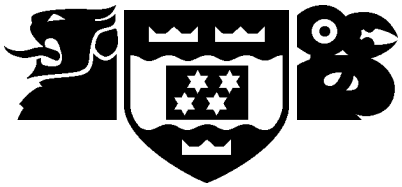
VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui



Department of Computer Science

PO Box 600
Wellington
New Zealand

Tel: +64 4 471 5328
Fax: +64 4 495 5232
Internet: Tech.Reports@comp.vuw.ac.nz

The Match Tracking Anomaly and its Minimisation in the Fuzzy ARTMAP Neural Network

G. Bartfai

Technical Report CS-TR-96/3
February 1996

Abstract

This paper looks at the anomalous situation that can arise in the ARTMAP and Fuzzy ARTMAP neural networks — as a result of their "match tracking" process — whereby a current input–target association cannot be learned because the network is unable to find an alternative category for an input pattern that matches its selected category prototype perfectly. We show that this Match Tracking Anomaly (MTA) cannot be avoided in the Fuzzy ARTMAP network if its ART_b vigilance level is less than 1. To alleviate this problem, we propose two improvements to the Fuzzy ARTMAP learning algorithm. One of them is concerned with the timing according to which input patterns and corresponding target outputs are processed by the network. The other one is the explicit overwriting of an existing association between an input and an output category in case the input is matched perfectly and yet the network's prediction is wrong. Both of these modifications are needed to reduce the occurrence of MTA during learning, and eliminate it altogether in a trained network (on a finite training set). As a result, training time is also reduced, which is demonstrated through the performance of the network on a machine learning benchmark database. This paper expands on the results presented in [3].

Keywords: Adaptive Resonance Theory, Fuzzy ARTMAP, Match Tracking Anomaly, Zoo database.

Publishing Information

This paper will appear in the *Australian Journal of Intelligent Information Processing Systems*.

Author Information

Guszti Bartfai is with the Department of Computer Science.
His e-mail address is guszti@comp.vuw.ac.nz .

1 Introduction

The *Fuzzy ARTMAP* neural network [6] implements a supervised learning mechanism capable of self-organising stable recognition categories in response to arbitrary sequences of analogue and binary input patterns. It offers a unique solution to the *stability-plasticity dilemma* that autonomous learning systems have to face, and overcomes the problems of long training and "catastrophic forgetting" associated with many popular networks such as back-propagation [10]. The Fuzzy ARTMAP network has been applied to several machine learning benchmark problems [6] as well as real-world tasks (for example [1]), and compared favourably with other neural networks and traditional AI machine learning algorithms.

Under certain conditions, however, anomalous situations can occur in the network during training as a result of its internal "match tracking" process. As a result, the network is unable to learn a new input-target association (see details in section 3). This was first reported in [7] for the ARTMAP network, but it was also claimed that this anomaly would never arise if the (binary) input patterns have the same number of 1's [7, pages 584-585]. We showed in [4] that this claim holds only if the *vigilance* parameter of the ART_b module (ρ_b, see in section 2) is set to 1. Although $\rho_b = 1$ is a typical setting in pattern recognition, there can be applications where $\rho_b < 1$ is needed (see [2] for example). In such situation, we have shown that the anomaly can still be eliminated if the presentation of input-target pairs satisfies certain timing conditions [4].

A distinctive advantage of the Fuzzy ARTMAP network over ARTMAP is that it is capable of operating in slow learning mode. This difference, however, causes the above anomaly to re-emerge even if the timing conditions above needed for ARTMAP are satisfied. Consequently, the Fuzzy ARTMAP learning algorithm needs to be modified to cope with this kind of problem.

This paper looks at the *Match Tracking Anomaly* (*MTA*) in the Fuzzy ARTMAP network, and introduces two improvements to its learning algorithm: one that eliminates the *match tracking anomaly* (or *MTA*) when the network is in fast learning mode, and another that is needed to initially correct and eventually eliminate the *MTA* — on a finite training set — in slow learning mode. The effect of these improvements are demonstrated when the network is trained on a machine learning benchmark database.

Expanding on the results presented in [3], some further experiments were also carried out in order to gain insight into the joint effect of two parameters (learning rate and choice) on the original and improved Fuzzy ARTMAP learning algorithms.

Section 2 introduces the Fuzzy ARTMAP network at a level that is necessary for understanding the main results of this paper. Section 3 deals with the Match Tracking Anomaly in the ARTMAP and Fuzzy ARTMAP networks. In section 4, the two improvements to the standard Fuzzy ARTMAP learning algorithm are described. The experiments are discussed and the results are presented in section 5. After some discussion in section 6, conclusions are drawn in section 7.

2 The Fuzzy ARTMAP network

The Fuzzy ARTMAP neural network architecture is capable of incremental learning of multi-dimensional mappings in response to arbitrary sequences of analogue and binary input/target vector pairs [6]. The vectors can represent fuzzy or crisp sets of features. The architecture of the network (see figure 1) is identical to that of an ARTMAP network [7], except that the binary ART1 [5] modules are replaced with Fuzzy ART [8] ones.

A Fuzzy ART network performs unsupervised clustering on analogue and binary input patterns. It has three layers: the *input* layer ($F0$), the *comparison* layer ($F1$), and the

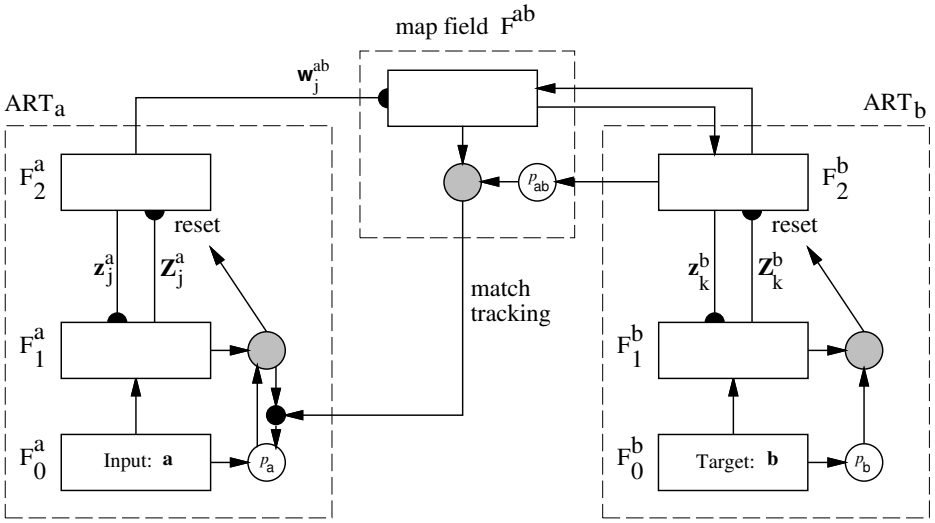


Figure 1: Architecture of the Fuzzy ARTMAP network. Note that the bottom-up weights $\mathbf{Z}_j^a$ and $\mathbf{Z}_k^b$ are not used explicitly in Fuzzy ART, but are included in (1) implicitly.

recognition layer ($F2$), with M, M and N neurons, respectively (see modules ART_a and ART_b in figure 1). The neurons, or nodes, in the F2 layer represent input categories. Each F2 node j stores a prototype vector $\mathbf{z}_j$ for the category it represents.

Upon presentation of an input vector $\mathbf{x}$ ($x_i \in [0,1], i = 1,2,\ldots,M$), the network attempts to classify it into one of its existing categories based on its similarity to the stored prototype of each category node. More precisely, for each node j in the F2 layer, the *choice function*

$$T_j(\mathbf{x}) = \frac{|\mathbf{x} \wedge \mathbf{z}_j|}{\beta + |\mathbf{z}_j|} \tag{1}$$

is calculated, where $\beta > 0$ is the *choice parameter*, the fuzzy AND [11] operator $\wedge$ is defined by

$$(\mathbf{p} \wedge \mathbf{q})_i \equiv min(p_i, q_i) \ \ ,$$

and the norm $|\cdot|$ is defined by

$$|\mathbf{p}| \equiv \sum_{i=1}^{M} p_i$$

for any M-dimensional vector $\mathbf{p}$.

The network then makes a category choice by selecting the F2 node J where

$$T_J = \max\{T_j : j = 1,2,\ldots,N\} \ .$$

The F1 layer activity vector then becomes $\mathbf{x} \wedge \mathbf{z}_J$ and is compared to the current input vector $\mathbf{x}$: the strength of the match is given by

$$\frac{|\mathbf{x} \wedge \mathbf{z}_J|}{|\mathbf{x}|} \ , \tag{2}$$

which is compared with a system parameter ρ called *vigilance* ($0 \leq \rho \leq 1$). If the input matches sufficiently, i.e. the match strength $\geq \rho$, then it is assigned to F2 node J and the network learns it by adjusting $\mathbf{z}_J$ according to the equation

$$\mathbf{z}_J^{(new)} = \eta(\mathbf{x} \wedge \mathbf{z}_J^{(old)}) + (1 - \eta)\mathbf{z}_J^{(old)} \tag{3}$$

where η is the *learning rate* ($0 \leq \eta \leq 1$).

If the stored prototype $\mathbf{z}_J$ does not match the input sufficiently (match strength $< \rho$), the winning F2 node J is reset for the period of presentation of the current input. Then another F2 node (or category) will be selected whose prototype will be matched against the input. This "hypothesis-testing" cycle is repeated until the network either finds a stored category whose prototype matches the input closely enough, or allocates a new F2 node. Then learning takes place as described above.

After an initial period of self-stabilisation, the network will directly (i.e. without search) access the prototype of one of the categories it has found in a given training set. The higher the vigilance level, the larger number of smaller, or more specific, categories will be created. (If $\rho = 1$, the network will assign a new category for every unique input pattern.)

The two Fuzzy ART modules in a Fuzzy ARTMAP network are linked together by an "inter-ART" associative memory called *map field* (F^{ab}). Module ART_a (with a *baseline* vigilance $\overline{\rho_a}$) learns to categorise input patterns $\mathbf{a}^{(i)}$ presented at layer F_0^a, while module ART_b (with vigilance ρ_b) develops categories of target patterns $\mathbf{b}^{(i)}$ presented at layer F_0^b. Modules F_2^a and F^{ab} are connected via associative links (w_{jk}^{ab}) whose strengths are adjusted through learning (see below). There are one-to-one, two-way, and non-modifiable connections between nodes in the F^{ab} and F_2^b layers, i.e. each F_2^b node is connected to its corresponding F^{ab} node, and vice versa. A new association between an ART_a category J and an ART_b category K is learned by setting the corresponding $F_2^a \rightarrow F^{ab}$ link (or connection weight) to one and all other links from the same ART_a node to zero, i.e.

$$w_{jk}^{ab} = \begin{cases} 1 & \text{if } j = J \text{ and } k = K \\ 0 & \text{otherwise} \quad . \end{cases} \tag{4}$$

When an input pattern is presented to the network, the F^{ab} layer will receive inputs from both the ART_a module ($\mathbf{w}_J^{ab}$ vector through the previously learned $J \rightarrow K$ associative link) and the ART_b module (F_2^b output vector $\mathbf{y}^b$ with the winning F_2^b node set to one). If the two F^{ab} inputs match, i.e. the network's prediction is confirmed by the selected target category, the network will learn by modifying the prototypes of the chosen ART_a and ART_b categories according to the Fuzzy ART learning equations shown above. If there is a mismatch at the F^{ab} layer, a map field reset signal will be generated, and a process called *match tracking* will start whereby the baseline vigilance level of the ART_a module will be raised by the minimal amount needed to cause mismatch with the current ART_a input at the F_1^a layer. This will subsequently trigger a search for another ART_a category, whose prediction will be matched against the current ART_b category at the F^{ab} layer again. This process continues until the the network either finds an ART_a category that predicts the category of the current target correctly, or creates a new F_2^a node and a corresponding link in the map field, which will learn the current input/target pair according to (4). The ART_a vigilance is then allowed to return to its resting level ($\overline{\rho_a}$).

After a few presentations of the entire training set, the network will self-stabilise, and will read out the expected output for each input without search.

Note that if the Fuzzy ARTMAP network is trained on binary patterns only, then the $\wedge$ operator in (1), (2) and (4) will be equivalent to $\cap$, the binary intersection operator. In this

case, the Fuzzy ARTMAP network in *fast learning* mode ($\eta = 1$) will be equivalent to the ARTMAP network.

3 The Match Tracking Anomaly in the Fuzzy ARTMAP network

The match tracking anomaly (a term that was introduced in [4]) was first discussed by Carpenter et al. in relation to the ARTMAP network [7]. Since the Fuzzy ARTMAP network is equivalent to ARTMAP in fast learning mode ($\eta = 1$) when both are trained on binary patterns, it is instructive to look at the *MTA* in the ARTMAP network first, and then discuss the implications of slow learning in Fuzzy ARTMAP.

The ARTMAP network was originally introduced for supervised learning of *classification* of arbitrary sequences of input patterns. Carpenter et al. in [8] noted that the following anomalous situation can arise in the network as a result of its match tracking process. Suppose ART_a input vector $\mathbf{a}^{(i)}$ is perfectly coded by category J, i.e. $\mathbf{z}_J^a = \mathbf{a}^{(i)}$. (This is the case, for example, when category J is created, or node J becomes committed, while input $\mathbf{a}^{(i)}$ is presented.) Furthermore, the network has learned to associate ART_b category K with ART_a category J. Suppose the prototype vector $\mathbf{z}_K^b$ of category K is subsequently recoded during further training such that it no longer matches target $\mathbf{b}^{(i)}$ enough at the current $\rho_b < 1$ vigilance level, but $\mathbf{z}_J^a$ remains unchanged. So when the $(\mathbf{a}^{(i)}, \mathbf{b}^{(i)})$ pair is presented again, ART_b will select another category that will cause category mismatch at the map field with the network's prediction of category K. This will trigger the match tracking process, which will raise ρ_a above 1 since prototype $\mathbf{z}_J^a$ matches input $\mathbf{a}^{(i)}$ perfectly. The network will then be unable to find another ART_a category that would match the input any better. In such cases, the network will be shut down for the duration of the current input/target pair [8]. It was claimed in [8] that the *MTA* will not arise if the ART_a inputs have the same number of 1's. (The same number of 1's in the input can be guaranteed by representing input patterns in *complement coding* [8], according to which both input $\mathbf{a}$ and its complement $\mathbf{a}^c$ (where every bit in $\mathbf{a}$ is inverted) are shown to the network. Thus the norm of the new input vector $|(\mathbf{a}, \mathbf{a}^c)|$ will be constant.)

In [4], we showed that in more general learning tasks, in which target outputs are arbitrary binary vectors (not just simple category codes), the *MTA can* in fact arise even if the inputs to the network are complement coded, and the training set is non-contradictory, i.e. no two identical inputs have different target values.

An example training sequence that leads the ARTMAP network into a typical *MTA* condition (which was referred to as *MTA-S* in [4]) is shown in table 1.

A graphical illustration of this situation is shown in figure 2, which helps understand why this kind of *MTA* occurs in ARTMAP.

It was shown that the timing condition under which ART_a inputs and ART_b targets are presented to the network is crucial. Namely, theorem 1 in [4] states:

> "The match tracking anomaly will *never* arise in the ARTMAP network when complement coded input/target pairs from a non-contradicting training set are presented to the network such that in each learning trial the ART_a input is presented first, and the network is allowed to make a prediction through the F^{ab} map field before the ART_b (target) input is presented."

If the ARTMAP network is able to make a prediction before the target is presented, the ART_b module will be "primed" by the F_2^b layer, which will send down its category template to

Step	Input $\mathbf{a}^{(i)}$	ART_a cat #	$F_2^a \rightarrow F_2^b$ Mapping	ART_b categories $\mathbf{z}_1^b$	$\mathbf{z}_2^b$	Target $\mathbf{b}^{(i)}$
1	1 1 1 1 1	1	$1 \rightarrow 1$	1 1 1 1 1 ◁	unused	1 1 1 1 1
2	1 1 1 0 0	2	$2 \rightarrow 1$	1 1 1 x x ◁	unused	1 1 1 0 0
3	0 0 0 1 1	3	$3 \rightarrow 2$	1 1 1 x x	0 0 0 1 1 ◁	0 0 0 1 1
4	1 0 0 1 1	4	$4 \rightarrow 2$	1 1 1 x x	x 0 0 1 1 ◁	1 0 0 1 1
5	0 1 0 1 1	5	$5 \rightarrow 2$	1 1 1 x x	x x 0 1 1 ◁	0 1 0 1 1
6	0 0 1 1 1	6	$6 \rightarrow 2$	1 1 1 x x	x x x 1 1 ◁	0 0 1 1 1
7	1 1 1 1 1	1	$1 \rightarrow 1$	▷ 1 1 1 x x ◁	x x x 1 1	1 1 1 1 1
8	1 0 0 0 0	7	$7 \rightarrow 1$	1 x x x x ◁	x x x 1 1	1 0 0 0 0
9	1 1 1 1 1	1	$1 \rightarrow 1?2$	▷ 1 x x x x	x x x 1 1 ◁	1 1 1 1 1

Table 1: An example sequence of input/target pairs that lead the ARTMAP network to the *MTA*. Here, each input pattern is identical to its corresponding target (i.e. auto-associative learning), and the norm of all patterns is 5. For clarity, the complement patterns are not shown. In category prototypes, a "x" means "don't care", which corresponds to a "00" bit pair in complement coding. For the ART_a module, only the winning nodes are shown. We assume that the ART_a vigilance is high enough (e.g. $\rho_a = 0.9$) so the 11111 pattern will be the only one in its own category. ART_b vigilance is less than 1/5 (e.g. $\rho_b = 0.1$). The symbol "▷" appears beside the ART_b category that the network predicts through the $F_2^a \rightarrow F_2^b$ associations in each trial. The symbol "◁" denotes the category that is selected by the ART_b module before the corresponding input is presented. So in step 7, the network's prediction is confirmed (i.e. "▷" and "◁" appear at the same category), while there is a mismatch in step 9, which eventually causes the *MTA*. (The absence of the "▷" symbol in rows 1-6 and 8 indicates that the network did not have a prediction, and the corresponding $J \rightarrow K$ links were created in those trials.) The $F_2^a \rightarrow F_2^b$ mappings that the network created or used in each trial are also shown. Steps 1 to 6 illustrate a kind of "erosion" process of ART_b category $\mathbf{z}_2^b$, while steps 7 and 8 "prepare" the *MTA* condition that occurs in step 9. (Here we assume that new ART categories are created in the order of 1, 2, 3 and so on.)

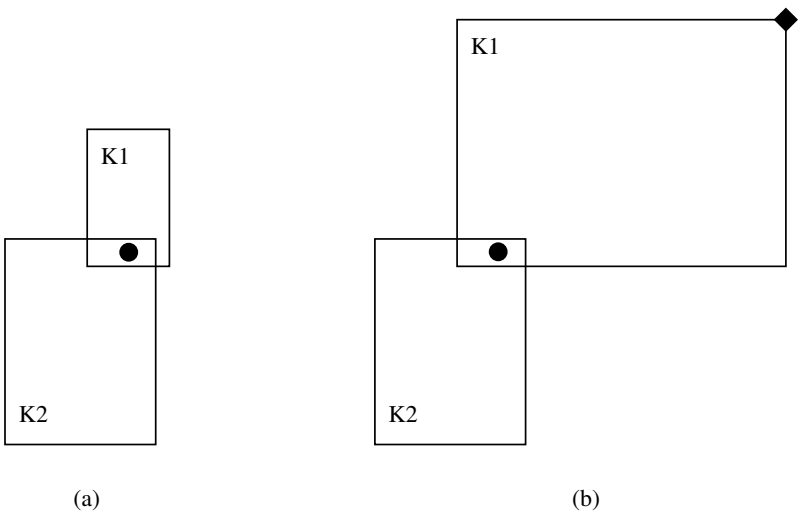


Figure 2: Graphical illustration of a typical *MTA* condition. Target patterns are represented as points on the two-dimensional plane. Categories are represented as rectangles, the area of each rectangle being proportional to the size of the category they represent. Figure 2a shows that category $K1$ is initially smaller (i.e. more specific) than $K2$, and target $\mathbf{b}^{(i1)}$ (circle) is inside (i.e. proper subset of) both $K1$ and $K2$. If at some later stage, another target input is presented (diamond, in figure 2b) that causes $K1$ to generalise to include that input; $K1$ could grow larger than $K2$ as a result. When target $\mathbf{b}^{(i1)}$ is re-presented, this time category $K2$ will capture it since it has now become more specific than $K1$. This will cause the anomaly to emerge as the network still predicts $K1$ through its previously learned $J1 \rightarrow K1$ link.

the F_1^b layer via the top-down pathways. We assume that the network dynamics are such that this top-down template will be tested first at the F_1^b layer as the target vector is registered into the F_0^b layer. If this top-down template (i.e. the network's prediction) and the target are sufficiently close to each other, no reset wave will be triggered, and the current F_2^b selection will be confirmed, and subsequently learned.

Since the Fuzzy ARTMAP network does incorporate the ARTMAP network as a special case, we can expect the *MTA* to occur in this network, too. However, the timing issue was not discussed in [6], probably because in all the experiments ρ_b was set to 1. If $\rho_b = 1$, no recoding of ART_b categories can occur, and therefore the match tracking anomaly will not arise (see theorem 3 in [4]).

In the Fuzzy ARTMAP network, when the ART_b module is in slow learning mode, i.e. $\eta < 1$, it can no longer be guaranteed that

$$\mathbf{z}_K^b = \mathbf{b}^{(i)} \wedge \mathbf{z}_K^b \tag{5}$$

once F_2^b node K has encoded target input $\mathbf{b}^{(i)}$ according to (3). However, (5) was an important assumption for the proof of theorem 1 in [4]. As a consequence of this,

> the match tracking anomaly *cannot be avoided* in the Fuzzy ARTMAP network unless $\rho_b = 1$.

4 Improvements to the learning algorithm

We propose the following two improvements to the original Fuzzy ARTMAP learning algorithm (which will be called FAM henceforth).

Improvement 1 ("priming"): Upon presentation of each $(\mathbf{a}^{(i)}, \mathbf{b}^{(i)})$ pair from the training set, find an ART_a category J for the $\mathbf{a}^{(i)}$ input first, select an output category K (via the F^{ab} map field), then "prime" the F_1^b layer with the prototype vector $\mathbf{z}_K^b$ of the selected F_2^b node before presenting the target output $\mathbf{b}^{(i)}$[1]. This is essentially the "timing condition" of theorem 1 in [4] applied here to the Fuzzy ARTMAP network.

Improvement 2 ("unlearning"): Once the *MTA* condition is detected[2], "unlearn" the existing $J \to K$ link and create a new $J \to K'$ one where K' is the selected ART_b category on the current target pattern.

The learning algorithm with the above modifications will be referred to as FAM-UP (for "FAM with Unlearning and Priming").

Note that the above improvements will *not* eliminate the *MTA* altogether (see section 7), but will

- minimise the occurrence of *MTA*, and
- ensure *MTA* will not re-emerge on the same input/target pair from a finite training set.

As a consequence of the reduction in *MTA* occurrences, training time is also reduced, which is demonstrated in the following section.

[1] This way node K will be tested against vigilance before any other F_2^b node, which may not happen were the ART_b module left to categorise $\mathbf{b}^{(i)}$ independently.

[2] Note that it can only occur in slow learning mode if improvement 1 is already applied.

ART_b learning rate η_b	Training time in epochs		MTA % of Epochs		MTA % of Trials	
	FAM	FAM-UP	FAM	FAM-UP	FAM	FAM-UP
0.1	14.66	10.14	79.60	25.44	4.42	0.33
0.2	11.49	6.18	81.54	31.71	3.86	0.41
0.3	10.45	4.45	73.77	33.25	3.18	0.43
0.4	9.36	3.27	63.46	27.21	2.34	0.36
0.5	9.03	2.90	61.57	22.75	1.89	0.31
0.6	9.64	2.69	60.16	22.67	2.74	0.30
0.7	7.91	2.24	52.46	13.83	2.13	0.17
0.8	6.79	2.76	49.77	9.42	2.13	0.10
0.9	5.03	1.97	29.22	1.01	0.75	0.01
1.0	2.12	1.94	11.32	0	0.19	0

Table 2: This table compares the original and improved Fuzzy ARTMAP learning algorithms when the ART_b learning rate changes from 0.1 to 1. The ART_a learning rate η_a was set to 1. Each row shows averages over the same set of 100 training sessions. In each training session, the network was trained completely on the training set, i.e. training stopped when no further weight changes occurred in the network for 5 full training cycles (or epochs). The ART_a and ART_b vigilance parameters (ρ_a, ρ_b) were 0.4 and 0.1, respectively. The "MTA % of epochs" columns show the percentage of epochs in which MTA occurred at least once. The "MTA % of trials" columns show the percentage of learning trials (i.e. input/target presentations) in which MTA occurred during the 100 training sessions. The choice parameters (β_a and β_b) were set to 0.001.

5 Experimental results

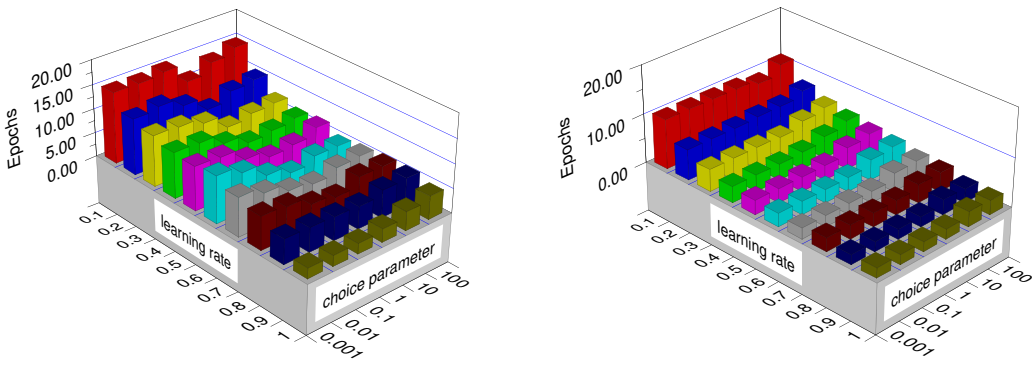
In order to evaluate the new Fuzzy ARTMAP learning algorithm (FAM-UP), we carried out experiments on the "zoo" machine learning benchmark database [9]. Unlike in [7], where target outputs were codes for correct output categories, here the Fuzzy ARTMAP network was used in an "auto-associative" setting to develop two-level class hierarchies from the training set (similar to [2]). In particular, each input was identical to its target pair (i.e. $\mathbf{a}^{(i)} \equiv \mathbf{b}^{(i)}$), and both the ART_a and ART_b modules developed clusterings of patterns from the same training set at different levels of vigilance ($\rho_a > \rho_b$).

The "zoo" database contains 101 instances of animals described with 18 attributes. Out of these attributes, we used the 15 boolean ones that indicate the presence or absence of certain features like "hair", "aquatic", "domestic" and so on. We also used the "number of legs" attribute, which is a set of 6 integers. The "type" (or class) attribute was ignored.[3] The patterns were presented to the Fuzzy ARTMAP network in complement coding resulting in 36-dimensional binary vectors of equal norm (= 16).

Training proceeded by showing each input/target pair from the training set repeatedly until convergence was reached, i.e. the network stabilised itself on the training set. Convergence was established if the number of input and target patterns for each ART_a and ART_b category did not change for the past 5 presentations of the entire training set. After training, each input and target pattern had direct access, i.e. without search, to its corresponding prototype.

Table 2 shows a comparison of the FAM and FAM-UP algorithms.

[3]The 18th attribute ("animal name") was simply used as a label for the individual instances.



(a) FAM (b) FAM-UP

Figure 3: This set of 3D bar charts show the training times in epochs for the original Fuzzy ARTMAP algorithm and FAM-UP with respect to the ART_b learning rate (η_b, along the y-axis, in the range of 0.1 to 1) and the ART_a and ART_b choice parameters ($\beta = \beta_a = \beta_b$, along the x-axis, in the range of 0.001 to 100). The height of each bar measures the average number of epochs over the same set of 100 sessions as in table 2. All other parameter settings were identical, too.

It can be seen from the table that applying FAM-UP reduced training time by 9 to 72%. We can also observe a 2 to 30-fold decrease in the number of epochs where *MTA* occurred during training. A more significant (6 to 75-fold) reduction is shown in the "*MTA* per trial" measure with the improved algorithm. Note that the *MTA* was eliminated completely when the network stabilised itself.

We can also see that *MTA* disappears with $\eta = 1$. This is the case that was discussed in theorem 1 in [4].

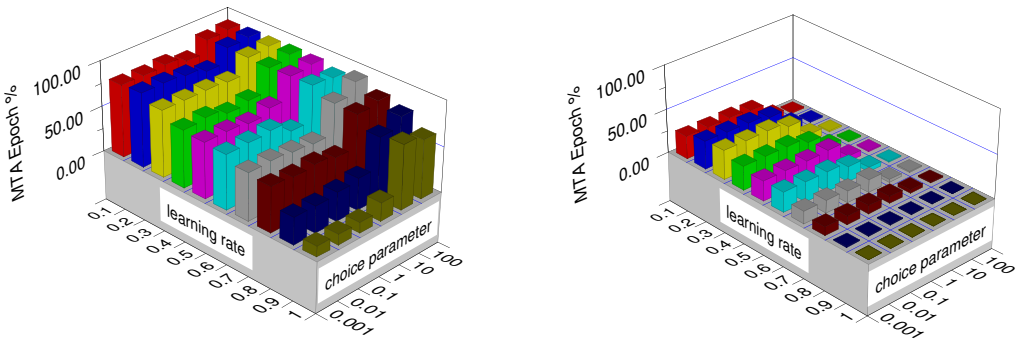
We also carried out experiments to see how changing the choice parameter (β) and learning rate (η) affects training time and the frequency of *MTA*.

Figure 3 shows the average number of epochs for both the original Fuzzy ARTMAP training algorithm (FAM) and the improved one (FAM-UP).

First, we can see a general reduction of training time (measured in epochs) in almost all settings of η and β. (Note that the bars along the "learning rate" axis in the $\beta = 0.001$ row correspond to the values in columns 1 and 2 in table 2.) This is due mainly to the unlearning capability of FAM-UP (see improvement 1) which enables the network to correct its mistakes quickly when *MTA* occurs. Also, the choice parameter appears to have a smaller effect on training time when the FAM-UP algorithm is used.

Figures 4a and 4b show the *MTA* frequencies of FAM and FAM-UP, respectively, with respect to η and β.

Here, a dramatic drop in *MTA* occurrences can be observed over all settings of η and β, especially in the $\beta = 10$ and $\beta = 100$ cases. (Note that the bars along the "learning rate" axis in the $\beta = 0.001$ row correspond to the values in columns 3 and 4 in table 2.) This is due to priming, which eliminates *MTA* completely in the $\eta = 1$ case, as well as unlearning, which causes the network not to repeat the same *MTA* in subsequent epochs. The *MTA* frequencies are higher in the $\beta = 10$ and $\beta = 100$ rows of the original algorithm, which is



(a) FAM (b) FAM-UP

Figure 4: This set of 3D bar charts show the *MTA* frequencies for the original Fuzzy ARTMAP algorithm and FAM-UP with respect to the ART_b learning rate (η_b, along the y-axis, in the range of 0.1 to 1) and the ART_a and ART_b choice parameters ($\beta = \beta_a = \beta_b$, along the x-axis, in the range of 0.001 to 100). The height of each bar measures the average percentage of epochs in which *MTA* occurred at least once over the same set of 100 sessions as in table 2. All other parameter settings were identical, too.

the result of a combined effect of two different *MTA* conditions (referred to as *MTA*-S and *MTA*-β in [4]). Figure 4b shows that one condition (*MTA*-β) is completely eliminated in the improved learning algorithm.

6 Discussion

On the basis of the analytical and experimental results presented in this paper, one might think that the Fuzzy ARTMAP has some sort of "design flaw" which needs to be rectified. While the *MTA* is indeed an undesirable condition that affects the network's performance, we point out again that the *MTA does not occur if* $\rho_b = 1$. This is the typical setting in supervised learning of recognition categories that the ARTMAP network *was designed for*. It is also how the Fuzzy ARTMAP in [6] was applied throughout even when the task was to approximate a continuous function. However, ART_b vigilance is a free parameter in Fuzzy ARTMAP, and there is no reason why it cannot be set to < 1 in certain applications.

We also note that one of the improvements, "priming" the ART_b module before presenting the target, is not the obvious timing to apply in computer simulations. Since the ARTMAP network is built up of two ART modules, it would seem more appropriate from the point of view of effective modular programming to handle those two networks independently and loosely coupled (i.e. letting them interact only outside of each other and one not affecting the internal state of the other). Therefore, the timing whereby we first let each ART module classify their input independently (which translates to presenting the target pattern without the ART_b module being primed) appears to be better. However, the proposed timing that helps avoid *MTA* is directly the opposite: the output of ART_a changes the internal state of ART_b (by priming it through the map field), which affects how ART_b processes its input! Perhaps this paper could also serve as a lesson that certain assumptions and simplifications in computer simulations cannot be made arbitrarily, and without any consequences.

7 Conclusion

In this article, we discussed the match tracking anomaly in the Fuzzy ARTMAP neural network, and introduced two improvements to the standard Fuzzy ARTMAP learning algorithm: one that is concerned with the timing at which ART_a inputs and ART_b targets are presented to the network; and another that explicitly "unlearns" an $F_2^a \rightarrow F_2^b$ link once the anomaly occurs. We have shown that these modifications not only result in faster training, but also reduce the occurrence of *MTA* significantly. Moreover, if the network is trained on a finite training set, the *MTA* can be eliminated this way completely. If, however, the network is used in a real-time autonomous learning environment, the *MTA* cannot be eliminated, but at least can be *detected*. Note also that in real-world environments the *MTA* can be caused by contradicting input/target patterns. How the network should handle such situations must be the subject of further investigation.

Acknowledgement

This work was in part supported by the Department of Computer Science at Victoria University of Wellington, New Zealand. I wish to thank Andrew Vignaux for his programming support as well as the anonymous reviewers for their comments.

References

[1] D. Alpsan. Auditory Evoked Potential Classification by Unsupervised ART 2-A and Supervised Fuzzy ARTMAP Networks. In *Proceedings of the IEEE International Conference on Neural Networks*, volume 6, pages 3512–3515. IEEE Press, 1994.

[2] G. Bartfai. Hierarchical Clustering with ART Neural Networks. In *Proceedings of the IEEE International Conference on Neural Networks*, volume 2, pages 940–944. IEEE Press, 1994.

[3] G. Bartfai. An Improved Learning Algorithm for the Fuzzy ARTMAP Neural Network. In K. Kasabov and G. Coghill, editors, *ANNES'95, The Second New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems*, pages 34–37. IEEE Computer Society Press, November 1995.

[4] G. Bartfai. On the Match Tracking Anomaly of the ARTMAP Neural Network. *Neural Networks*, 9(2):295–308, 1996.

[5] G.A. Carpenter and S. Grossberg. A Massively Parallel Architecture for a Self-Organizing Neural Pattern Recognition Machine. *Computer Vision, Graphics, and Image Processing*, 37:54–115, 1987.

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

[7] G.A. Carpenter, S. Grossberg, and J.H. Reynolds. ARTMAP: Supervised Real-Time Learning and Classification of Nonstationary Data by a Self-Organizing Neural Network. *Neural Networks*, 4:565–588, 1991.

[8] G.A. Carpenter, S. Grossberg, and D.B. Rosen. Fuzzy ART: Fast Stable Learning and Categorization of Analog Patterns by an Adaptive Resonance System. *Neural Networks*, 4:759–771, 1991.

[9] P. M. Murphy and D. W. Aha. UCI Repository of Machine Learning Databases [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Technical report, Department of Information and Computer Science, University of California, Irvine, CA, 1994.

[10] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning internal representations by error propagation. In D.E. Rumelhart and J.L. McClelland, editors, *Parallel Distributed Processing; Explorations in the Microstructure of Cognition; Foundations*, volume 1, pages 318–362. The MIT Press, 1986.

[11] L. A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.