Experiments with Safe μARTMAP and Comparisons to Other ART Networks

Mingyu Zhong, Bryan Rosander, Michael Georgiopoulos,
Georgios Anagnostopoulos, Mansooreh Mollaghasemi, and Samuel Richie

***Abstract*—Fuzzy ARTMAP (FAM) is currently considered as one of the premier neural network architectures in solving classification problems. 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 overlapping. A modified version of Fuzzy ARTMAP, referred to as Safe μARTMAP, has been introduced in the literature by Gomez-Sanchez and his colleagues, in order to remedy the category proliferation problem. However, Safe μARTMAP's performance depends on a number of network parameters. In this paper, we analyzed each parameter of Safe μARTMAP to set up the candidate values for evaluation. We performed an exhaustive experimentation to identify good default values for the Safe μARTMAP network parameters for a variety of problems (simulated and real problems), and compared the best performing Safe μARTMAP network with other best performing ART networks, including other ART networks that claim that resolve the category proliferation problem in Fuzzy ARTMAP.**

I. Introduction

THE Adaptive Resonance Theory (ART) was developed by Grossberg [1]. One of the most celebrated ART architectures is Fuzzy ARTMAP [2], 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 overlapping.

In this paper we focus our attention on one Fuzzy ARTMAP modification, called Safe μARTMAP, and introduced by Gomez-Sanchez, et al [3] that addresses this category proliferation problem. We first analyze each parameter of Safe μARTMAP and provide representative values for each parameter. We then perform an exhaustive experimentation to identify good default μARTMAP network parameter for a variety of problems (simulated data and real data). We also compare the best performing μARTMAP network with other best performing ART networks, such as Fuzzy ARTMAP [2], Ellipsoidal ARTMAP [4], Gaussian ARTMAP [5] [6], and their semi-supervised versions (see [7] and [10)].

In this paper, we assume that the reader is familiar with Fuzzy ARTMAP, Ellipsoidal ARTMAP], and Gaussian ARTMAP, and their semi-supervised versions, but most importantly we assume that the reader is familiar with μARTMAP and Safe μARTMAP (see [3] and [8]).

II. μARTMAP Architecture

As it is the case with other ART architectures that solve classification problems, μARTMAP consists of three layers of nodes: the input layer, the category representation layer, and the output layer. When an input pattern is presented, it first goes through a pre-processing phase called complementary encoding (for more details see [2]). The expanded input, designated as **I**, is then fed to the category representation layer. The category representation layer of Safe μARTMAP contains nodes, referred to as category nodes. Each one of these nodes represents (in a compressed form) a group of input patterns. Each category is represented

Manuscript received December 5, 2005. This work was supported in part by a National Science Foundation (NSF) grant CRCD: 0203446. Georgios Anagnostopoulos and Michael Georgiopoulos acknowledge the partial support from the NSF grant CCLI 0341601.

M. Zhong is with the School of Electrical Engineering and Computer Science, University of Central Florida, Orlando, FL 32816, USA (e-mail: myzhong@ucf.edu).

B. Rosander was with the School of Electrical Engineering and Computer Science, University of Central Florida, Orlando, FL 32816, USA (e-mail: bdrosander@gmail.com).

M. Georgiopoulos is with the School of Electrical Engineering and Computer Science, Orlando, FL 32816, USA (phone: (407) 823-5338, fax: (407) 823 5835; e-mail: michaelg@mail.ucf.edu).

G. Anagnostopoulos is with the Department of Electrical and Computer Engineering, Florida Institute of Technology, Melbourne, FL 32901, USA (e-mail: georgio@fit.edu).

M. Mollaghasemi is with the Department of Industrial Engineering and Management Systems, University of Central Florida, Orlando, FL 32816, USA (e-mail: mollagha@mail.ucf.edu).

S. Richie is with the School of Electrical Engineering and Computer Science, University of Central Florida, Orlando, FL 32816, USA (e-mail: richie@mail.ucf.edu).

by a weight vector $\mathbf{w}_j^a$ (the subscript index j designates the category), which is referred to as template with a hyper-box geometrical interpretation – the boundary of this hyperbox encloses all the input patterns that chose and were encoded by the corresponding category, as in Fuzzy ARTMAP (this is the kind of compression of input patterns that Safe μARTMAP enforces). One of the differences between μARTMAP and Fuzzy ARTMAP is that the training patterns encoded by a category in μARTMAP can belong to various classes. Through the weights $\mathbf{W}_j^{ab} = (W_{j1}^{ab}, \ldots, W_{jk}^{ab}, \ldots, W_{jN_b}^{ab})$ which emanate from the activated category to the output layer and store the class distribution of the activated category, the input pattern **I** is mapped to the major class associated with this activated category.

The training phase of μARTMAP is succinctly described as follows (Steps 1-2). In all of the following equations, the notation $|\cdot|$ stands for the size of a vector and it is equal to the sum of its components, while the notation $\wedge$ stands for the "fuzzy-min" of two vectors and it is defined to be the minimum, component-wise of these two vectors

1) (Learning Phase) Find the nearest category in the category representation layer of μARTMAP that resonates with the input patterns. That is, for each pattern **I**, the existing (committed) categories compete and the winner category is chosen to be the one that maximizes the following value (called bottom-up input):

$$T_j(\mathbf{I}, \mathbf{w}_j^a, \alpha) = \frac{|\mathbf{I} \wedge \mathbf{w}_j^a|}{\alpha + |\mathbf{w}_j^a|} \tag{1}$$

However, if the winner fails either of the following tests, it will be deactivated and the next winning category will be chosen and tested.

a. Vigilance test: $|\mathbf{I} \wedge \mathbf{w}_j^a| / M_a \geq \rho_a$ (2)

where M_a is the dimensionality of the unexpanded input. The parameter ρ_a is category-specific: each category has its own ρ_a which is never changed once initialized. This test prevents the category from growing too large.

b. Entropy test: $h_j \leq h_{\max}$ (3)

$$h_j = \frac{|\mathbf{W}_j^{ab}|}{|\mathbf{W}^{ab}|} ent(j) = -\frac{|\mathbf{W}_j^{ab}|}{|\mathbf{W}^{ab}|} \sum_{k=1}^{N_b} \frac{\mathbf{W}_{jk}^{ab}}{|\mathbf{W}_j^{ab}|} \log_2\left(\frac{W_{jk}^{ab}}{|\mathbf{W}_j^{ab}|}\right) \tag{4}$$

$h_{\max}$ is a predefined parameter. $\mathbf{W}_j^{ab}$ is temporarily updated according to (5) before this test and restored afterwards. This test ensures the accuracy of the category.

If none of the committed categories passes the above tests, an uncommitted node will be selected, with $\mathbf{w}_j^a$ initialized as a vector of all ones, $\mathbf{W}_j^{ab}$ initialized as a vector of all zeros, and its ρ_a initialized as the current global vigilance level. In any case, the selected node will learn the pattern by updating its weight vectors, as follows:

$$\mathbf{w}_j^a = \mathbf{w}_j^a \wedge \mathbf{I} \qquad W_{jk}^{ab} = W_{jk}^{ab} + 1 \tag{5}$$

where $k = label(\mathbf{I})$.

2) (Offline Evaluation Phase) After the learning phase is finished (i.e., all input/associated label pairs of the training set have chosen a committed node) we present all the input patterns again to check the total entropy of the created categories, without changing any $\mathbf{w}_j^a$ vector. One pass of the learning phase and the offline evaluation phase is called one *epoch*.

a. If the total entropy is below a designated threshold $H_{\max}$ or the maximum number of epochs is reached, training is completed.

b. If not, the category that contributes the most to the total entropy value is destroyed, and the global vigilance level increased at:

$$\rho_a = \min\left(1, \frac{|\mathbf{w}_j^a|}{M_a} + \Delta\rho\right) \tag{6}$$

where j is the index of the destroyed category. Note that this change affects only future categories rather than the committed ones. In the learning phase of the next epoch, we present to μARTMAP only the training patterns that chose the destroyed category in the learning phase (rather than the offline evaluation phase) of this epoch or the previous epochs. In the offline evaluation of the next epoch, we still present all the patterns.

In the performance phase of μARTMAP, a test input is presented to the input layer of μARTMAP and the node in the category representation layer that receives the maximum bottom-up input (T_j) is chosen, according to (1), but without any test. Then the predicted label for this test input is chosen to be $\arg\max_k W_{jk}^{ab}$.

Safe μARTMAP is an improved version of μARTMAP. It differs from μARTMAP only in that it has a third test in the learning phase:

$$\frac{|\mathbf{w}_j^a| - |\mathbf{I} \wedge \mathbf{w}_j^a|}{M_a} \leq \delta \tag{7}$$

where δ is also specified by the user. This test requires that the size change of the winner category should not be too large due to a single pattern. If the winner category fails this test, no other categories will be chosen to learn this pattern at this point. Instead, this pattern remains "unlearned". After all patterns are presented (which is called a *pass*), the unlearned patterns are presented again in the next *pass*. The previous winner categories may learn these patterns if they pass this test (they might have been expanded and thus it is possible that (7) is satisfied now). If no pattern is learned in a whole pass, an unlearned pattern will be selected and a new category will be committed to learn this pattern; then all the other unlearned patterns are presented in the next pass. The above is repeated until all patterns are learned. In this way, the learning phase of a single epoch may consist of many passes.

III. μARTMAP Parameters

A. Parameters α and e

The choice parameter α, first introduced in Fuzzy ARTMAP, affects the competition of the nodes according to (1). For μARTMAP, it is desired that:

1) if a point (representing an input pattern) is inside two hyper-boxes (whose boundaries are defined by the corresponding categories in the network), it should choose the smaller hyper-box;
2) if a point is inside one hyper-box and outside another hyper-box, it should choose the former one regardless of the size of either hyper-box.

Condition 1) simply requires $\alpha > 0$. Condition 2) cannot be satisfied if $|\mathbf{w}_j^a|$can be arbitrarily small (or the hyper-box can be arbitrarily large). If the minimum value of the components in the patterns is 0 and the maximum value is 1, no positive α value allows a hyper-box to cover the whole input space (which means $|\mathbf{w}_j^a| = |\mathbf{I} \wedge \mathbf{w}_j^a| = 0$) and satisfy condition 2) at the same time. The authors of μARTMAP (personal communication with Gomez-Sanchez) adjusted the algorithm by normalizing the input elements to the interval [e, 1–e] instead of [0, 1], and require both the following:

$$\alpha << \min |\mathbf{w}_j^a| = 2M_a e \qquad (8)$$

$$e<<1 \qquad (9)$$

Equation (8) implies that when a point is inside a box, the corresponding T_j is close to one even if the box covers the whole input space (note that the parameter M_a denotes the dimensionality of the input pattern **I**). Equation (9) prevents the vigilance test from passing when the vigilance parameter ρ_a is small, since $|\mathbf{I} \wedge \mathbf{w}_j^a| / M_a \geq 2M_a e / M_a = 2e$.

In our experiments, the choice parameter α was set to 0.01 and 0.001 for all ART algorithms (note that the minimum M_a was 2). Due to the above constraints (1/400<<e<<1), we set e to 0.05 in our experiments. We also did some preliminary experiments and found that the μARTMAP is not sensitive to α or to e as long as the above constraints are satisfied.

B. Parameter h_{max}

The parameter $h_{\max}$ controls the impurity of each node (category) defined in (4), according to (3). A node may be both very large and very pure (which means most of the patterns that select it have the same class label). μARTMAP permits the creation of a large node (category) by allowing ρ_a to be zero, and maintains the accuracy by controlling the impurity. This is the main reason why μARTMAP can achieve a good accuracy with very few category nodes.

The parameter $h_{\max}$ affects the training process mostly in the first epoch. Setting $h_{\max}$=0 means all the nodes must be completely pure when created or expanded; they may become impure as more patterns are presented and more nodes are created. In most cases, $h_{\max}$=0 causes each node to learn very few nodes and thus results in a large network with poor generalization (accuracy on unseen data). Setting $h_{\max}=\infty$ means that the entropy test always passes.

According to (4), it is difficult to estimate a good $h_{\max}$ value, since $|\mathbf{W}_j^{ab}|$, the number of patterns learned by category j, is not easy to predict. Moreover, h_j is much more sensitive to the order in which the patterns are presented than the total entropy is during the offline evaluation phase. For example, suppose there is only one category in the network, and the first four patterns that μARTMAP learned have the class labels 1, 1, 1, 2, respectively (as in the order of the list presentation). The h_j values would be 0, 0, 0, 0.8113, after category j learns these patterns. If we swap the second and the fourth patterns, then the h_j values would be 0, 1, 0.9183, 0.8113, after category j learns these patterns. If we set $h_{\max}$ to 0.9, then category j would learn all the four patterns in the first case (before swapping), but it would not learn the pattern with class label 2 in the second case (after swapping).

We assume that the proper value of $h_{\max}$ is proportional to the proper value of $H_{\max}$ and varied the ratio between $h_{\max}$ and $H_{\max}$ in order to search for the optimal $h_{\max}$ value in our experiments.

C. Parameter H_{max}

The parameter $H_{\max}$ controls the impurity of the whole μARTMAP network, which is defined as the sum of the impurities of all the categories formed in the training phase of μARTMAP. The parameter $H_{\max}$ terminates the training process to prevent over-training. $H_{\max}$ has a direct effect on the final accuracy of the μARTMAP. Setting $H_{\max}$=0 means that the ARTMAP must have 100% accuracy on the training set in the offline evaluation, which is usually impractical. In most cases, $H_{\max}$=0 not only keeps the training algorithm running for a long time, but also over fits the network to the training set as $h_{\max}$=0 does. On the other hand, setting $H_{\max}$ to a very high value will terminate the training process too soon and result in low generalization, as well.

Apparently, the proper $H_{\max}$ value is problem-dependent. Nevertheless, we can come up with some estimates of the total entropy H. First, let us define by N_b the number of classes (namely the number of nodes in the output layer), and $\hat{A}$ the expected accuracy given by the user and assumed in the interval (1/N_b, 1]. If there is a known theoretical optimal accuracy in a problem, assume $\hat{A}$ is equal to this theoretically optimal accuracy. Of course, $\hat{A}$ is sometimes unknown. Nevertheless, estimating $\hat{A}$ (using for example information existing in the literature) is much easier than guessing $H_{\max}$. It is proven in the appendix that if the accuracy of the network is (8) $\hat{A}$, the network entropy H is bounded as follows:

$$H_L \leq H \leq H_U$$

$$H_L = -\log_2 \hat{A}$$

$$H_U = -\hat{A}\log_2 \hat{A} - \left(1-\hat{A}\right)\log_2 \frac{1-\hat{A}}{N_b - 1}$$

H_L is the entropy when $1/\hat{A}$ is an integer and the proportions of the classes in all categories are either 0 or $\hat{A}$. H_U is the entropy when the proportion of the major class in each category is $\hat{A}$ and the other classes are evenly distributed for all categories. However, neither H_L nor H_U is a good estimate for $H_{\max}$, since both of them can be quite different from the actual entropy. Therefore, two other estimates for the entropy $H_{\max}$ are given in the appendix:

$$H_{E1} = \frac{(1-\hat{A})N_b \log_2 N_b}{N_b - 1} \qquad (10)$$

$$H_{E2} = -\frac{N_b(1-\hat{A}) - (N_b - 1)p}{1-p}\log_2 p - \log_2 \hat{A} \qquad (11)$$

where p is the solution in [0, 1] to the equation $(1-p)/(1-p^{N_b}) = \hat{A}$. H_{E1} is the entropy when the accuracies of all the categories are either 1 (pure categories) or $1/N_b$ (completely impure categories); H_{E2} is the entropy when the accuracies of all the categories are equal to $\hat{A}$, and the proportion of the minor classes within each category forms a geometric progress. In our experiments, we have used all these estimates of the entropy to come up with legitimate values of $H_{\max}$ to run our μARTMAP experiments.

D.Parameters $\overline{\rho}_a$, $\Delta\rho$, and δ

The baseline vigilance threshold $\overline{\rho}_a$ can be initialized as any value in [0, 1]. In our experiments, we chose $\overline{\rho}_a$ within the set of values {0, 0.2, 0.4, 0.6, 0.8}.

$\Delta\rho$ is introduced to make sure the most entropic category cannot be created again after it is removed. Our preliminary experiments show that this parameter does not affect the network performance as long as it is far less than 1. In our experiments we fixed $\Delta\rho$ to 0.02.

The parameter δ controls the *size change per pattern* of each category, as it is demonstrated in (7). This parameter alleviates the overlapping problem in μARTMAP and reduces the effect of μARTMAP's dependence on the order of pattern presentation in the training set. Small δ means that the size change must be small. Usually it will cause longer training time because in each epoch, more patterns will be placed into the unlearned set for many passes, until they are finally learned. If δ=0, then no category can increase its size, which is equivalent to set $\overline{\rho}_a$=1. If $\delta \geq 1-\overline{\rho}_a$, then (7) is always satisfied, and Safe μARTMAP reduces to μARTMAP. The optimal δ value is also dependent on the distribution of patterns. Although δ makes the algorithm less sensitive to the order of pattern presentation in the training set, the optimal value of δ depends on the distribution of the data points more than the other parameters do, since the former is even sensitive to the number of patterns.

IV. Experiments

We have performed a number of experiments with μARTMAP. The purpose of these experiments was two-fold: First, we have made an effort to identify "optimal" settings of the network parameters in μARTMAP. Secondly, we compared μARTMAP's performance with the performance of other ART classifiers in the literature, including those attempting to address the category proliferation problem in Fuzzy ARTMAP. In the sequel, we are reporting results from both of these sets of experiments.

A. Databases

We experimented with both artificial and real databases. In particular, the artificial databases correspond to 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 (or $\hat{A} = 0.95$). There are a total of 3×4=12 Gaussian databases. Each Gaussian database has approximately 500 points in the training set and 5000 in the validation set and the test set. Each class is equal probable to happen (which means A_0, the accuracy of wild guess, equals 1/#classes). 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 a 5% overlap database.

The real databases are the Iris (500/4800/4800 points, 2 attributes, 2 classes, $A_0 = 0.5,\ \ \hat{A} = 0.95$), Page-blocks (500/2486/2487 points, 10 attributes, 5 classes, $A_0 = 0.83,\ \ \hat{A} = 0.95$) and Abalone (501/1838/1838 points, 7 attributes, 3 classes, $A_0 = 0.33,\ \ \hat{A} = 0.6$) databases, which were obtained from the UCI Repository [9] (note that the Iris database has been expanded in size by introducing noisy patterns in the already existing set of 150 patterns).

B. Parameter Settings:

For each database, we simulated Safe μARTMAP with all the following combinations of the five Safe μARTMAP parameters $H_{\max}$, $h_{\max}$, $\overline{\rho}_a$, α and δ.

$$H_{\max} = \{H_1,\ H_2,\ H_3,\ H_4,\ H_5\}$$

$$H_1 = \frac{1}{2}(H_L + H_2)$$

$$H_2 = \min\{H_{E1}, H_{E2}\}$$

$$H_3 = \frac{1}{2}(H_{E1} + H_{E2})$$

$$H_4 = \max\{H_{E1}, H_{E2}\}$$

$$H_5 = \begin{cases} H_U, & H_U > H_4 \\ 2H_U - H_3, & H_U = H_4 \end{cases}$$

$$h_{\max} = \left\{0,\ \frac{1}{4}H_{\max},\ \frac{1}{2}H_{\max},\ H_{\max},\ 2H_{\max}, \infty\right\}$$

$$\overline{\rho}_a = \left\{0,\ \frac{1}{5},\ \frac{2}{5},\ \frac{3}{5},\ \frac{4}{5}\right\}$$

$$\Delta\rho = 0.02$$

$$\alpha = \{0.001,\ 0.01\}$$

$$\delta = \left\{\frac{1}{25}(1-\overline{\rho}_a),\ \frac{1}{5}(1-\overline{\rho}_a),\ (1-\overline{\rho}_a)\right\}$$

$$e = 0.05$$

$$\mathit{MaxNumberOfEpochs} = 100$$

We experimented with all the above parameter combinations, which amounted to 5×6×5×2×3=900 combinations.

C. Experimental Procedure – Experimental Results

As we have emphasized above, our experiments were divided into two parts. In the first part, we compared Safe μARTMAP with other ARTMAP classifiers (see Table 1). For each database, we evaluated all the possible parameter combinations of Safe μARTMAP for a 100 different orders of the pattern list presentation (the performance of μARTMAP depends on the order according to which patterns are presented in the training set). The 100 orders were fixed in all experiments and are exactly the same as those used to test the other ARTMAP algorithms. Therefore, for each database, we trained 900×100=90000 μARTMAP networks. We evaluated each network by the following score:

$$score = \frac{A - A_0}{\hat{A} - A_0} 0.9^{\left(N_a/5N_b\right)^2} \qquad (12)$$

where A is the accuracy on the validation set, N_a is the number of categories formed in the training phase of μARTMAP, and A_0, $\hat{A}$, and N_b were defined for each dataset in the *Databases* subsection. In the definition of the *score* we have used the normalized accuracy of a database (i.e., $\left(A - A_0\right)/\left(\hat{A} - A_0\right)$ instead of the actual accuracy (i.e., A) so that the scores corresponding to different databases can be summed up without bias. Apparently, the above score is monotonically increasing with A and monotonically increasing with N_a; when N_a is small, $\partial score/\partial N_a \approx 0$.

For the Gaussian databases (for which we know the exact value of $\hat{A}$), we examined the parameters of the best networks we previously selected. For each parameter combination and each one of the 12 Gaussian databases, we set the score of the parameter combination as the maximum score of the 100 networks trained with 100 different orders of pattern presentation and for that specific parameter combination. Then, for every parameter combination we have 12 of these maximum scores corresponding to the 12 Gaussian datasets. We sum up these 12 maximum score numbers for every parameter combination, and then we rank these sums from highest to lowest. The highest 5 of these sums of maximum scores point us to the best 5 parameter settings for Safe μARTMAP.

In Table I we list all Safe μARTMAP's performance with the chosen 5 sets of best parameters over all the databases, including the Gaussian databases and the real ones. For comparison, we also list, in the first column, the performance corresponding to the problem-dependent best parameter combination found in validation. An obvious observation is that the 5 best parameters produce almost optimal results and they do not differ very much in performance. It is also important to know that the identification of good, default parameter values for Safe μARTMAP is saving us significant computations when Safe μARTMAP is used with a new database. Furthermore, the identification of good, default parameter values is essential in cases where the number of data-points in our dataset is not large enough to allow us the luxury of splitting the data into training and validation sets and performing cross-validation using the validation set.

Although the networks were ranked by cross-validation, the accuracy on the validation set is not shown, because it is always close to the accuracy on the test set.

We observe that the elements in the fourth best parameter combination, except δ, appear in most other best parameter combinations. Thus, we suggest the following optimal settings, assuming the maximum number of epochs is large enough:

$$H_{\max} = H_4,\ h_{\max} = \infty,\ \overline{\rho}_a = 0,\ \alpha = 0.001 \qquad (13)$$

We do not claim an optimal δ value because it depends on the size of the training set and the relationship is not clear yet.

TABLE II
BEST PERFORMANCE OF ALL ART ALGORITHMS

Database	Safe μAM		FAM		ssFAM		EAM		ssEAM		GAM		ssGAM		dGAM		ssdGAM	
	%Acc	N_a	%Acc	N_a	%Acc	N_a	%Acc	N_a	%Acc	N_a	%Acc	N_a	%Acc	N_a	%Acc	N_a	%Acc	N_a
G2c-05	95	2	91	14	95	2	92	26	95	2	94	4	94	4	95	4	95	2
G2c-15	85	2	78	47	85	3	78	79	85	2	85	6	85	2	85	8	85	2
G2c-25	75	2	64	75	75	2	65	123	75	2	75	6	75	2	75	7	75	2
G2c-40	61	3	54	110	61	3	54	177	61	2	60	12	61	3	60	9	61	3
G4c-05	95	4	93	21	94	7	93	24	94	4	95	10	95	4	95	10	95	4
G4c-15	83	4	78	55	81	11	78	76	83	4	84	18	84	9	84	18	84	9
G4c-25	75	4	67	101	71	9	67	110	73	4	74	49	72	21	75	46	75	35
G4c-40	60	5	49	127	58	14	50	161	56	13	58	36	59	14	59	36	59	14
G6c-05	94	9	92	26	91	11	92	23	94	7	94	12	94	8	95	13	95	6
G6c-15	81	6	76	58	81	7	76	85	82	6	85	19	84	13	85	19	84	11
G6c-25	71	13	67	87	70	15	64	124	71	7	73	30	73	20	74	32	73	20
G6c-40	58	11	51	196	56	17	51	193	54	17	59	70	56	13	59	70	56	13
Modified Iris	95	2	92	23	93	8	93	28	95	2	95	4	95	2	95	4	95	2
Abalone	57	4	46	29	60	6	46	86	57	7	46	12	55	3	46	12	55	3
Page Blocks	89	6	83	10	91	3	77	34	90	3	86	9	89	5	86	9	89	5

Safe μAM: Safe μARTMAP; FAM: Fuzzy ARTMAP; EAM: Ellipsoidal ARTMAP; GAM: Gaussian ARTMAP; dGAM: Distributed Gaussian ARTMAP; ss* : semi-supervised version

%Acc: the accuracy (in percentage) on the test set; N_a: the number of categories; Epochs: the number of epochs required in training.

For $H_{\max}$, we are very confident since all the best 5 parameter combinations have this value. In fact, all the best 65 networks have $H_{\max} = H_4$. This result is not surprising, since H_4 is a good estimate of the entropy without over-training. $\overline{\rho}_a = 0$ means we should allow a category to be very large in the first epoch, which is one of the benefits of μARTMAP. $\alpha = 0.001$ is better than $\alpha = 0.01$, which agrees with (8).

Although the optimal value of $h_{\max}$ seems unexpected, it can be explained as follows. This value allows a category to be very impure and tends to result in many more epochs of training because many impure categories must be removed in the future. In the first epoch, large categories will be created due to the small $\overline{\rho}_a$ value. In only a few epochs, the size of the categories will be controlled by ρ_a only. The number of categories will be very small in the beginning and it will grow slowly afterwards, until the total entropy is no more than $H_{\max}$. Therefore, the minimum number of categories may be achieved. Of course, sufficient epochs of training must be allowed, or otherwise the training process would be terminated prematurely and the network performance would be even worse than when $h_{\max} = 0$. In contrast, setting $h_{\max} = 0$ will cause a large number of categories to be created in the first epoch, including many trivial categories. In this case, the training process may finish in only one epoch, resulting in a network that may still be over-trained, exhibiting poor

TABLE I
BEST PARAMETER COMBINATIONS FOR SAFE μARTMAP

Rank	Best			1			2			3			4			5		
$H_{\max}$	-			H_4			H_4			H_4			H_4			H_4		
$h_{\max} / H_{\max}$	-			∞			∞			1			∞			∞		
$\overline{\rho}_a$	-			0.4			0			0.2			0			0.2		
a	-			0.001			0.01			0.001			0.001			0.001		
$\delta/(1-\overline{\rho}_a)$	-			0.2			0.2			0.2			1			1		
Database	%Acc	N_a	Epochs	%Acc	N_a	Epochs	%Acc	N_a	Epochs	%Acc	N_a	Epochs	%Acc	N_a	Epochs	%Acc	N_a	Epochs
G2c-05	95.22	2	1	95.16	2	14	95.14	2	4	95.2	2	1	95.2	3	10	95.2	3	10
G2c-15	85	2	1	85.06	2	4	84.98	3	15	85.06	2	1	85.24	2	27	85.24	2	27
G2c-25	74.98	2	1	74.96	2	16	74.96	3	18	74.18	2	1	75.02	3	8	75.02	3	8
G2c-40	61.4	3	1	61.54	4	8	61.34	4	18	61.44	3	10	61.32	4	32	61.32	4	32
G4c-05	95.04	4	22	94.82	4	25	94.36	6	50	94.64	4	1	94.46	6	48	94.46	6	48
G4c-15	83.28	4	20	81.74	6	44	84.18	7	65	83.58	9	82	83.64	9	61	83.64	9	61
G4c-25	74.5	4	44	74.78	5	37	75.06	6	52	75.06	4	48	75.02	6	49	75.02	6	49
G4c-40	59.76	5	39	59.26	4	52	59.76	5	39	58.84	5	41	59.72	7	37	59.72	7	37
G6c-05	93.57	9	9	93.09	10	85	91.87	9	74	93.23	10	58	93.53	13	93	93.53	13	93
G6c-15	80.92	6	1	81.18	12	100	81.87	13	100	81.16	14	76	82.27	12	100	82.27	12	100
G6c-25	70.74	13	88	71.18	13	83	69.54	14	85	69.76	11	100	69.16	13	90	69.16	13	90
G6c-40	58.03	11	100	56.77	16	100	56.45	13	81	56.41	13	100	56.3	14	77	56.3	14	77
Modified Iris	94.92	2	2	94.92	4	10	95.15	4	19	94.92	4	16	94.63	3	10	94.63	3	12
Abalone	57.18	4	4	55.06	2	2	54.08	2	4	54.52	3	2	53.59	2	6	53.59	2	6
Page Blocks	88.82	6	17	88.34	5	10	92.32	5	24	89.14	8	35	89.75	4	11	89.75	4	11

%Acc: the accuracy (in percentage) on the test set; N_a: the number of categories; Epochs: the number of epochs spent in training
Note that the parameter combinations in the column “Best” is problem dependent; they have the best score in validation, not testing

generalization.

In the second part, we compare the best Safe μARTMAP network to the best of each other ARTMAP architectures, namely Fuzzy ARTMAP, Ellipsoidal ARTMAP and Gaussian ARTMAP, and compares very favorably with ssFAM, ssEAM, and ssGAM and ssdGAM. Actually, the algorithms that produce as good results as safe μARTMAP are ssEAM and ssdGAM. The term "best" is also based on the score defined in (12) in validation. The best of the 90000 trained ARTMAP networks is selected and shown in Table II. According to the results, one can conclude that Safe μARTMAP can achieve almost the best accuracy using the smallest network, as long as the parameters are set properly.

V. Conclusions

Safe μARTMAP is one of the recently proposed ART architectures, which can produce small size classifiers with high accuracy. The main issue of using μARTMAP is the correct selection of its many parameters. In this paper, we studied the effect of the parameters, both theoretically and experimentally. Furthermore, we have identified a procedure that came up with a way of choosing good default μARTMAP parameter values, independently of the database used, despite the obvious fact that the best μARTMAP parameter values are data-base dependent. This is a significant simplification for anyone experimenting with μARTMAP on new datasets. It is also very beneficial in cases when the dataset is small and we do not have the option of splitting the dataset in training and validation sets. Also, we compared the performance of μARTMAP with a number of ART classifiers, including a number of them that have been reported in the literature and claim that they also address the category proliferation problem in Fuzzy ARTMAP. The result from this experimentation is that μARTMAP outperforms Fuzzy ARTMAP (FAM), Ellipsoidal ARTMAP (EAM), and Gaussian ARTMAP (GAM), and it exhibits comparable performance with semi-supervised EAM and distributed GAM. Finally, it is worth pointing out that our performance comparison of various ART algorithms and the identification of good, default parameter values for μARTMAP relied on a performance measure (score) that takes into consideration both the accuracy of the network on a cross-validation set and the size of the network that training creates. Despite its obvious benefits this is an approach that has not been quantified in the ART literature before.

Appendix – Estimates of $H_{\max}$

A. Preliminaries

It is important to note that μARTMAP utilizes the $\mathbf{W}^{ab}$ matrix for computing the node entropy in the entropy test during training as in (4) while, it computes the total entropy based on another matrix $\mathbf{V}^{ab}$, which is the same as $\mathbf{W}^{ab}$ except that it is computed in offline evaluation and reset at the end of each epoch. Before we study the relationship between $H_{\max}$ and $\hat{A}$, we have to define four accuracies: the accuracy on the training set produced by using $\mathbf{W}^{ab}$ matrix (designated by A_W^{Train}), the accuracy on the training set produced by using $\mathbf{V}^{ab}$ (designated by A_V^{Train}), the accuracy on the validation set produced by using $\mathbf{W}^{ab}$ (designated by A_W^{Val}), and the accuracy on the test set produced by using $\mathbf{W}^{ab}$ matrix (designated by A_W^{Test}). After training, only the $\mathbf{W}^{ab}$ matrix is used to produce the classification results. For this reason, we do not examine the accuracy on the validation/test set using $\mathbf{V}^{ab}$.

It is a well known result that when the accuracy on the training set is increased too much, the accuracy on the test set will drop since the network is over trained. Here we do not consider the case where the database is so small that the training set might not be representative. Following are our observations from our experiments, whose results are not shown in this paper.

1) When $A_V^{Train} < \hat{A}$, $\max A_W^{Test}$ (the A_W^{Test} value of the network with the best parameter settings) increases with A_V^{Train} and $A_V^{Train} < \max A_W^{Test} < \hat{A}$; when $A_V^{Train} = \hat{A}$, $\max A_W^{Test} \approx \hat{A}$; when $A_V^{Train} > \hat{A}$, $\max A_W^{Test}$ decreases with A_V^{Train} and $\max A_W^{Test} < \hat{A}$
2) $A_W^{Test} \approx A_W^{Val}$. This is reasonable since both the test set and the validation set are unseen by the network, and they represent the same problem.

It is clear that the training algorithm should be terminated when A_V^{Train} reaches $\hat{A}$. From now on we assume the expected accuracy $\hat{A}$ is given and $1/N_b < \hat{A} \le 1$.

B. Theoretical Upper Bound

Next, we try to estimate H given that $A_V^{Train} = \hat{A}$. In the following part, we find the maximum and the minimum of H. Let $p_j = |\mathbf{V}_j^{ab}| / |\mathbf{V}^{ab}|$ and $p_{jk} = \mathbf{V}_{jk}^{ab} / |\mathbf{V}_j^{ab}|$. The accuracy of node j on the training set produced by using the entries of matrix $\mathbf{V}^{ab}$ can be expressed as $A_j = \max_k p_{jk}$. The maximum problem can be described as:

$$\begin{aligned}
&\text{Maximize} && H = -\sum_{j=1}^{N_a}\left(p_j \sum_{k=1}^{N_b} p_{jk}\log_2 p_{jk}\right) \\
&\text{Subject to} && \sum_{j=1}^{N_a} p_j = 1 \\
& && \sum_{k=1}^{N_b} p_{jk} = 1 \qquad \text{for } j = 1,2,\ldots N_a \\
& && \sum_{j=1}^{N_a}\left(p_j \max_k p_{jk}\right) = \hat{A} \\
& && 0 \le p_j \le 1 \qquad \text{for } j = 1,2,\ldots N_a \\
& && 0 \le p_{jk} \le 1 \qquad \text{for all } j \text{ and all } k
\end{aligned}$$

To simplify the problem, we can first maximize the entropy of each node given A_j by adjusting p_{jk}, and then maximize H by adjusting p_j and A_j (and N_a, if necessary). Without loss of generality, we can assume $p_{j1} = \max_k p_{jk} = A_j$ for all j. Since the function $f(x) = -x\log_2 x$ is strictly concave for x>0,

$$\frac{\sum_{k=2}^{N_b} f(p_{jk})}{N_b - 1} \le f\left(\frac{\sum_{k=2}^{N_b} p_{jk}}{N_b - 1} \right), \text{ i.e.,}$$

$$-\sum_{k=1}^{N_b} p_{jk} \log_2 p_{jk} \le -(1 - A_j) \log_2 \frac{1 - A_j}{N_b - 1}$$

The equality holds if and only if $p_{j2} = p_{j3} = ... = p_{jk} = \frac{1 - A_j}{N_b - 1}$.

In this case, we can simplify H as:

$$\begin{aligned} H &= -\sum_{j=1}^{N_a} p_j \left[A_j \log_2 A_j + (1 - A_j) \log_2 \frac{1 - A_j}{N_b - 1} \right] \\ &= -\sum_{j=1}^{N_a} p_j \left[A_j \log_2 A_j + (1 - A_j) \log_2 (1 - A_j) \right] \\ &+ \sum_{j=1}^{N_a} p_j (1 - A_j) \log_2 (N_b - 1) \\ &= -\sum_{j=1}^{N_a} p_j \left[A_j \log_2 A_j + (1 - A_j) \log_2 (1 - A_j) \right] \\ &+ (1 - \hat{A}) \log_2 (N_b - 1) \end{aligned}$$

Therefore, the problem reduces to:

$$\begin{aligned} \text{Maximize} \quad & H = -\sum_{j=1}^{N_a} p_j \left[A_j \log_2 A_j + (1 - A_j) \log_2 (1 - A_j) \right] \\ & \quad + (1 - \hat{A}) \log_2 (N_b - 1) \\ \text{Subject to} \quad & \sum_{j=1}^{N_a} p_j = 1 \\ & \sum_{j=1}^{N_a} p_j A_j = \hat{A} \\ & 0 \le p_j \le 1 \quad \text{for } j = 1,2,...N_a \\ & \frac{1}{N_b} \le A_j \le 1 \quad \text{for } j = 1,2,...N_a \end{aligned}$$

Using again the concavity of $f(x) = -x \log_2 x$, we have:

$$\sum_{j=1}^{N_a} p_j f(A_j) \le f\left(\sum_{j=1}^{N_a} p_j A_j \right)$$

$$\sum_{j=1}^{N_a} p_j f(1 - A_j) \le f\left(\sum_{j=1}^{N_a} p_j (1 - A_j) \right)$$

Thus, $H \le -\hat{A} \log_2 \hat{A} - (1 - \hat{A}) \log_2 (1 - \hat{A}) + (1 - \hat{A}) \log_2 (N_b - 1)$

The equality holds if and only if $A_j = \hat{A}$ for all *j*. Thus, we get the following theoretical upper bound for *H*:

$$H_U = -\hat{A} \log_2 \hat{A} - (1 - \hat{A}) \log_2 \frac{1 - \hat{A}}{N_b - 1}$$

C. Theoretical Lower Bound

Following the same approach used to derive the theoretical upper bound for *H*, we can extract the theoretical lower bound for *H*. We first minimize $-\sum_{k=1}^{N_b} p_{jk} \log_2 p_{jk}$ subject to $A_j = p_{j1} \ge p_{j2} \ge p_{j3} = ... = p_{jk} \ge 0$ and $\sum p_{jk} = 1$. Based on the concavity of the function $f(x) = -x \log_2 x$, we know that $-\sum_{k=1}^{N_b} p_{jk} \log_2 p_{jk}$ is minimized when $p_{j1} = p_{j2} ... = p_{jn} = A_j$ and $p_{jn+1} = 1 - nA_j$, where $n = \lfloor 1/A_j \rfloor$. If two p_{jk1} and p_{jk2} are less than A_j, we can construct an example with $p'_{jk1} = \min(A_j, p_{jk1} + p_{jk2})$ and $p'_{jk2} = p_{jk1} + p_{jk2} - p'_{jk1}$ which leads to lower entropy. Therefore, the minimum of $-\sum_{k=1}^{N_b} p_{jk} \log_2 p_{jk}$ is $-nA_j \log_2 A_j - (1 - nA_j) \log_2 (1 - nA_j)$. The floor function in the expression of *n*, however, makes our analysis somehow difficult since it is not continuous. Note that $-nA_j \log_2 A_j - (1 - nA_j) \log_2 (1 - nA_j) \ge -\log_2 A_j$ (the equality holds if and only if $1/A_j$ is an integer). We use $-\log_2 A_j$ as the lower bound for convenience. The problem becomes:

$$\begin{aligned} \text{Minimize} \quad & H = -\sum_{j=1}^{N_a} p_j \log_2 A_j \\ \text{Subject to} \quad & \sum_{j=1}^{N_a} p_j = 1 \\ & \sum_{j=1}^{N_a} p_j A_j = \hat{A} \\ & 0 \le p_j \le 1 \quad \text{for } j = 1,2,...N_a \\ & \frac{1}{N_b} \le A_j \le 1 \quad \text{for } j = 1,2,...N_a \end{aligned}$$

The function $g(x) = -\log_2 x$ is strictly convex for $x>0$. Hence,

$$H = \sum_{j=1}^{N_a} p_j g(A_j) \ge g\left(\sum_{j=1}^{N_a} p_j A_j \right) = g(\hat{A}) = -\log_2 \hat{A},$$

where the equality holds if and only if $A_j = \hat{A}$ for all *j*. We, therefore, obtain the theoretical lower bound of *H*:

$$H_L = -\log_2 \hat{A}$$

D. Typical Case 1

In most cases, both the theoretical upper bound and lower bound are far from the actual value of *H*, since they require that many constraints must be met, as shown above. Here we just consider two typical cases to estimate *H*.

In the first case, the accuracies of all the categories are either 1 or $1/N_b$. Thus,

$$H = -\sum_{j=1}^{K} p_j \log_2 N_b$$

$$\sum_{j=1}^{N_a} p_j = 1$$

$$\sum_{j=1}^{K} p_j \frac{1}{N_b} + \sum_{j=K+1}^{N_b} p_j = \hat{A}$$

It is not difficult to solve the above equations and find the corresponding *H* value. We denote the resulting *H* value by H_{E1} and we are providing it below.

$$H_{E1} = \frac{N_b (1 - \hat{A})}{N_b - 1} \log_2 N_b$$

E. Typical Case 2

In the second case, $A_j = \hat{A}$ for all *j*, and $p_{jk} = p^{k-1} \hat{A}$, where *p* is a constant in the interval $[0,1)$ (it cannot be one because $\hat{A} > 1/N_b$) and satisfies $\sum p_{jk} = 1$, i.e., $1 - p = \hat{A}(1 - p^{N_b})$. This means all the class fractions make a geometric progress. Solving for *p* is not difficult: when $2 \le N_b \le 5$, we can solve this equation analytically; when $N_b > 5$ and $\hat{A} \ge 0.5$, $p \approx 1 - \hat{A}$; when $N_b > 5$ and $\hat{A} < 0.5$, we can solve the equation numerically, which is not difficult since $(1 - p)/(1 - p^{N_b})$ is monotonically increasing in *p*.

In this case, the resulting *H* value, designated by H_{E2}, can be computed as follows:

$$\begin{aligned}
H_{E2} &= -\sum_{k=1}^{N_b} p^{k-1}\hat{A}\log_2\left(p^{k-1}\hat{A}\right)\\
&= -\sum_{i=0}^{N_b-1} p^{i}\hat{A}\log_2\left(p^{i}\hat{A}\right)\\
&= -\sum_{i=0}^{N_b-1} p^{i}\hat{A}\left(i\log_2 p+\log_2 \hat{A}\right)\\
&= -\hat{A}(\log_2 p)\sum_{i=0}^{N_b-1} ip^{i} - \hat{A}\left(\log_2 \hat{A}\right)\sum_{i=0}^{N_b-1} p^{i}\\
&= -\hat{A}(\log_2 p)\frac{p-N_b p^{N_b}+(N_b-1)p^{N_b+1}}{(1-p)^2} - \hat{A}\left(\log_2 \hat{A}\right)\frac{1-p^{N_b}}{1-p}
\end{aligned}$$

Since $1-p=\hat{A}\left(1-p^{N_b}\right)$, it is not difficult to simplify the above expression to obtain:

$$H_{E2} = -\frac{N_b\left(1-\hat{A}\right)-\left(N_b-1\right)p}{1-p}\log_2 p - \log_2 \hat{A}$$

Acknowledgment

This work was supported in part by a National Science Foundation (NSF) grant CRCD: 0203446. Georgios Anagnostopoulos and Michael Georgiopoulos acknowledge the partial support from the NSF grant CCLI 0341601.

References

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

[2] G. A. Carpenter et al., "Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multi-dimensional maps," *IEEE Trans. Neural Networks*, vol. 3, no. 5, pp. 698-713, 1992.

[3] E. Gomez-Sanchez et al., "Safe-μARTMAP: a new solution for reducing category proliferation in Fuzzy ARTMAP," in *Proc. IEEE Int. Joint Conf. Neural Networks*, vol. 2, July 2001, pp. 1197-1202.

[4] G. C. Anagnostopoulos, and M. Georgiopoulos, "Ellipsoid ART and ARTMAP for incremental clustering and classification," in *Proc. IEEE-INNS Int. Joint Conf. Neural Networks*, July 2001, pp. 1221-1226.

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

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

[7] G. C. Anagnostopoulos et al., "Exemplar-based pattern recognition via semi-supervised learning," in *Proc. IEEE Int. Joint Conf. Neural Networks*, July 2003, vol. 4, pp. 2782-2787.

[8] E. Gomez-Sanchez et al., "μARTMAP: use of mutual information for category reduction in Fuzzy ARTMAP", *IEEE Trans. Neural Networks*, vol. 13, no. 1, pp. 58-69, 2002.

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

[10] S. Verzi, "Rademacher penalization applied to Fuzzy ARTMAP and Boosted ARTMAP," in *Proc. IEEE-INNS Int. Joint Conf. Neural Networks*, July 2001, pp. 1191-1196.