VICTORIA UNIVERSITY OF WELLINGTON

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 Problem of the ARTMAP Neural Network

G. Bartfai

Technical Report CS-TR-94/2
January 1994

Abstract

In this paper, we report a problem with the ARTMAP neural network that can arise as a result of its *match tracking* process [6]. In particular, we will show that the anomalous situation that is claimed in [6] never to arise when (binary) input patterns have the same number of 1's *can* in fact occur. We have constructed a class of counter-examples, and have given necessary conditions for the network parameters, for which a given sequence of the example patterns will drive the network into the error condition. A concrete example is also given to illustrate the result of the analysis of the problem. Results of some further experiments have supported our claims, and also suggested there might be some other, more general conditions for the match tracking problem.

Author Information

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

1 Introduction

Self-organising neural networks are a class of neural architectures that can develop recognition categories by learning without a teacher. They are in direct contrast to many popular feed-forward neural networks, like the back-propagation network [12] that can learn mappings by showing them desired input/target pairs repeatedly. Unsupervised learning can be found both at sensory-level of mammals, and at higher, cognitive levels of humans. Therefore, it has been an important topic in neural network research (e.g. [13], [10], [9]) for decades. Many self-organising models, however, suffer from the problems of computationally expensive training and instability.

The *Adaptive Resonance Theory* (ART) architectures (e.g. [2], [3], [7], [4]) were developed to offer a solution to the above problems. They are self-organising architectures that are capable of developing stable recognition codes in real time, in response to arbitrary sequences of input patterns. While ART networks apply unsupervised learning, they can also be combined to implement arbitrary mappings through supervised learning. The binary version of these architectures is called *ARTMAP* [6], and the continuous one is called *Fuzzy ARTMAP* [5].

In this article, we look at a particular aspect of the ARTMAP network, which is mentioned in [6], but not analysed in detail. Namely, there can be a situation, where an input pattern that perfectly matches an ART_a category causes a predictive error at the output, thereby triggering a so called *inter-ART reset* signal. That reset signal normally raises the ART_a vigilance p_a with a minimal amount necessary to cause ART_a to search for a better ART_a category. However, since the input is matched perfectly by ART_a, there will be no better category found by raising the ART_a vigilance (in this case above 1), or otherwise. In this situation, "F_2^a shuts down for the rest of the input presentation", according to [6]. It is also claimed that such "situation does not arise when all ART_a inputs $\mathbf{a}$ have the same number of 1's". However, we will show that such situation *can* occur by constructing a class of patterns that can cause this problem under certain conditions. An analysis of such conditions is also presented here. We shall also show a concrete counter-example to illustrate the results of the analysis.

In section 2, we discuss the major features of the ART architectures, and the ARTMAP network in particular. The analysis of the match tracking problem is discussed in section 3. Conclusions are drawn in section 4.

2 The ARTMAP neural network

The *Adaptive Resonance Theory* (ART) architectures are neural networks that develop stable recognition codes in real time by self-organisation, in response to arbitrary sequences of input patterns. They were designed to solve the stability-plasticity dilemma that every intelligent machine learning system has to face. Namely, how to keep learning new events without forgetting previously learned information. A basic ART module has three layers[1]: the *input* layer ($F0$), the *comparison* layer ($F1$), and the *recognition* layer ($F2$), with M, M and N neurons, respectively (see module ART_a or ART_b on figure 1). The F1 and F2 layers interact with each other through bottom-up and top-down connections, whose weights are modified when the network learns. At each presentation of a new non-zero binary input pattern $\mathbf{x}$ ($x_i \in \{0,1\}, i = 1,2,...,M$)[2], the network attempts to classify it into one of the existing categories based on its similarity to the stored prototype of each category,

[1] There are both binary (ART1) and continuous (ART2) versions of ART. From now on, we shall use ART to refer to ART1, unless otherwise stated.

[2] Throughout this article, we try to use the same notation as in [6] for easy reference to it whose claim is in the centre of this article.

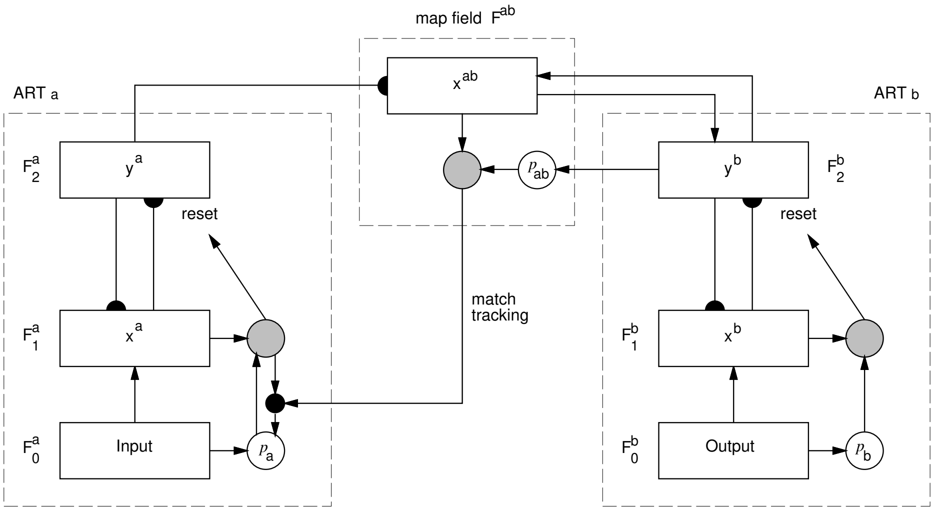

Figure 1: ARTMAP network architecture

represented by individual nodes in the F2 layer. More precisely, for each node j in the F2 layer, $T_j = \sum_{i=1}^{M} x_i Z_{ij}$ is calculated, where Z_{ij} is the strength of the bottom-up connection between F1 node i and F2 node j. Then the top-down template $\mathbf{z}_J$ ($z_{Ji} \in \{0,1\}, i = 1,2,...,M$) of the F2 node with the highest T_j is compared to the current input pattern at F1. If $|\mathbf{x} \cap \mathbf{z}_J||\mathbf{x}|^{-1} \geq p$ (where p is a system parameter called *vigilance* ($0 \leq p \leq 1$), and $|\cdot|$ is the norm operator, $|\mathbf{x}| \equiv \sum_{i=1}^{M} x_i$) then input $\mathbf{x}$ is classified to F2 node J and both the bottom-up and top-down connection weights are adjusted for that node: $\mathbf{Z}_J^{new} = (\mathbf{x} \cap \mathbf{z}_J^{old}) \cdot (\beta + |\mathbf{x} \cap \mathbf{z}_J^{old}|)^{-1}$ and $\mathbf{z}_J^{new} = \mathbf{x} \cap \mathbf{z}_J^{old}$, respectively, where $\beta > 0$ (*choice parameter* [6]) is constant. Otherwise, i.e. when the stored prototype $\mathbf{z}_J$ does not match the input sufficiently, the winning F2 node 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 either the network finds a stored category whose prototype matches the input closely enough, or allocates a new F2 node, and 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 finds in a given training set. The higher the vigilance level, the more, finer categories will be developed.

The architecture of the ARTMAP network can be seen in figure 1.[3] It consists of two ART modules that are linked together through an "inter-ART" associative memory, called *map field* (F^{ab}). Module ART_a (with vigilance p_a) learns to categorise the binary input patterns presented at layer F_0^a, while module ART_b (with vigilance p_b) develops categories of the target patterns presented at layer F_0^b. They do not work independently though. More specifically, when an input pattern is presented, the ART_a module will attempt to make a prediction of the category to which the current target belongs through the map field. If the prediction is right, both modules will learn their respective inputs by modifying the prototypes of the corresponding categories. Otherwise, a process called *match tracking* starts, whereby the vigilance level of the ART_a module is raised by the minimal amount needed to cause mismatch with the current input, which will subsequently trigger a search for a new, better category. 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

[3]This figure is a simplified version of the one that appeared in [5].

corresponding link in the map field, which will learn the current input-target pair correctly. The ART_a vigilance is then allowed to return to its resting level (p_a).

The following section looks at the claim in [6] that is the focus of this article.

3 Analysis of the match tracking problem

The match tracking problem is introduced in [6] as follows:

> "In particular, if $\mathbf{a} \subseteq \mathbf{z}_J^a$ [$\mathbf{a}$ is an ART_a input here], match tracking makes $p_a > 1$, so $\mathbf{a}$ cannot activate another category in order to learn the new prediction. The following anomalous case can thus arise. Suppose that $\mathbf{a} = \mathbf{z}_J^a$ but the ART_b input $\mathbf{b}$ mismatches the ART_b expectation $\mathbf{z}_K^b$ previously associated with J. Then match tracking will prevent the recoding that would have associated $\mathbf{a}$ with $\mathbf{b}$. That is, the ARTMAP system with fast learning and choice will not learn the prediction of an examplar that *exactly* matches a learned prototype when the new prediction contradicts the previous predictions of the examplars that created the prototype. This situation does not arise when all ART_a inputs $\mathbf{a}$ have the same number of 1's... Once this occurs, no input $\mathbf{a}$ can ever be a subset of $\mathbf{z}_J^a$... In particular, the situation described in the previous section [see above] cannot arise."

The same number of 1's in the input can be guaranteed by representing input patterns in *complement coding* [8], according to which both the input $\mathbf{a}$ and its complement $\mathbf{a}^c$ are shown to the network. Thus the norm of the new input vector $|(\mathbf{a}, \mathbf{a}^c)|$ will be constant (M_a in this case).

So the claim is that it will never become necessary to recode a category $\mathbf{z}_J^a$ that perfectly matches an input, as such situation simply never arises.

In the following, we shall show that such situations in fact *can* occur by constructing such an example and giving necessary conditions for it to arise.

Let us take three input pairs $(\mathbf{a}^{(1)}, \mathbf{b}^{(1)})$, $(\mathbf{a}^{(2)}, \mathbf{b}^{(2)})$ and $(\mathbf{a}^{(3)}, \mathbf{b}^{(3)})$. For simplicity, let us assume that

$$\mathbf{a}^{(i)} \equiv \mathbf{b}^{(i)}, i = 1, 2, 3.$$

(This will essentially mean training the network in an auto-associative way, similar to the N-2-N encoder problem, for example.) All three patterns are of equal norm[4], i.e.

$$|\mathbf{a}^{(1)}| = |\mathbf{a}^{(2)}| = |\mathbf{a}^{(3)}| = M \quad ,$$

and they satisfy the following conditions:

$$|\mathbf{a}^{(1)} \cap \mathbf{a}^{(2)}| = K \quad (K > 0), \tag{1}$$

$$|\mathbf{a}^{(1)} \cap \mathbf{a}^{(3)}| = L \quad (L > 0), \tag{2}$$

such that

$$L \geq K, \tag{3}$$

and

$$|\mathbf{a}^{(2)} \cap \mathbf{a}^{(3)}| = 0 \ . \tag{4}$$

For simplicity, let us further assume that

$$K + L = M. \tag{5}$$

[4] This can be satisfied by assuming that $\mathbf{a}^{(i)}$ are binary patterns in complement coding.

Inputs	ART_a categories $\mathbf{z}_1^a$	$\mathbf{z}_2^a$	$\mathbf{z}_3^a$	ART_b categories $\mathbf{z}_1^b$	$\mathbf{z}_2^b$
$\mathbf{a}^{(1)}$	$\mathbf{a}^{(1)}$	—	—	$\mathbf{a}^{(1)}$	—
$\mathbf{a}^{(2)}$	$\mathbf{a}^{(1)}$	$\mathbf{a}^{(2)}$	—	$\mathbf{a}^{(1)} \cap \mathbf{a}^{(2)}$	—
$\mathbf{a}^{(3)}$	$\mathbf{a}^{(1)}$	$\mathbf{a}^{(2)}$	$\mathbf{a}^{(3)}$	$\mathbf{a}^{(1)} \cap \mathbf{a}^{(2)}$	$\mathbf{a}^{(3)}$

Table 1: Input patterns $\mathbf{a}^{(1)}$, $\mathbf{a}^{(2)}$ and $\mathbf{a}^{(3)}$, and the developed ARTMAP categories.

We now show that, under certain conditions, presenting the above patterns in the order of $(\mathbf{a}^{(1)}, \mathbf{a}^{(1)})$, $(\mathbf{a}^{(2)}, \mathbf{a}^{(2)})$, $(\mathbf{a}^{(3)}, \mathbf{a}^{(3)})$, $(\mathbf{a}^{(1)}, \mathbf{a}^{(1)})$ will cause the situation to arise even if the input has the same number of 1's.

Firstly, let p_a be high enough to cause ART_a to allocate an uncommitted F_2^a node for each new, previously unseen pattern (e.g. $p_a = 1$). Also, let

$$p_b < \frac{K}{M} \ . \tag{6}$$

Table 1 shows the input patterns and the developed ART_a and ART_b categories in response to the first 3 of the 4 input pairs. We can see from the table that each new input creates a new ART_a category, as was assumed above.

We shall now look at the behaviour of the ART_b module. Pattern #2 will be captured by category #1, as it will achieve the maximal bottom-up activation (no other F_2^b nodes active as yet, and the initial bottom-up weights are chosen to be small enough for the uncommitted nodes), and

$$|\mathbf{a}^{(2)} \cap \mathbf{z}_1^b| = K > p_b M \ ,$$

due to (6). Prototype $\mathbf{z}_1^b$ is then modified to learn $\mathbf{a}^{(1)} \cap \mathbf{a}^{(2)}$. Pattern #3 will create a new ART_b category $\mathbf{z}_2^b$ as there will be a total mismatch between $\mathbf{a}^{(3)}$ and $\mathbf{z}_1^b$ due to (4), and that $\mathbf{a}^{(1)} \cap \emptyset = \emptyset$.

The interesting case is when pattern $(\mathbf{a}^{(1)}, \mathbf{a}^{(1)})$ is presented again. Here both ART_b categories will match the input sufficiently due to (3) and (6), so the selection will be based entirely on the bottom-up activation levels, which can be calculated in the following way. The bottom-up weights for ART_b categories #1 ($\mathbf{z}_1^b$) and #2 ($\mathbf{z}_2^b$) will be $\frac{1}{\beta+K}$ and $\frac{1}{\beta+M}$, respectively, where $\beta > 0$ is constant (see above). Now the bottom-up activations will therefore be $\frac{K}{\beta+K}$ and $\frac{L}{\beta+M}$, respectively. For fixed β and K, L can always be set such that the input pattern will select either category #1 or #2. This can be seen by examining the

$$\frac{K}{\beta + K} < \frac{L}{\beta + M} \tag{7}$$

inequality. If (7) is satisfied, then ART_b category #2 will be selected. However, since the first presentation of the $(\mathbf{a}^{(1)}, \mathbf{a}^{(1)})$ pair (pattern #1) caused the association to be created between ART_a category #1 and ART_b category #1, this time the network will make a prediction error at ART_b. Therefore, it will try to raise the ART_a vigilance level p_a to cause ART_a to search for a better match. But there is no better match than a perfect one! So we get the anomalous situation that was described in [6]. Now we will show that (7) can be satisfied with choosing β and K arbitrarily.

After some rearrangement of (7), and noting (5), we get the equivalent inequality of

$$\frac{\beta}{K} > \frac{\beta + K}{L} \ . \tag{8}$$

	ART_a categories			ART_b categories	
Inputs	$\mathbf{z}_1^a$	$\mathbf{z}_2^a$	$\mathbf{z}_3^a$	$\mathbf{z}_1^b$	$\mathbf{z}_2^b$
0 0 0 0	0 0 0 0	—	—	0 0 0 0	—
0 1 1 1	0 0 0 0	0 1 1 1	—	0 − − −	—
1 0 0 0	0 0 0 0	0 1 1 1	1 0 0 0	0 − − −	1 0 0 0

Table 2: Example input patterns, and the developed ARTMAP categories

The right hand side of (8) decreases monotonically as $L \to \infty$, with fixed β and K. So we get the following condition for L to satisfy (7).

$$L > \frac{(\beta + K)K}{\beta} \tag{9}$$

We can now see that for any fixed $\beta_0 > 0$ (however small!) and $K_0 \geq 1$ (however large!), we can always find an $L_0 = (\beta_0 + K_0)K_0/\beta_0$ for which any (integer) $L > L_0$ will satisfy (9), and therefore (7).

For example, let $\beta = 1$ and $K = 1$. Then $L = 3$ will satisfy (7), and therefore will cause the anomalous mismatch to occur, even when the inputs have the same number of 1's, as opposed to what was claimed in [6]. The input patterns and the developed categories can be seen in table 2.

4 Conclusion

We have looked at a class of situations in which the anomalous case that is claimed in [6] never to arise when the input (binary) patterns have the same number of 1's *can* in fact occur. The experiments we have carried out have supported the above analysis of the problem.

However, we have also found some other input sequences that do not belong to the class of inputs that have been analysed above. In particular, we have found that a similar problem occurred in about 10% of the training cycles when the ARTMAP network was trained on the "zoo" machine learning benchmark database [11]. Most of these, however were due to racing conditions in the bottom-up activation levels (something that we did not address here). In [1], we used a slightly modified algorithm to deal with such anomalous situations explicitly. More precisely, the ART_a node that caused the mismatch in such cases, was subsequently re-initialised (i.e. freed up) in order to prevent that node from catching the same input again. Although this simple modification worked adequately, we believe this problem needs to be analysed further in order for us to understand the nature of the most general conditions that cause it, and to find an alternative algorithm that would prevent the network from getting into such situations altogether. This would also require more extensive computer simulations on a variety of learning tasks.

Acknowledgement

This work was in part supported by Victoria University of Wellington (Research Grant No.RGNT/594/371).

References

[1] G. Bartfai. Hierarchical Clustering with ART Neural Networks. Technical Report CS-TR-94/1, Department of Computer Science, Victoria University of Wellington, New Zealand, January 1994.

[2] 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.

[3] G.A. Carpenter and S. Grossberg. ART2: Self-Organizing of Stable Category Recognition Codes for Analog Input Patterns. *Applied Optics*, 26(23):4919–4930, December 1987.

[4] G.A. Carpenter and S. Grossberg. ART3: Hierarchical Search Using Chemical Transmitters in Self-Organizing Pattern Recognition Architectures. *Neural Networks*, 3:129–152, 1990.

[5] 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.

[6] 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.

[7] G.A. Carpenter, S. Grossberg, and D.B. Rosen. ART2-A: An Adaptive Resonance Algorithm for Rapid Category Learning and Recognition. *Neural Networks*, 4:493–504, 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] R. Hecht-Nielsen. Counterpropagation networks. *Applied Optics*, 26(23):4979–4984, December 1987.

[10] Teuvo Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59–69, 1982.

[11] P. M. Murphy and D. W. Aha. UCI repository of machine learning databases [machine-readable data repository]. Technical report, Department of Information and Computer Science, University of California, Irvine, CA, 1992.

[12] 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.

[13] D.E. Rumelhart and J.L. McClelland, editors. *Parallel Distributed Processing; Explorations in the Microstructure of Cognition*, volume 1, chapter 5, pages 151–193. The MIT Press, 1986.