Proceedings of International Joint Conference on Neural Networks, Orlando, Florida, USA, August 12-17, 2007

Analyzing the Fuzzy ARTMAP Matchtracking mechanism with Co-Objective Optimization Theory

José Castro and Michael Georgiopoulos and Jimmy Secretan

Abstract— **In the process of learning a pattern I, the Fuzzy ARTMAP algorithm templates (i.e., the weight vectors corresponding to nodes of its category representation layer) compete for the representation of the given pattern. This competition can induce *matchtracking*: a process that iterates a number of times over the template set searching for a template $\mathbf{w}^*$ of the correct class that best represents the pattern I. In this paper, we analyze the search for a winning template from the perspective of bi-criterion optimization and prove that it is actually a walk along the Pareto front of an appropriately defined co-objective optimization problem. This observation allows us to propose the basis for an implementation variant of Fuzzy ARTMAP that (a) produces exactly the same network as Fuzzy ARTMAP, (b) avoids matchtracking by explicitly keeping track of a subset of the Pareto front, (c) finds the correct template to represent an input pattern through a single pass over the template set and (d) eliminates the need for the Fuzzy ARTMAP parameter ε**

I. Introduction

Fuzzy ARTMAP [2] is a supervised neural network learning algorithm that has many desirable properties. This architecture exhibits incremental on-line learning capabilities, has a learning process that is guaranteed to converge to a solution, possesses a novelty detector feature that recognizes novel inputs (inputs that are significantly different from inputs that the architecture has seen before). It can, under analysis, provide explanations for the answers that it produces, and as a result it can address the neural network opacity problem (i.e., inability to explain the answers that they produce) for which most neural networks have been criticized for [3]. Also, Fuzzy ARTMAP neural networks have the property that they can dynamically increase their size as the learning process progresses and only when the learning task at hand requires it, a characteristic that eliminates the need to specify an arbitrary neural network architecture prior to the initiation of the learning process.

Fuzzy ARTMAP can be used both for classification as well as function approximation problems but it has been almost exclusively used for classification problems. Kasuba [5], with only classification problems in mind, developed a simplified Fuzzy ARTMAP structure (called simplified Fuzzy ARTMAP) that is faster than the original. Furthermore, Taghi, et al., in [6], describe variants of simplified Fuzzy ARTMAP, called Fast Simplified Fuzzy ARTMAP, that reduce some of the redundancies of Simplified Fuzzy ARTMAP and speed up its convergence to a solution, even further. One of the Fuzzy ARTMAP fast algorithmic variants, presented in [6], is called SFAM2.0 and it has the same functionality as Fuzzy ARTMAP [2] for classification problems. In this article we are interested in the behavior of Fuzzy ARTMAP's matchtracking mechanism, but this mechanism is used in most of ARTMAP's descendants, of which Fuzzy ARTMAP is one. Nevertheless, in order to put the marchtracking mechanism in a concrete context we will only use Taghi's SFAM2.0 as the basis of our analysis. From now on, we will refer to SFAM2.0 as FS-FAM, and occasionally as Fuzzy ARTMAP. Keeping this in mind, the results presented here are applicable to any neural network that uses the matchtracking mechanism.

More to the point, when Fuzzy ARTMAP is in the process of learning an input pattern $\mathbf{I}$, the templates in its category representation layer compete to represent this pattern, and eventually one of these templates learns the pattern. This competition can induce *matchtracking*: a mechanism that iterates over the template set searching for a template $\mathbf{w}^*$ of the correct class that best represents the input pattern $\mathbf{I}$. It is our experience that this matchtracking induced iteration does not have a high computational cost, and therefore, we believe that this is the likely reason why it has not been subject of a rigorous analysis for optimization. Nevertheless, there is theoretical value and insight obtained by analyzing the behavior of matchtracking. Furthermore, there are instances, where matchtracking implemented as an iterative process can become a liability, such as when using a processor pipeline for parallel implementation of Fuzzy ARTMAP or in a distributed sensor network Fuzzy ARTMAP implementation. In this case it is best if the template that is to represent the input pattern is found with a single pass through the Fuzzy ARTMAP template set, but this can only be done if we develop a non-sequential mathematical characterization of the matchtracking mechanism.

The main contribution of this article, proven in section IV, is that finding the winner of the matchtracking competition is actually a walk along the Pareto front of an appropriately defined co-objective optimization problem. This insight allows us to propose in section V an implementation variant of Fuzzy ARTMAP that addresses the previous issues. For one, and most importantly, it leads us to a Fuzzy ARTMAP implementation that discovers the correct template to represent an input pattern through a single pass over the template set (a desirable property when Fuzzy ARTMAP is implemented in a pipeline). As an added bonus, this implementation discovers the correct template to represent

José Castro is with the Computing Research Center, Costa Rica Institute of Technology (phone 011 506 550 2407; fax: 011 506 552 6665, email: jose.r.castro@gmail.com), Michael Georgiopoulos and Jimmy Secretan are with the School of Electrical Engineering and Computer Science, University of Central Florida, Orlando, FL 32816 (phone:407-823-5338; fax:407-823-5835; emails: michaelg@mail.ucf.edu, jimmy@thepublicgrid.org)

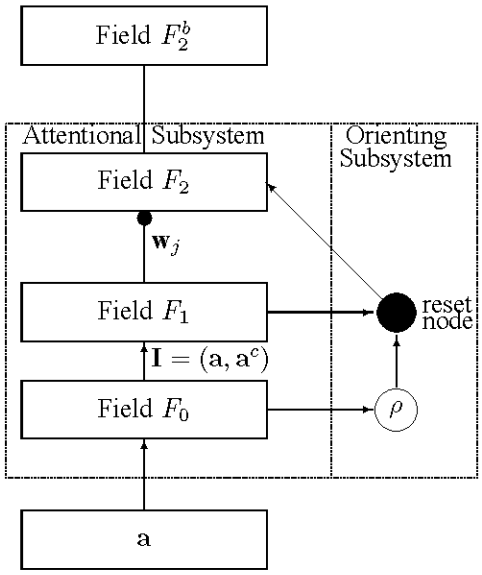

Fig. 1. Block Diagram of the FS-FAM Architecture.

an input pattern without making use of the matchtracking parameter ε of Fuzzy ARTMAP (see main part of the paper for the explanation of ε's utility).

To make the article self contained we present the Fuzzy ARTMAP architecture and FS-FAM algorithm in section II. Special emphasis should be paid to the matchtracking description, presented in subsection II-A, since it is the focus of our work. The basic co-objective optimization theory is outlined in section III and can be found in many optimization textbooks. Our contribution is described in Section IV through an example and appropriate theorems. Conclusive remarks are made in Section V.

II. The Fuzzy-ARTMAP Neural Network Architecture

A block diagram of the FS-FAM components relevant to our discussion is shown in figure 1. This architecture has three major layers. The *input layer* (F_1) where the input patterns (designated by $\mathbf{I}$) are presented; the *category representation layer* (F_2), where compressed representations of these input patterns are formed (designated as $\mathbf{w}_j$); and the *output layer* (F_2^b) that holds the labels of the categories formed in the category representation layer. All input patterns $\mathbf{I}$ presented at the input layer (F_1) of FS-FAM have the following form:

$$\mathbf{I} = (\mathbf{a}, \mathbf{a}^c) = (a_1, a_2, \ldots, a_M, a_1^c, a_2^c, \ldots, a_M^c)$$

where,

$$a_i^c = 1 - a_i;\ \forall i \in \{1, 2, \ldots, M\}$$

The assumption here is that the input vector $\mathbf{a}$ is such that each one of its components lies in the interval [0, 1]. The above operation that creates $\mathbf{I}$ from $\mathbf{a}$ is called *complementary coding* and it is required for the successful operation of Fuzzy ARTMAP.

FS-FAM can operate in two distinct phases: the *training phase* and the *performance phase*. In this paper we are only interested in the training phase of FS-FAM, and we omit any references to its performance phase. The training phase of FS-FAM can be described as follows: Given a set of PT inputs and associated labels pairs, $\{(\mathbf{I}^1, label(\mathbf{I}^1)), \ldots, (\mathbf{I}^{PT}, label(\mathbf{I}^{PT}))\}$, we want to train FS-FAM to map every input pattern of the training set to its corresponding label. To achieve this goal we present the training set to the FS-FAM architecture repeatedly. That is, we present $\mathbf{I}^1$ to F_1, *label*($\mathbf{I}^1$) to F_2^b, $\mathbf{I}^2$ to F_1, *label*($\mathbf{I}^2$) to F_2^b, and finally $\mathbf{I}^{PT}$ to F_1, and *label*($\mathbf{I}^{PT}$) to F_2^b. We present the training set to FS-FAM as many times as it is necessary for FS-FAM to correctly classify all these input patterns. The task is considered accomplished (i.e., the learning is complete) when no new weights are created and the weights do not change during a training set presentation. This training scenario is called *off-line learning*. There is another training scenario, the one considered in this paper, that is called *on-line training*, where each one of the input/label pairs are presented to FS-FAM only once. FS-FAM's training phase is described in Taghi's et al., paper [6], and repeated below.

1) Find the nearest category in the category representation layer of FS-FAM that "resonates" with the input pattern.
2) If the labels of the chosen category and the input pattern match, update the chosen category to be closer to the input pattern.
3) Otherwise, we reset the winner, temporarily increase the resonance threshold (called *vigilance parameter*), and try the next winner. This process is called *matchtracking*.
4) If the winner is uncommitted, create a new category (assign the representative of the category to be equal to the input pattern, and designate the label of the new category to be equal to the label of the input pattern).

The nearest category to an input pattern $\mathbf{I}$ presented to FS-FAM is determined by finding the category that maximizes the function:

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

This equation introduces two operands, one of them is the *fuzzy min operand*, and designated by the symbol $\wedge$. The fuzzy min operation of two vectors $\mathbf{x}$, and $\mathbf{y}$, designated as $\mathbf{x} \wedge \mathbf{y}$, is a vector whose components are equal to the minimum of components of $\mathbf{x}$ and $\mathbf{y}$. The other operand introduced is designated by the symbol $|\cdot|$. In particular, $|\mathbf{x}|$ is the size of a vector $\mathbf{x}$ and is defined to be the sum of its components. The above function is called the *bottom-up input* (or choice function value, or activation value) pertaining to the F_2 node j with category representation (template) equal to the vector $\mathbf{w}_j$, due to the presentation of input pattern $\mathbf{I}$. This function obviously depends on an FS-FAM network parameter α, called *choice parameter*, that assumes values in the interval $(0, \infty)$. In most simulations of Fuzzy ARTMAP the useful range of α is the interval $(0, 10]$.

The resonance of a category to a given input pattern $\mathbf{I}$ is determined by examining if the function, called *vigilance ratio*, and defined below

$$\rho(\mathbf{I}, \mathbf{w}_j) = \frac{|\mathbf{I} \wedge \mathbf{w}_j|}{|\mathbf{I}|} \tag{2}$$

satisfies the following condition:

$$\rho(\mathbf{I}, \mathbf{w}_j) \geq \rho \tag{3}$$

If the above equation is satisfied we say that resonance is achieved. The parameter ρ is called the *vigilance parameter* and assumes values in the interval $[0, 1]$. As the vigilance parameter increases, more category nodes are created in the category representation layer (F_2) of Fuzzy ARTMAP. At the beginning of training with an input, output label pair the value of ρ is set equal to the value of the baseline vigilance, $\bar{\rho}$, which is a value chosen in the interval $[0, 1]$. If the label of the input pattern ($\mathbf{I}$) is the same as the label of the resonating category, then the category's template ($\mathbf{w}_j$) is updated to incorporate the features of this new input pattern ($\mathbf{I}$). The update of a category's template ($\mathbf{w}_j$) is performed as depicted below:

$$\mathbf{w}_j \leftarrow \mathbf{w}_j \wedge \mathbf{I} \tag{4}$$

The update of templates, illustrated by the above equation, has been called *fast-learning* in Fuzzy ARTMAP. Our paper is concerned only with the fast learning FS-FAM, although using slow learning as found in [2] would not compromise the algorithms proposed in this paper.

At the beginning of the FS-FAM training the choice parameter (chosen in $(0, 10)$), the baseline vigilance parameter (chosen in $[0, 1]$) and the uncommitted node $\mathbf{w}_0$ (chosen to be equal to an all ones vector) are initialized. In FS-FAM, $\mathbf{w}_0$ is the only template that exists when training commences. Given it's value (all ones vector) it is guaranteed to pass vigilance. Also, a new template $\mathbf{w}$ equal to the input pattern $\mathbf{I}$ will be created whenever the uncommitted node $\mathbf{w}_0$ wins the activation competition. In FS-FAM, the uncommitted node $\mathbf{w}_0$ is never eliminated or changed and is used as a placeholder to identify when to create new templates.

A. The Matchtracking mechanism

When the situation for a given input pattern is that the category template $\mathbf{w}_j$ is chosen as the winner, it is not the uncommitted node $\mathbf{w}_0$, and the label of this template $\mathbf{w}_j$ is different than the label of the input pattern $\mathbf{I}$, then this template is reset and the vigilance parameter ρ is increased to the level:

$$\rho \leftarrow \rho(\mathbf{I}, \mathbf{w}_j) + \varepsilon \tag{5}$$

In the above equation ε takes very small values. Increasing the value of vigilance as shown in equation 5 guarantees that in the next activation competition the last template winner $\mathbf{w}_j$ is excluded from competition. It is difficult to correctly set the value of ε so as to guarantee that after category resets no legitimate category templates are missed by FS-FAM. Nevertheless, in practice, typical values of the parameter ε are taken from the interval $[0.00001, 0.001]$. After the reset of

FAM–LEARNING$(\{\mathbf{I}^1, \ldots, \mathbf{I}^{PT}\}; \bar{\rho}, \alpha, \varepsilon)$

```
1  w_0 ← (1, 1, ..., 1)   [2M]
2  templates ← {w_0}
3  for each I in {I^1, I^2, ..., I^PT}
4  do ρ ← ρ̄
5     repeat
6           S ← {w_j : ρ(I, w_j) ≥ ρ}
7           j_max ← maxarg_j {T(I, w_j, α) : w_j ∈ S}
8           if label(I) ≠ label(w_jmax)
9             then ρ ← ρ(I, w_jmax) + ε
10    until (w_jmax = w_0) or (label(I) = label(w_jmax))
11    if w_jmax ≠ w_0
12      then
13           w_jmax ← w_jmax ∧ I
14      else
15           templates ← templates ∪ {I}
16 return templates
```

Fig. 2. FS-FAM on-line training phase algorithm

template j (if that's the case), other templates are searched to find the one that maximizes the bottom-up input while satisfying the vigilance constraint. This process continues until the uncommitted node $\mathbf{w}_0$ wins or a category template is found that maximizes the bottom-up input, satisfies vigilance and has the same label as the input pattern presented to FS-FAM. Once this happens, the corresponding creation of a new template or update of the template as indicated by equation (4) ensues.

This iterative process of increasing the ρ parameter and searching again for a competition winner is called match-tracking and will be the segment of the FS-FAM algorithm that will interest us in our analysis.

Pseudocode for the FS-FAM algorithm (on-line training phase) is shown in Figure 2. In the on-line training phase of the network the learning process (lines 3–15) passes through the data once. We present the on-line version because it is simpler and the iteration over the training set is irrelevant for the analysis of the matchtracking mechanism. The match-tracking mechanism itself is presented in lines 5–10.

III. PARETO FRONTS AND CO-OBJECTIVE OPTIMIZATION

In the multi-objective optimization (MOP) model of interest to us we will be dealing with a vector of decision variables $\mathbf{x} = (x_1, \ldots, x_n)$ that optimize the vector $\mathbf{y}$, comprised from a series of objective functions f_i, as shown below

$$\mathbf{f}(\mathbf{x}) = \mathbf{y} = (f_1(\mathbf{x}), \ldots, f_k(\mathbf{x}))$$

subject to a set of general inequality constraints

$$g_i(\mathbf{x}) \geq 0, \; i \in \{1, \ldots, p\}$$

The functions f_i form a mathematical description of performance criteria that usually conflict with each other. Therefore *optimizing* the vector means finding a compromise

between the values of the f_i that form an acceptable balance to the decision maker.

Since we now have a number of objective functions to optimize, the notion of optimum, as we usually understand it, has to change. The first to recognize this fact was Francis Ysidro Edgeworth in 1881. This notion was latter generalized by Wilfrido Pareto, and therefore it is now called *Edgeworth–Pareto optimum* or simply Pareto optimum.

To introduce the Pareto front concept we will start with a number of definitions. But before we do so it's important to note that in optimization we can *decrease* a cost function or *increase* a gain function. We will take the point of view of increasing a gain (activation) function, since this approach is consistent with FS-FAM's approach of choosing the template achieving the highest activation. Many optimization texts take the opposite approach.

Definition 3.1: A vector $\mathbf{u} = (u_1, \ldots, u_k)$ is said to *(pareto)* dominate another vector $\mathbf{v} = (v_1, \ldots, v_k)$ if and only if $\forall i \in \{1, \ldots k\}$, $u_i \geq v_i$, and $\exists j \in \{1, \ldots, k\}$ such that $u_j > v_j$. If this is the case, we write $\mathbf{u} \succeq \mathbf{v}$.

Definition 3.2: In a MOP problem we say that a point $\mathbf{x} \in \mathcal{F}$, is a *strongly dominated solution* if and only if there does not exist another point $\mathbf{x}'$ such that $f_i(\mathbf{x}) \leq f_i(\mathbf{x}')$ for all i and $f_j(\mathbf{x}) < f_j(\mathbf{x}')$ for some j.

Notice that the definition of a strongly dominated solution is very similar to pareto dominance. The difference being that in pareto dominance we compare the vectors themselves, but in strongly dominated solutions we compare the objective function vectors. This takes us to the formal definition of Pareto optimality, pareto optimal set, and pareto front.

Definition 3.3: A vector of decision variables $\mathbf{x} = (x_1, \ldots, x_n)$ is *Pareto optimal* if there does not exist another $\mathbf{x}' \in \mathcal{F}$, where $\mathcal{F}$ is the feasible region, such that $f_i(\mathbf{x}) \leq f_i(\mathbf{x}')$ for all i, and $f_j(\mathbf{x}) < f_j(\mathbf{x}')$ for at least one j. Or more succinctly stated, there does not exist an $\mathbf{x}'$ such that $\mathbf{f}(\mathbf{x}') \succeq \mathbf{f}(\mathbf{x})$.

This definition basically states that a vector is pareto optimal if it is not dominated by another vector from the feasible region. This definition usually does not produce a single optimal value, giving rise to the following definition.

Definition 3.4: Given a MOP with objective function vector $\mathbf{y} = \mathbf{f}(\mathbf{x})$, the *pareto optimal set* $\mathcal{P}^*$ is defined as:

$$\mathcal{P}^* = \{\mathbf{x} \in \mathcal{F} : \neg\exists \mathbf{x}' \in \mathcal{F} \cdot \ \mathbf{f}(\mathbf{x}') \succeq \mathbf{f}(\mathbf{x})\}$$

Or simply stated as: the set of pareto optimal points.

Definition 3.5: Given a MOP with a designated $\mathcal{P}^*$, the pareto front $\mathcal{PF}^*$ is defined as $\mathbf{f}(\mathcal{P}^*)$, or as

$$\mathcal{PF}^* = \{\mathbf{f}(\mathbf{x}) : \mathbf{x} \in \mathcal{P}^*\}$$

A. Co–Objective Optimization

Co–Objective optimization is a special case of multi-objective optimization where the vector $\mathbf{f}(\mathbf{x}) = (y_1, y_2) = (f_1(\mathbf{x}), f_2(\mathbf{x}))$. Under these circumstances we can easily graph and visualize the co–objective optimization problem and its pareto front. For example, in Figure 3, the pareto front points can be identified by simple observation (i.e., these points do not have other points that dominate them in the way that was stated in definition 3.1). In Figure 3, the pareto front points are designated as open circles, the dashed lines converging to these pareto front points define a region in the objective space that encloses all the points that these pareto front points dominate.

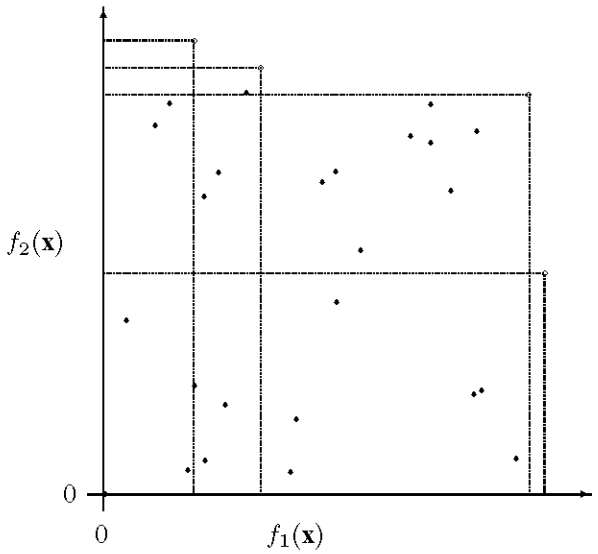

Fig. 3. Plot of co-objective optimization points with pareto front points designated as open circles.

IV. Matchtracking and Pareto Fronts

It is clear from the explanation of the FS-FAM algorithm, and the pseudo-code in Figure 2, that the process of searching for a winning template is NOT co-objective optimization. It is the iterative process of finding:

$$\max_{\mathbf{w}_j \in templates} \{T(\mathbf{I}, \mathbf{w}_j, \alpha)\}$$

subject to the constraint $\rho(\mathbf{I}, \mathbf{w}_j) \geq \rho$. When the maximum is found, and only if this maximum does not belong to the correct class will the process iterate, increasing the value of ρ as presented in equation 5 and thereby excluding the last winning template from the competition.

FS-FAM is more correctly modeled as a single valued optimization problem subject to an inequality constraint. Nevertheless, the iteration produced by matchtracking constantly changes the feasible region of the problem. And it is in the context of the match-tracking mechanism that it becomes valuable to model the FS-FAM optimization as a two dimensional co-objective optimization problem with a fixed feasible region.

Consider the graph in Figure 4. This graph plots the value of vigilance against activation for templates in the benchmark circle within a square square problem, also mentioned in [2]. The circle within a square is a famous benchmark problem where the classifier is trying to discriminate whether a point in the input space is in a circle (located within a square)

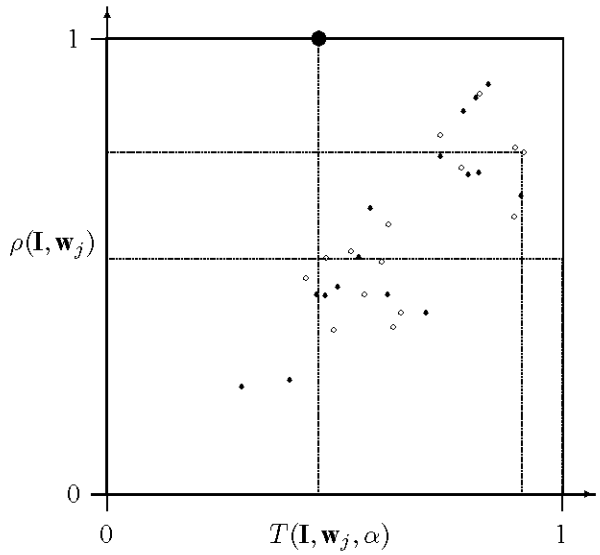

Fig. 4. In this example, the dots (solid and empty) represent templates in the circle within a square problem. The big solid dot represents the uncommitted template. Eligible templates are located in the northeastern quadrant defined by the lines of vigilance equal to 0.5, and activation value of approximately equal to 0.5. In the figure the template identified (empty dot with the dashed lines converging to it) is the template that was first chosen during the presentation of the input pattern $\mathbf{I} = (0.13207, 0.661587)$, which resides inside the circle. The winner template is not of the correct class (i.e., it is template representing patterns that are outside the circle) so matchtracking will ensue.

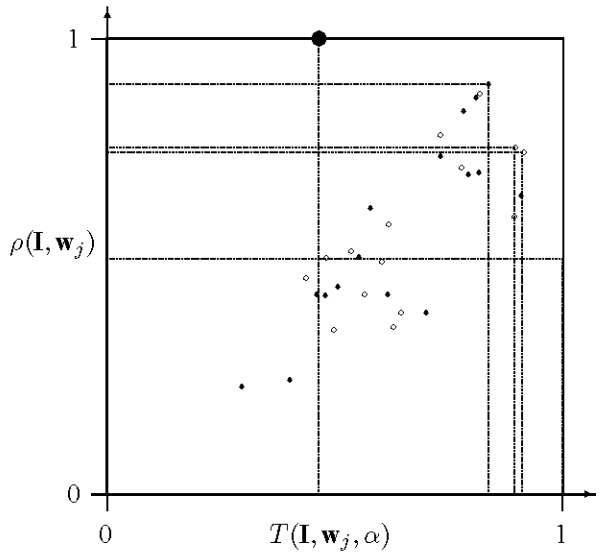

Fig. 5. Scatter plot of the co-objective vector, $\mathbf{f}(\mathbf{w}) = (T(\mathbf{I}, \mathbf{w}, \alpha), \rho(\mathbf{I}, \mathbf{w}))$, values. The $\mathbf{w}$'s correspond to the FS-FAM templates in the circle in the square problem, and the input pattern $\mathbf{I}$, located inside the circle, is equal to $(0.13207, 0.661587)$. In the figure the pareto optimal points (templates) that FS-FAM examines are shown (these points have dashed lines converging to them). Of these templates two are of the incorrect class (empty dots) and one is of the correct class (solid dot) that eventually encodes the input pattern.

or outside the circle; the circle is centered at the center of the square and its area is half of the area of the square. In Figure 4, solid dots are templates that represent input patterns located in the circle and empty dots are templates that represent input patterns located outside of the circle. The large dot at the top of the vertical dotted line represents the uncommitted node, whose vigilance is equal to one but whose activation is less than one half. The horizontal dotted line (at level=0.5) represents the baseline vigilance $\bar{\rho}$, which for this example has been chosen equal to 0.5. Eligible templates (i.e., templates that are likely to code the presented input pattern) are all inside the north-eastern quadrant defined by the dotted lines at $\rho = 0.5$ and $T \approx 0.5$.

It is worth observing from Figure 4 that there is a positive correlation between vigilance values and activation values. This relationship is to be expected (see equations 1 and 2). Within the region of eligible templates (i.e., northeastern quadrant of Figure 4) the winner of the competition will be the template with highest activation $T(\mathbf{I}, \mathbf{w}_j, \alpha)$ value. This template winner is shown with dotted lines leading to it; these lines define a region of points (templates) that are pareto dominated by this template. This template winner is not of the correct class, so matchtracking will happen. This process increases the value of ρ as shown in equation 5, so if ε is small enough, all the templates above the horizontal dotted (marked by the current template winner) line will be the only eligible templates during the second pass through the template set. As this process continues we will eventually obtain a template winner that is of the correct class or we will obtain the uncommitted node as the template winner. In this example, the final winner is of the correct class and it is shown in Figure 5 as a solid dot with dashed lines converging to it.

Notice that the winners of the matchtracking competition, shown in Figure 5, lie in a pareto front of an appropriately defined co-objective optimization problem. The example (in Figures 4 and 5) allows us to visualize the main contributions of our article, presented as a sequence of theorems (Theorem 4.1 and Theorem 4.2):

Theorem 4.1: The winner of FS-FAM competition (lines 6–13 of Figure 2) for a given input pattern $\mathbf{I}$ will always be the uncommitted node or a member of the pareto front of the co-objective optimization problem $\mathbf{f}(\mathbf{w}) = (T(\mathbf{I}, \mathbf{w}, \alpha), \rho(\mathbf{I}, \mathbf{w}))$ with the constraint $\rho(\mathbf{I}, \mathbf{w}) \geq \bar{\rho}$.

Proof: We could make the statement of the theorem more concise by stating that the winner of the competition is a member of the pareto front, since an uncommitted node is on the pareto front. This is true because the vigilance of the uncommitted node for any input pattern is equal to one which is the maximum vigilance value attainable. Hence, it is impossible for another template to pareto dominate the uncommitted node.

Now let us consider the template $\mathbf{w}_j$ that has won the competition loop (lines 6–13 of figure 2). And let us also consider another arbitrary template $\mathbf{w}_k$. Template $\mathbf{w}_k$ either complies or does not comply with the vigilance constraint.

If it doesn't comply then it cannot pareto dominate $\mathbf{w}_j$ since its vigilance is less that that of $\mathbf{w}_j$. If it complies with the vigilance constraint then its activation $T(\mathbf{I}, \mathbf{w}_k) \leq T(\mathbf{I}, \mathbf{w}_j)$ because $\mathbf{w}_j$ won the competition in the presence of $\mathbf{w}_k$; thus, once more, it cannot pareto dominate $\mathbf{w}_j$. Consequently, since an arbitrary template $\mathbf{w}_k$ cannot pareto dominate the winner of the competition in lines 6–13 of Figure 2, then $\mathbf{w}_j$ is pareto optimal. ■

Theorem 4.2: The template winner of FS-FAM for the whole algorithm in Figure 2 will always be the uncommitted node or a member of the pareto front of the co-objective optimization problem $\mathbf{f}(\mathbf{w}) = (T(\mathbf{I}, \mathbf{w}, \alpha), \rho(\mathbf{I}, \mathbf{w}))$ with the constraint $\rho(\mathbf{I}, \mathbf{w}) \geq \bar{\rho}$.

Proof: The proof here is an obvious repetition through the arguments presented in the proof of the previous theorem. If no match-tracking happens, during the presentation of an input pattern, the previous theorem and its proof suffice to prove this theorem. If match-tracking happens the vigilance threshold is increased and the arguments presented before (in Theorem 4.1) that the winner template has to be on the pareto front are still valid. ■

V. Discussion

The previous result can be used to implement a variant of the Fuzzy ARTMAP algorithm that has particular advantages over the original. However, care must be taken to implement this variant in an efficient way. In this variant implementation of Fuzzy ARTMAP the algorithm must keep track of the pareto front points lying in the northeastern quadrant of Figure 4. Although keeping track of the pareto front points might seem expensive computationally, two factors make this process computationally reasonable. For one, there is a positive correlation between activation values and vigilance ratio values which makes the templates *spread out* along the vicinity of the identity line of the graph (see Figure 4). On the contrary, the pareto front, cuts across a curve with a negative slope. This combination produces a small number of points in any pareto front for a Fuzzy ARTMAP problem compared with the total eligible points that pass the baseline vigilance threshold. Secondly, our experience with Fuzzy ARTMAP on a variety of problems is that the number of matchtracking instances for an input pattern, output label pair presentation rarely exceed five (5). Furthermore, it is important to notice that the only templates that we need to keep track of are the members of the pareto front that belong to the same classification category as the presented input pattern, since these templates are the only ones that might eventually be modified by the algorithm. For all the other pareto front templates we simply need to retain their activation and vigilance ratio values, a two-dimensional real-valued vector for each such template. Thus, the amount of storage corresponding to pareto front templates is not as excessive as keeping track of all the template values of FS-FAM.

By keeping track of the aforementioned needed information regarding the pareto front templates we can select the correct FS-FAM template, during an input-pattern/output-label pair presentation, in only one pass through the FS-FAM template set. This has an advantage in certain FS-FAM implementation scenarios. For example, in a related article [4] we propose a pipelined parallel implementation of the FS-FAM algorithm, to speed-up its training phase. This pipelined implementation permitted efficient parallel on-line FS-FAM training with large databases. However, since it is not trivial to parallelize the matchtracking mechanism, a no-match tracking FS-FAM was only considered there (see [4]). To use an analogy, the pipeline structure can be compared to a product assembly line, and the process of matchtracking as the process of returning products back to the starting point of the assembly, a procedure that clearly disrupts the pipeline. Matchtracking is an intrinsically sequential process, and in order to parallelize it a global non sequential characterization of the matchtracking mechanism is needed. The work presented in this paper (pareto front structure of match-tracking) achieves precisely this feat. Furthermore, it is worth mentioning that the results obtained here can be generalized to other ART architectures, such as Ellipsoidal ARTMAP [1], Gaussian ARTMAP (see [7]), that have incorporated the match-tracking mechanism in their design.

As a last note and added bonus, since we know that the winners of the Fuzzy ARTMAP competition lie in the pareto front the need for an ε parameter is removed. The ε parameter is chosen a-priori in Fuzzy ARTMAP, but its a-priori setting might lead into problems such as omitting eligible templates or even ending up with an out of bounds value of ρ ($\rho > 1$)!

Acknowledgment

Jimmy Secretan acknowledges the support of the NSF Graduate Fellowship program. Michael Georgiopoulos acknowledges the partial support from the following NSF grants: NSF CRCD: 0203446, NSF CCLI 0341601, and NSF DUE 05254209.

References

[1] G. C. Anagnostopoulos and M. Georgiopoulos, "Ellipsoid ART and ARTMAP for incremental unsupervised and supervised learning," in *Proceedings of the IEEE–INNS–ENNS*, vol. 2, International Joint Conference on Neural Networks. Washington DC: IEEE–INNS–ENNS, 2001, pp. 1221–1226.

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

[3] G. A. Carpenter and A. H. Tan, "Rule extraction: From neural network architecture to symbolic representation," *Connection Science*, vol. 7, no. 1, pp. 3–27, 1995.

[4] J. Castro, M. Georgiopoulos, J. Secretan, R. DeMara, G. Anagnostopoulos, and A. Gonzalez, "Pipelining of fuzzy artmap without matchtracking: Correctness, performance bound, and beowulf evaluation," *Neural Networks Journal*, vol. 20, no. 1, pp. 109–128, January 2006. [Online]. Available: http://dx.doi.org/10.1016/j.neunet.2006.10.003

[5] T. Kasuba, "Simplified Fuzzy ARTMAP," *AI Expert*, pp. 18–25, November 1993.

[6] M. Taghi, V. Baghmisheh, and N. Pavesic, "A fast simplified fuzzy artmap network," *Neural Processing Letters*, vol. 17, pp. 273–316, 2003.

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