touch only at one point, $\mathbf{t}(\mathbf{x},\mathbf{w}_j^{old})$.

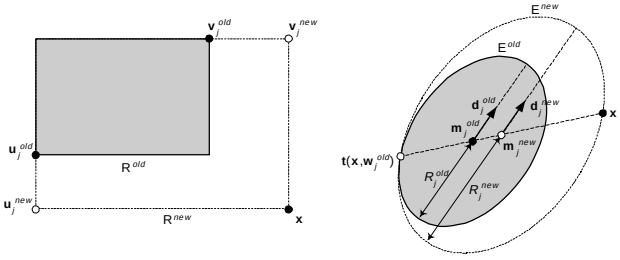

Figure 2: Category update assuming fast learning in FA/FAM and EA/EAM in 2 dimensions.

A typical lifecycle of an EA category under fast learning assumptions is depicted in Figure 3. When category j is first created upon presentation of pattern $\mathbf{x}_1$, its center $\mathbf{m}_j$ coincides with $\mathbf{x}_1$, its direction vector is $\mathbf{d}_j=\mathbf{0}$ and its radius is $R_j=0$. Assuming that category j is eligible to encode pattern $\mathbf{x}_2$, the category's representation region expands into an ellipse with its center $\mathbf{m}_j$ amidst $\mathbf{x}_1$ and $\mathbf{x}_2$, R_j equal to the Euclidian distance between $\mathbf{x}_1$ and $\mathbf{x}_2$ and, finally, $\mathbf{d}_j$ is set equal to the unit Euclidian-length vector along the direction of $\mathbf{x}_2$-$\mathbf{x}_1$. Assuming that category j is also eligible to encode a third pattern $\mathbf{x}_3$, the representation region expands enough to include the previous representation region and the new pattern, while maintaining constant its relative shape (constant ratio μ of minor axis length over major axis length) and constant direction (constant direction vector $\mathbf{d}_j$).

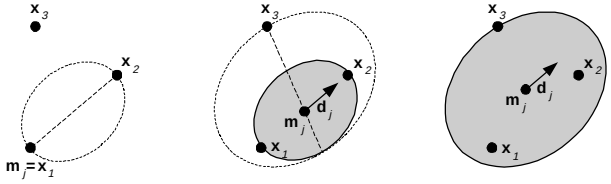

Figure 3: Lifecycle of an EA category in 2 dimensions under fast learning.

The performance phases of EA/EAM are also identical to the ones of FA/FAM. To force EAM to classify a test pattern to one of the existing classes, the vigilance ρ has to be set equal to 0 and $\omega\rightarrow\infty$, so that uncommitted nodes are excluded from node competition. An interesting fact regarding EAM is that, if it is trained using $\rho=1$ and then is being used as a classifier with $\rho=0$ and $\omega\rightarrow\infty$ during

performance, EAM becomes equivalent to the L_2-norm (Euclidian) 1-Nearest Neighbor classifier [9]. Furthermore, if $\mu=1$ EA/EAM become equivalent to HA/HAM, which justifies our statement, that EA/EAM are generalizations of HA/HAM.

3 Preliminary experimental results

In general, an objective comparison of clustering algorithms is difficult to be achieved, thus, we will not attempt to directly compare EA with FA. However, we can show the potential of the EA/EAM family by comparing EAM with FAM on the basis of classification performance. We chose a simple, artificial classification example for our purposes, namely the Circle-in-a-Square problem. It has been used in the past as a benchmark problem in the DARPA artificial neural network technology (ANNT) program [10]. A circle of radius $R = 1/\sqrt{2\pi}$ is inscribed in the unit square, so that its interior covers exactly half of the unit square's surface. The classifiers have to learn how to distinguish points inside the circle (class label 1) from points outside it (class label 0). Figures 4 and 5 display typical decision regions of FAM and EAM for the problem at hand. Black areas correspond to points that the classifier labels as class 0, white ones correspond to class 1 and finally gray ones to points that the classifiers were unable to label. These figures also illustrate training patterns as stars and categories created via training. For our experiments, ten training sets were created by drawing samples from each class with equal probability. Each set contained a multiple of 10 worth of training patterns: the first set contains 10, the second 20, etc. Also, a test set was created with 10,000 equally spaced labeled patterns that formed a grid of points inside the unit square.

Figure 4: An example of decision regions of FAM for the Circle-in-a-Square problem.

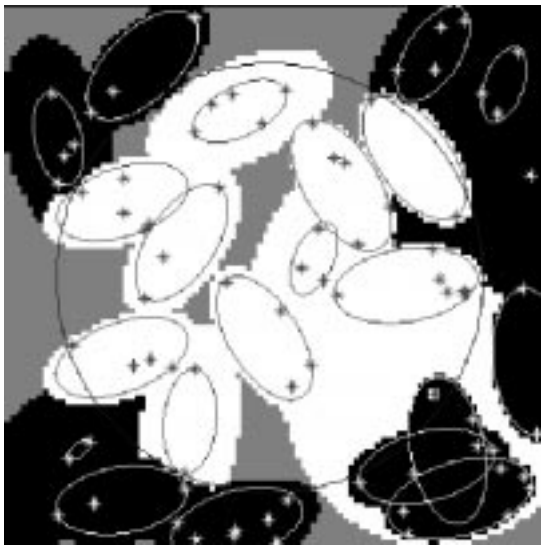

Figure 5: An example of decision regions of EAM for the Circle-in-a-Square problem.

Along with FAM and EAM another 11 classifiers were considered: L_1-norm, L_2-norm, L_∞-norm k-Nearest Neighbor (KNN) [10], a variant of the Restricted Coulomb Energy (RCE) classifier [11] using L_1-norm, L_2-norm and L_∞-norm, Parzen classifiers [12] with Gaussian, sinc-squared, Laplacian and Cauchy kernels and, finally, GAM. All classifiers were exposed to the same order of training patterns and were trained on a variety of parameter settings; except the KNN and Parzen classifier, the rest of them were trained with approximately 1,000 distinct parameter settings. Moreover, FAM, EAM and RCE classifiers were trained until completion of training (FAM and EAM used fast learning). Finally, GAM's training phase was terminated every time, when, after two consecutive epochs, no new categories were created.

Table 1: Classification results showing best and second best classifier along with their percent misclassification (PCM) on the test set and the number of exemplars (categories) used to achieve this performance.

Training set size	*Best Classifier (PMC% - categories)*	*Second Best Classifier (PMC% - categories)*
10	FAM (25.00% - 7)	GAM (25.87% - 5)
20	EAM (16.93% - 15)	RCE_2(19.88% - 8)
30	EAM (11.77% - 27)	RCE_1(12.43% – 22)
40	EAM (6.28% - 5)	RCE_∞(7.52% - 25)
50	FAM (7.76% - 54)	EAM (7.84% - 44)
60	EAM (8.25% - 10)	FAM (9.47% - 8)
70	EAM (7.43% - 56)	$Parzen_c$ (7.93% - 70)
80	RCE_2 (6.17% - 33)	GAM (6.81% – 37)
90	EAM (7.42% - 41)	FAM (7.99% - 30)
100	EAM (5.48% - 6)	RCE_1 (5.67% - 42)

The classification performance results are illustrated in Table 1. We notice that for 7 out of 10 training sets the best network was an EAM classifier. However, with training sets of 70 patterns and above the differences in percent misclassification are very small between the best and second best classifier. Notice that, depending on the number and distribution of training patterns, ellipsoids might be more efficient for category descriptions than other shapes and EAM achieves good classification performance, while utilizing only a few categories (see cases with 40 and 100 training patterns). KNN and Parzen classifiers did not perform that well in our experiments because of the small sizes of training sets; these classifiers typically exhibit improved misclassification rates, when there is an abundance of reference (training) patterns. If you find the ideas behind EA/EAM interesting and would like to further study these architectures, you may contact the primary author for MATLAB® binaries implementing the EAM classifier.

4 Conclusions

We have introduced Ellipsoid ART and Ellipsoid ARTMAP for clustering and classification tasks respectively. While Fuzzy ART and Fuzzy ARTMAP represent categories with hyper-rectangles embedded in the input domain and make use of the L_1-norm for size and distance calculations, their Ellipsoid counterparts utilize hyper-ellipsoids and weighted L_2-norm for the same purposes. Due to Ellipsoid ART's and ARTMAP's category and learning law definition, both families of networks, Fuzzy- and Ellipsoid-, share virtually all properties of learning. While in some learning problems hyper-rectangles are ideal for data summarization, in other ones hyper-ellipsoids seem to be more appropriate. Preliminary experimental results shown in this paper indicate that, on occasion, Ellipsoid ART and ARTMAP may be regarded as good alternatives to Fuzzy ART and ARTMAP for clustering and classification problems.

5 References

[1] G.A. Carpenter, S. Grossberg and D.B. Rosen, "Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system", *Neural Networks*, 4(6), 1991, pp. 759-771.

[2] G.A. Carpenter, S. Grossberg, N. Markuzon, J.H. Reynolds and D.B. Rosen, "Fuzzy ARTMAP: A Neural Network Architecture for Incremental Supervised Learning of Analog Multidimensional Maps", *IEEE Transaction on Neural Networks*, 3(5), 1992, pp. 698-713.

[3] S. Grossberg, "Adaptive pattern recognition and universal encoding II: Feedback, expectation, olfaction, and illusions", *Biological Cybernetics*, 23, 1976, pp. 187-202.

[4] A. Baraldi and P. Blonda, "A Survey of Fuzzy Clustering Algorithms for Pattern Recognition – Part II", IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics, 29(6), December 1999, pp. 786-801.

[5] J. Huang, M. Georgiopoulos and G.L. Heileman, "Fuzzy ART Properties", Neural Networks, 8(2), pp. 203-213, 1995.

[6] M. Georgiopoulos, H. Fernlund, G. Bebis and G.L. Heileman, "Order of Search in Fuzzy ART and Fuzzy ARTMAP: Effect of the choice parameter", Neural Networks, 9(9), 1996, pp. 1541-1559.

[7] J.R. Williamson, "Gaussian ARTMAP: A Neural Network for Fast Incremental Learning of Noisy Multidimensional Maps", *Neural Networks*, **9**(5), 1996, pp. 881-897.

[8] G.C. Anagnostopoulos and M. Georgiopoulos, "Hypersphere ART and ARTMAP for Unsupervised and Supervised, Incremental Learning", *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks*, 6, Como, Italy, 2000, pp. 59-64.

[9] G. Wilensky, "Analysis of neural network issues: Scaling, enhanced nodal processing, comparison with standard classification", DARPA Neural Network Program Review, October 29-30, 1990.

[10] R.O. Duda and P.E. Hart, "Pattern classification and scene analysis", Wiley, New York, New York, 1973.

[11] M.H. Hassoun, "Fundamentals of Artificial Neural Networks", MIT Press, Cambridge, Massachusetts, 1995.

[12] K. Fukunaga, "Introduction to Statistical Pattern Recognition", Academic Press, New York, New York, 1972.