COMPARISONS OF GAUSSIAN ARTMAP AND DISTRIBUTED GAUSSIAN ARTMAP CLASSIFIERS: THE CATEGORY PROLIFERATION PROBLEM

ANNA KOUFAKOU
School of EECS
University of Central Florida

NICK WEIHS
School of EECS
University of Central Florida

MICHAEL GEORGIOPOULOS
School of EECS
University of Central Florida

AHMAD AL-DARAISEH
School of EECS
University of Central Florida

ABSTRACT

Adaptive Resonance Theory (ART) neural networks are a popular class of neural network classifiers, and they are based on the adaptive resonance theory, developed by Grossberg. ART neural networks have a number of desirable features, such as guaranteed convergence to a solution, on-line learning capabilities, identification of novel inputs, offering an explanation for the answers that they produce, and finally, achieving good performance on a number of classification problems in a variety of application areas. Two members of the class of ART classifiers that have been introduced into the literature are Gaussian ARTMAP (GAM) and Distributed Gaussian ARTMAP (dGAM). The difference between dGAM and GAM is that in its learning phase, dGAM allows more than one ART node to learn the input pattern, contrary to GAM which allows only one ART node to learn the input pattern (winner-take-all ART network. The inventors of dGAM claimed that dGAM addresses the category proliferation problem, observed by many winner-take-all ART networks, such as Gaussian ARTMAP, Fuzzy ARTMAP, Ellipsoidal ARTMAP, amongst others. The category proliferation problem is the problem where an ART network, in the process of learning the required classification task, creates more than necessary ART nodes. This category proliferation problem is more acute when the ART networks are faced with noisy and or significantly overlapping data. However the claim, that dGAM outperforms GAM by creating smaller ART networks, has not been substantiated in the literature. In this paper, a thorough experimentation and comparison of the performance of Gaussian ARTMAP (GAM) and distributed Gaussian ARTMAP (dGAM) is provided. In the process of doing so, a new measure of performance for a neural network is introduced. This measure relies on two factors of goodness of the neural network: the network's size, and the network's generalization performance (i.e., performance of the trained ART classifier on unseen data). Obviously, a small size ART network of good generalization performance is desired. Previous comparisons of ART-like classifiers relied on a trial and error procedure (that is a time consuming and occasionally unreliable procedure) to produce a good performing ART network. The proposed measure of performance allows one to come up with a good ART network through an automated and reliable process.

INTRODUCTION

The Adaptive Resonance Theory (ART) was developed by Grossberg (1976). One of the most celebrated ART architectures is Fuzzy ARTMAP (Carpenter et al, 1992), which has been successfully used in the literature for solving a variety of classification problems. Some of the advantages that Fuzzy ARTMAP possesses is that it can solve arbitrarily complex classification problems, it converges quickly to a solution (within a few presentations of the list of the input/output patterns belonging to the training set), it has the ability to recognize novelty in the input patterns presented to it, it can operate in an on-line fashion (new input/output patterns can be learned by the system without re-training with the old input/output patterns), and it produces answers that can be explained with relative ease. One of the limitations of Fuzzy ARTMAP that has been extensively reported in the literature is the category proliferation problem. That is Fuzzy ARTMAP has the tendency of increasing its network size, as it is confronted with more and more data, especially if the data are noisy and/or contain a lot of overlap. This limitation of Fuzzy ARTMAP has been observed by other ART architectures, introduced later on in the ART literature, such as Ellipsoidal ARTMAP (Anagnostopoulos et al., 2001) and Gaussian ARTMAP (Williamson, 1996). The major difference between Fuzzy ARTMAP, Ellipsoidal ARTMAP and Gaussian ARTMAP is the way that these architectures choose to compress the input data. Fuzzy ARTMAP compresses the data by representing them through their minimum and maximum values across every dimension, resulting in a boxed structure that contains all the data coded by an ART node. Ellipsoidal ARTMAP compresses the data by representing them by the mean, direction of the major axis and ratio of the major to minor axis length of an ellipsoidal structure that contains all the data coded by an ART node. Finally, Gaussian ARTMAP compresses the data by representing them as the mean and variances of Gaussian curves corresponding to each ART node; the means and variances of these Gaussian curves are computed using the data that an ART node has encoded.

A number of authors have tried to address the category proliferation problem in ART. Amongst them we refer to the work by Marriott and Harrisson (1995), where the authors eliminate the match tracking mechanism of Fuzzy ARTMAP when dealing with noisy data, the work by Charalampidis, et al. (2001), where the ART equations are appropriately modified to compensate for noisy data, the work by Verzi, et al. (2001), Anagnostopoulos, et al. (2001, 2003), and Gomez-Sanchez, et al. (2001, 2002), where different ways are introduced of allowing the ART categories to encode patterns that are not necessarily mapped to the same label, the work by Koufakou, et al., (2001), where cross-validation is employed to avoid the category proliferation problem in ART, and the work by Carpenter (1998), Williamson (1997), Parrado-Hernandez, et al. (2003), where the ART structure is changed from a winner-take-all to a distributed version and simultaneously slow learning is employed with the intent of creating fewer ART categories and reducing the effects of noisy patterns.

In this paper we focus our attention on comparing two ART networks, Gaussian ARTMAP and distributed Gaussian ARTMAP. The purpose of this comparison is to determine whether distributed learning is indeed helping us in solving the category proliferation problem in ART. Despite the fact that such a

comparison of winner-take-all and distributed learning ART networks has been conducted before (see Carpenter, 1998; Parrado-Hernandez, 2003), the distributed ART network introduced in (Carpenter, 1998) was complex and the benefits from introducing it, in terms of reducing the number of categories created by ART networks, was not very clear from the results. On the contrary, distributed Gaussian ARTMAP is a natural distributed extension of the winner-take-all Gaussian ARTMAP and if experimentation proves that it improves the category proliferation problem observed by Gaussian ARTMAP then this distributed network can be extended to other ART networks, with similarly beneficial conclusions. Furthermore, in the process of comparing two ART networks (i.e., GAM and dGAM) we have introduced an appropriate fitness function that takes into consideration both the network size and the network's generalization performance. This fitness function, whose usefulness is demonstrated in the paper, is needed because otherwise the comparison of ART networks because cumbersome, and error prone.

GAUSSIAN ARTMAP ARCHITECTURES

In this section, we briefly explain the Gaussian ARTMAP (GAM) and distributed GAM (dGAM) architectures. For more details about these architectures the reader is referred to the papers by Williamson (Williamson, 1996, 1997).

The block diagram of a Gaussian ARTMAP (GAM or dGAM) architecture is shown in Figure 1. The Gaussian ARTMAP architecture, depicted in Figure 1, has three major layers. The *input layer* (F_1^a) where the input patterns (designated by I) are presented, the *category representation layer* (F_2^a), where compressed representations of these input patterns are formed, and the *output layer* (F_2^b) that holds the labels of the categories formed in the category representation layer.

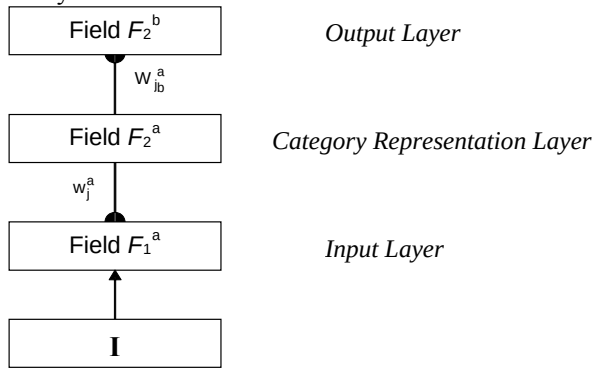

Figure 1: Block Diagram of Gaussian ARTMAP

GAM can operate in two distinct phases: the training phase and the performance (test) phase. The training phase of GAM can be described as follows: Given a set of inputs and associated label pairs,

$$\{(\mathbf{I}^1, label(\mathbf{I}^1)), ..., (\mathbf{I}^r, label(\mathbf{I}^r)), ..., (\mathbf{I}^{PT}, label(\mathbf{I}^{PT}))\}$$

called the *training set*), we want to train Gaussian ARTMAP to map every input pattern of the training set to its corresponding label. To achieve the aforementioned goal we present the training set to the GAM architecture repeatedly. That is, we present $\mathbf{I}^1$ to F_1^a, $label(\mathbf{I}^1)$ to F_2^b, $\mathbf{I}^2$ to F_1^a, $label(\mathbf{I}^2)$ to F_2^b, and finally, $\mathbf{I}^{PT}$ to F_1^a, $label(\mathbf{I}^{PT})$ to F_2^b. We present the training set to GAM up to designated number of times. After the training phase of Gaussian ARTMAP is completed we can enter the performance phase of GAM, which works as follows: Given a set of input patterns $\tilde{\mathbf{I}}^1, \tilde{\mathbf{I}}^2, ..., \tilde{\mathbf{I}}^{PS}$ (referred to as the *test set*), we want to find the Gaussian ARTMAP output (label) produced when each one of the aforementioned test patterns is presented at its F_1^a layer. In order to achieve this goal, we present the test set to the trained Gaussian ARTMAP architecture and we observe the network's output.

What is worth emphasizing is that Gaussian ARTMAP creates compressed representations of the input patterns that chose node j in the category representation layer as their representative node. Actually, the weight values corresponding to a node j in F_2^a are: $\boldsymbol{\mu}_j$ (mean of the data that have activated and were encoded by node j), $\boldsymbol{\sigma}_j$ (the standard deviation vector of the data that have activated and were encoded by node j), n_j (the number of training input patterns that were encoded by node j in F_2^a), and the inter-ART weights $\mathbf{W}_j^{ab}$ (with components W_{jk}^{ab}; $j = 1,..., N_a$, $k = 1,..., N_b$). For conciseness, the GAM architecture of Figure 1 uses the vector $\mathbf{w}_j^a = (\boldsymbol{\mu}_j, \boldsymbol{\sigma}_j, n_j)$ to represent all three weight values $\boldsymbol{\mu}_j, \boldsymbol{\sigma}_j, n_j$, at once. The compression of the data at every representation layer node happens because GAM chooses to represent the data that are coded by a single representation layer node by their mean vector, and their standard deviation across every dimension. Furthermore, GAM keeps track of another parameter (n_j) that corresponds to the number of data-points that chose and were coded by this node, during GAM's training phase. Finally, the vector $\mathbf{W}_j^{ab}$ corresponding to a committed representation layer node has one of its components equal to 1 (representing the label that this committed node is mapped to) and the other components equal to zero.

GAM's performance depends on two parameters. One of them, designated as γ is the initial value of the standard deviation (across every dimension) of a representation layer node, the first time that this node becomes committed and encodes an input pattern. In most Gaussian ARTMAP simulations this parameter is chosen to be value within the interval (0, 1]. The other parameter, designated as $\bar{\rho}_a$, and referred to as the baseline vigilance parameter assumes also values in the interval [0, 1] and controls the amount of similarity that input patterns should have before they are allowed to be encoded by the same representation layer node. In particular, as the value of $\bar{\rho}_a$ increases from 0 to 1, the more similar two input patterns should be in order to be encoded by the same representation layer node in Gaussian ARTMAP.

The distributed Gaussian ARTMAP training differs from the Gaussian ARTMAP training in two distinct ways. For one, during the presentation of an input pattern to distributed Gaussian ARTMAP more than one representation layer nodes are activated, in contrast to Gaussian ARTMAP, where only one node (the one that receives the highest bottom-up input is activated). Also, if a group of nodes are activated during the presentation of an input pattern to distributed Gaussian ARTMAP and these nodes, collectively, do not predict the correct label, all the nodes are de-activated and a new set of nodes is searched, whose vigilance ratio exceeds a weighted average vigilance ratio of the nodes that were previously active. The vigilance ratio of a node is a measure of similarity between the node's compressed representation of previously coded inputs and the currently presented input pattern.

EXPERIMENTS

We have performed a number of experiments to compare the performance of Gaussian ARTMAP and distributed Gaussian ARTMAP. The performance comparison is based on two measures of performance, the generalization performance of the network, as well as the size of the network created. The performance comparison was based on a fitness function, discussed below. The comparisons were conducted on a number of artificial as well as real databases. For each one of these databases we had a training set (to train the Gaussian ARTMAP networks), a validation set (to optimize the Gaussian ARTMAP network parameters) and a test set on which the generalization performance of the optimal Gaussian ARTMAP networks was tested. The parameters that we experimented with, to optimize Gaussian ARTMAP's performance were: baseline vigilance ($\overline{\rho}_a$), initial value of the standard deviation (γ) and the order of training pattern presentation presented to GAM. More details about the specific parameter settings are emphasized in Section c.

a. Fitness Function

To achieve this goal we relied on a fitness function that was first introduced in (Al-Daraiseh, et al., 2006). This fitness function depends on a network's generalization performance, as well as on a network's size. The specific form of the fitness function is presented below:

$$fitness = \ln\left[\frac{(C_{\max} - N_a)PCC^{2}}{(\frac{100}{C_{\min}}) - (\frac{PCC}{N_a}) + 0.001}\right]$$

where $C_{\max}$ is the maximum number of categories that a Gaussian ARTMAP network can have (set equal to the number of patterns in the training set), and $C_{\min}$ is the minimum number of categories that a Gaussian ARTMAP network can have. The parameter $C_{\min}$ is chosen to be equal to 1. It seems that a more natural choice would have been to choose $C_{\min}$ equal to the number of different classes in the problem at hand, but our experiments have shown that the setting $C_{\min}$ gives us good results. The parameter *PCC* is the percentage of correct classification of a trained Gaussian ARTMAP network on the validation set, and

N_a is the actual number of categories of a trained Gaussian ARTMAP network. Finally, ε is a small positive number.

The chosen fitness function has a number of good properties. First, it depends on both measures of performance: the size of the Gaussian ARTMAP network and the accuracy of the Gaussian ARTMAP network on the validation set. Higher accuracy leads to larger fitness values, with all other factors fixed. Note that if the size decreases, the numerator of the fitness increases and the denominator decreases, provided that everything else is kept fixed. Similarly, if the accuracy increases, the numerator of the fitness increases and the denominator decreases provided that everything else is kept fixed. It is also worth noting that when the size is equal to the minimum size and the accuracy is equal to the highest accuracy, the denominator of the fitness function practically approaches zero and the fitness function assumes a very high value. Hence, the fitness function shows a strong preference towards the creation of minimum size and highest accuracy networks, as it should. It is worth noting that other values choices for the fitness function were examined (including the obvious one PCC/N_a), but none of them gave as good results as the fitness function defined above.

b. Databases

We experimented with both artificial and real databases. The specifics of these databases are given in Table 1.

1. Gaussian Databases (G#c-##)

These are artificial databases, where we created 2-dimensional data, Gaussianly distributed, belonging to 2-class, 4-class, and 6-class problems. In each one of these databases we varied the amount of overlap of data belonging to different classes. In particular, we considered 5%, 15%, 25%, and 40% overlap. Note that 5% overlap means the optimal Bayesian Classifier would have 5% misclassification rate on the Gaussianly distributed data. There are a total of 3×4=12 Gaussian databases. We name the databases as "G#c-##" where the first number is the number of classes and the second number is the class overlap. For example, G2c-05 means the Gaussian database is a 2 class and 5% overlap database.

2. Modified Iris Database (MOD-IRIS)

In this database we started from the IRIS dataset (Hettich et al., 1998) of the 150 3-class problem. We eliminated the data corresponding to the class that is linearly separable from the others. Thus we ended up with 100 data-points. From the 4 input attributes of this IRIS dataset we focused on only 2 attributes (attribute 3 and 4) because they seem to have enough discriminatory power to separate the 2-class data. Finally, in order to create a reasonable size dataset from these 100 points (so we can reliably perform cross-validation to identify the optimal Safe μARTMAP parameters) we created noisy data around each one of these 100 data-points (the noise was Gaussian of zero mean and small variance) to end up with approximately 10,000 points. We named this database Modified Iris.

3. Modified Abalone Database (ABALONE)

This database is originally used for prediction of the age of an abalone. contains 4177 instances, each with 7 numerical attributes, 1 categorical attribute, and 1 numerical target output (age). We discarded the categorical attribute in our experiments, and grouped the target output values into 3 classes: 8 and lower (class 1), 9-10 (class 2), 11 and greater (class 3). This grouping of output values has been reported in the literature before.

4. Page Blocks Database (PAGE)

This database represents the problem of classifying the blocks of the page layout in a document (Hettich et al., 1998). One of the noteworthy points about this database is that, its major class has a high probability of occurring (above 80%).

The data in each one of the above databases was split into a training set, a validation set, and a test set. The percentage of classes in each one of these subsets resembled the percentage of classes in the original dataset. The summarized specifics of each one of these databases are depicted in Table 1. The training set was used to train the Gaussian ARTMAP networks, the validation test was used to assess the performance of the ART networks for various settings of their parameter values, and the test set was used to report the performance of the "best performing" networks.

	Database Name	*# Training Instances*	*# Validation Instances*	*# Test Instances*	*# Numerical Attributes*	*# Classes* (N_b)	*% Major Class* (A_0)
1	G2c-05	500	5000	5000	2	2	1/2
2	G2c-15	500	5000	5000	2	2	1/2
3	G2c-25	500	5000	5000	2	2	1/2
4	G2c-40	500	5000	5000	2	2	1/2
5	G4c-05	500	5000	5000	2	4	1/4
6	G4c-15	500	5000	5000	2	4	1/4
7	G4c-25	500	5000	5000	2	4	1/4
8	G4c-40	500	5000	5000	2	4	1/4
9	G6c-05	504	5004	5004	2	6	1/6
10	G6c-15	504	5004	5004	2	6	1/6
11	G6c-25	504	5004	5004	2	6	1/6
12	G6c-40	504	5004	5004	2	6	1/6
13	MOD-IRIS	500	4800	4800	2	2	1/2
14	ABALONE	501	1838	1838	7	3	1/3
15	PAGE	500	2486	2487	10	5	0.832

Table 1: Specifics of the Databases

c. Network Parameter Settings

Experiments were conducted with 11 different values of the baseline vigilance parameter, $\overline{\rho}_a$, ranging from 0.0 to 1.0 in intervals of 0.1, and 10 different values of the initial standard deviation parameter, γ, ranging from 0.1 to 1.0 in intervals of 0.1. Additionally, for each value of $\overline{\rho}_a$ and γ, we used a 100 different orders of pattern

presentation for the training data, resulting in a total of 11,000 networks being created per database. Note that the performance of Gaussian ARTMAP networks is affected by the order according to which the training patterns are presented to Gaussian ARTMAP.

d. Experimental Procedure – Experimental Results

The best GAM, dGAM network was chosen to be the one that maximized the value of the fitness function, defined in Section a. That is, we trained each of GAM 11,000 times (corresponding to the 11,000 parameters used in Section c) and we identified as the optimal GAM network the one that maximized the value of the fitness function. We also trained dGAM 11,000 times (corresponding to the 11,000 parameters used in Section c) and we identified as the optimal dGAM network the one that maximized the value of the fitness function. The comparison of the performances of the optimal dGAM size (Nodes) and accuracy on the validation set (PCC XV), as well as accuracy (generalization) on the test set (PCC Test)) for each one of the databases are demonstrated in Table 2. Table 2 also contains the fitness function value of the optimal network. Figures 2 (a-d) show how well this fitness function works by showing the network size and generalization performance of the best 100 (more or less) GAM, and dGAM networks, as well as the specific network that this fitness function chooses as optimal.

The results from Table 2 show that both of the algorithms, GAM and dGAM, achieved near optimal performance with respect to the percentage correct classification for the Gaussian datasets. Furthermore, the size of the networks created by dGAM is smaller than the size of the network created by GAM in datasets with significant amount of overlap between data belonging to different classes (e.g., see size for G2c_40, G4c_40 and G6c_40). In particular, the size of the network is decreased by 50% for the G2c_40 problem, by 23% for the G4c_40 problem, and by 20% for the G6c_40 problem. These results seem to indicate that distributed learning (where more than one node is activated in the representation layer of Gaussian ARTMAP) seems to be having a beneficial influence on the category proliferation problem. This result is also reinforced by the significant reduction of the nodes created by

Set	*Nodes*		*PCC Test*		*PCC XV*		*Fitness*	
	GAM	dGAM	GAM	dGAM	GAM	dGAM	GAM	dGAM
G2c_05	5	5	95.14	95.16	94.7	94.78	2.19	2.19
G2c_15	6	6	84.8	84.8	85.1	85	1.98	1.97
G2c_25	7	6	74.94	75	74.78	75.1	1.72	1.73
G2c_40	20	9	59.7	61.14	59.08	59.76	1.23	1.26
G4c_05	11	8	94.7	94.86	95.5	95.58	2.20	2.21
G4c_15	22	17	84.1	84.3	84.02	83.92	1.93	1.94
G4c_25	27	25	74.7	74.68	75.44	75.46	1.71	1.71
G4c_40	43	33	59.02	59.46	60.54	60.88	1.26	1.28
G6c_05	13	13	94.62	94.5	94.76	95.02	2.18	2.19
G6c_15	20	19	84.77	84.81	84.23	84.49	1.94	1.95
G6c_25	38	28	73.8	74.16	75.52	75.4	1.70	1.71
G6c_40	89	71	59.01	59.41	59.39	59.77	1.17	1.2
Iris	8	6	94.96	94.98	94.98	94.96	2.19	2.19
Abalone	235	165	63.15	62.25	64.71	63.25	1.16	1.21
Pageblocks	21	21	89.67	89.59	95.21	95.29	2.18	2.19

Table 2: Performance of Gaussian ARTMAP (GAM) versus distributed GAM (dGAM)

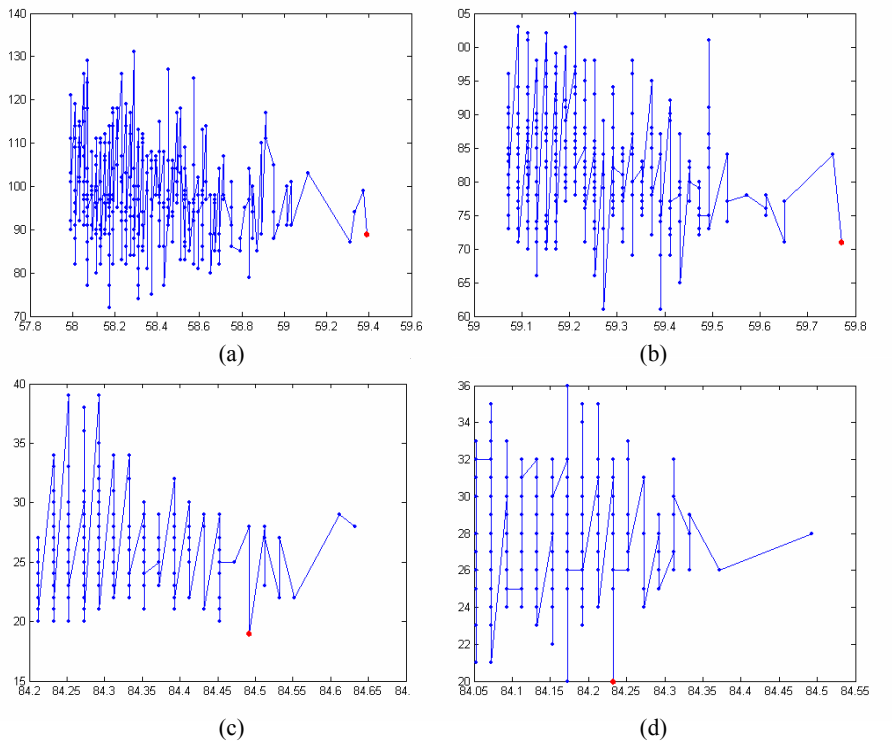

Figure 2: Top networks (Node Count (Vertical axis) vs. PCC on the validation set (Horizontal axis)) for G6c_15 ((a) GAM, (b) dGAM) and G6c_40 ((c) GAM, (d) dGAM); red dot denotes top fitness function value

by dGAM, compared with the nodes created by GAM, for the Abalone dataset (235 nodes for GAM versus 165 nodes for dGAM). The Abalone dataset can also be thought of as a dataset, where data belonging to different classes significantly overlap (since its classification accuracy on unseen data (validation or test set) is rather poor (only 60%)). For problems, where the overlap of data is not significant there is no difference between the sizes of GAM and dGAM.

CONCLUSIONS

We have experimented with the Gaussian (GAM) and distributed Gaussian (dGAM) networks. The goal of our experimentation is to determine of whether distributed learning has the inherent ability to improve the category proliferation problem observed by a number of winner-take-all ART networks. The conclusion from our experiments is that distributed learning in the context of how it was defined by Williamson in dGAM has a beneficial effect on the category proliferation problem, when the amount of data overlap, belonging to different classes, is significant. In the process of comparing the two networks with each other, we defined a fitness function whose value was affected by both network's generalization and network's size. It was verified experimentally that

this fitness function works well, and it gave us an automatic and reliable way of defining the optimal GAM and dGAM networks.

ACKNOWLEDGMENTS

This work was supported in part by a National Science Foundation (NSF) grant CRCD 0203446, the National Science Foundation grant DUE 05254209, and the NSF grant CCLI 0341601.

REFERENCES

Al-Daraiseh, A., Georgiopoulos, M., Wu, A. S., Anagnostopoulos, G. C., and Mollghasemi, M., 2006, "GFAM: Evolving Fuzzy ARTMAP Neural Networks," *Proceedings of the 19th Florida Artificial Intelligence Symposium (FLAIRS-2206)*, Melbourne Beach, FL.

Anagnostopoulos, G., and Georgiopoulos, M., 2001, "Ellipsoid ART and ARTMAP for incremental clustering and classification," *IEEE-INNS International Joint Conference on Neural Networks*, Washington, DC, July 14-19, pp. 1221-1226.

Anagnostopoulos, G.C., Bharadwaj, M., Georgiopoulos, M., Verzi, S.J., and Heileman, G. L., 2003, "Exemplar-based pattern recognition via semi-supervised learning", *Proc. of the International Joint Conference on Neural Networks*, Vol. 4, pp. 2782-2787, Portland, Oregon.

Carpenter, G. A. et al., 1992, "Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multi-dimensional maps," *IEEE Transactions on Neural Networks*, Vol. 3, No. 5, pp. 698-713.

Carpenter, G. and Milenova, B., 1998, "Distributed ARTMAP: A neural network for fast distributed supervised learning", *Neural Networks*, Vol. 11, pp. 323-336.

Charalampidis, D., Kasparis, T., and Georgiopoulos, M., 2001, "Classification of noisy signals using Fuzzy ARTMAP neural networks", *IEEE Trans. Neural Networks*, Vol. 12, pp.1023-1036.

Gomez-Sanchez, E., Dimitriadis, Y.A., Cano-Izquierdo, J.M., Lopez-Coronado, J., 2001, "Safe-μARTMAP: a new solution for reducing category proliferation in Fuzzy ARTMAP", *Proc. of the IEEE International Joint Conference on Neural Networks*, Vol. 2, pp. 1197-1202.

Gomez-Sanchez, E., Dimitriadis, Y.A., Cano-Izquierdo, J.M., and Lopez-Coronado, J., 2002, "μARTMAP: use of mutual information for category reduction in Fuzzy ARTMAP", *IEEE Trans. Neural Networks*, Vol. 13, pp. 58-69.

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

Hettich, S., Blake, C.L., and Merz, C.J, 1998, UCI Repository of machine learning databases [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science.

Koufakou, A., Georgiopoulos, M., Anagnostopoulos, G.C., and Kasparis, T., 2001, "Cross-validation in Fuzzy ARTMAP for large databases", *Neural Networks*, Vol. 14, pp. 1279-1291.

Marriott, S., and Harrison, R.F., 1995, "A modified Fuzzy ARTMAP architecture for the approximation of noisy mappings", *Neural Networks*, Vol. 8, pp. 619-641.

Parrado-Hernandez E., Gomez-Sanchez, E., and Dimitriadis, Y.A., 2003, "Study of distributed learning as a solution to category proliferation in Fuzzy ARTMAP-based neural systems", *Neural Networks*, Vol. 16, pp. 1039-1057.

Verzi, S.J., Georgiopoulos, M., Heileman, G.L., and Healy, M., 2001, "Rademacher penalization applied to Fuzzy ARTMAP and Boosted ARTMAP", *Proc. of the IEEE-INNS International Joint Conference on Neural Network*, pp. 1191-1196, Washington, DC.

Williamson, J. R., 1996, "Gaussian ARTMAP: A Neural Network for Fast Incremental Learning of Noisy Multi-Dimensional Maps," *Neural Networks*, Vol. 9, No. 5, pp. 881-897.

Williamson, J. R., 1997, "A constructive, incremental-learning network for mixture modeling and classification," *Neural Computation*, Vol. 9, pp. 1517-1543.