Supervised Learning of the Adaptive Resonance Theory System

Yuen-Tai So and Kwok-Ping Chan*

Dept. of Computer Science, The Chinese University of Hong Kong and
Dept. of Computer Science, The University of Hong Kong*

Email : ytso@cs.cuhk.hk

Abstract

A supervised learning ART model (SART) is proposed which is based on the structure of ARTMAP but is much simpler. The techniques of match tracking and complement coding have been implemented to ensure the correct selection of category and stability during the training and testing phases. Two simulations have been done in order to verify and evaluate the classification power of SART. The result of identification of poisonous mushroom by SART is compared with that by ARTMAP.

1. Introduction

Adaptive Resonance Theory (ART), introduced by Grossberg[1], is a neural network model that self-organizes stable recognition codes in real time in response to arbitrary sequences of input patterns. This network has the characteristics to remain adaptive in response to novel events but remain stable in response to familiar events. The stability-plasticity property of ART enables the network to combine the information from external world with the internal knowledge stored[2]. These self-organizing and self-stabilizing features play an important role in human congitive information processing[3].

ART may be considered as a kind of unidirectional pattern clustering[4]; its learning mode is unsupervised. Sometimes, it is desirable to make ART to perform supervised learning so that not only it can self-organise the input samples into different conceptual categories, but also it can be told what actual concepts the categories represented. ARTMAP, introduced by Carpenter and Grossberg[5], is a supervised learning version of ART. However, the overall structure of ARTMAP is too complicated and it is rather difficult to analyse the whole mechanism.

The following sections propose a new model call SART which is a Supervised ART model based on the structure of ARTMAP but with much simpler structure. *Match Tracking* and *complement coding* are employed to solve the problem of instability from random input sequences and improve the system generalization. Two simulations, one in simple character recognition and the other in mushrooms classification, have been conducted and the result is comparable with that by ARTMAP. In this paper, we assume that the reader is familiar with both ART1 and ARTMAP models. To know more details about them, please refer to the corresponding sources.

2. Structure of SART

SART consists of a binary-value ART module (ART1) and an analog-value single layer R (figure 1). The nodes in F2 and R are one-to-one connected. If F2 and R are disconnected, SART is just an ART1 model. When the ART module is in the resonant state after an input pattern is presented to F1, the selected node in F2 will activate the corresponding node in R. The expected result, given by the *teacher,* is learnt and stored in the weight from F2 to R, and that node is then called *committed.* One can see that the layer R replaces the ART_b module in the ARTMAP.

The nodes in R are just the *outstar*[6] structure as shown in figure 2. The vector $\boldsymbol{I}$ is the input from the teacher with all the components have the same value while the vector $\boldsymbol{G} = (g(y_1), g(y_2), \ldots)$ is the output where $g(y_j)$ is the jth component which is a linear threshold function and equals to y_j for $y_j > 0$, and 0 otherwise. The *prediction* (P) for an input pattern is then given by

$$P = \sum_j g(y_j) \qquad ---(1)$$

The activity of the node in R is governed by the following *additive* differential equation :

$$\dot{y}_j = -ay_j + bI + cz_j f(x_j) \qquad ---(2)$$

where y_j is the activity of the node j, z_j is the weight from F2 to R, I is the input from teacher, x_j is the activity of node v_j in F2, $f()$ is the output function in F2, and a, b, c are system parameters.

Without the existence of the teacher ($I = 0$), equation (2) is used in the recognition process; that means the activity of R gives the prediction of the input pattern. It should be noted that only *one* node in F2 is active while the others are asleep during the resonance; that means $f(x_j) = 0$, for all j except $j = J$, where J is the winning node in F2. Therefore y_j, and also $g(y_j)$, diminish when $f(x_j) = 0$.

If the ART module stays in the resonant state for long enough time, the fired node in R will reach equilibrium with $\dot{y}_J = 0$. Since the output of F2 takes a binary value, we let $f(x_J) = 1$, then $y_J = kz_J > 0$ if $z_J > 0$, where $k = c/a$.

Therefore, from (1), $P = g(y_J) = kz_J$. The prediction is proportion to the strength of the synaptic weight. Assume $k = 1$, then y_J is just equal to the value stored in z_J; that means the output of prediction is z_J. We can see from the next section that z_J learns the expected value from the teacher during training.

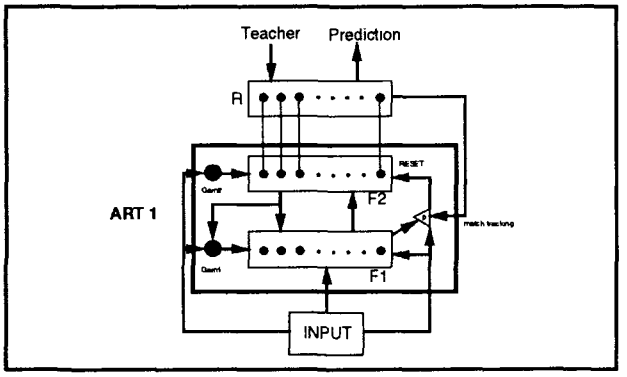

Figure 1. Structure of SART

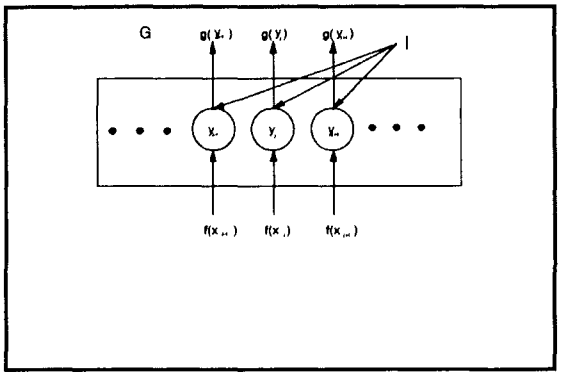

Figure 2. Outstar Configuration in Layer R.

3. Learning in SART

The learning equations of both bottom-up and top-down weights in the ART module satisfy the following *gated decay* differential equation[2] :

$$\dot{w}_{ij} = Kf(x_j)[-E_{ij}w_{ij} + h(x_i)]$$

where w_{ij} is the weight from v_i to v_j, x_i is the activity of node v_i, and K, $-E_{ij}$, $h()$ are system parameters.
Simpler form of equations are used in the implementation of ART1 systems. For the bottom-up learning :

$$w_{iJ} = \frac{ax_i}{\beta + |X^*|}$$

where X^* is the intersection of the bottom-up input pattern and the top-down template from F2, x_i is the ith output of X^*, and a and β are system parameters.
For the top-down learning :

$$\dot{w}_{Ji} = \alpha(x_i - w_{Ji})$$

where α is the learning rate.

If $\alpha > 0$, w_{Ji} is attracted to x_i. In vector terms, if $\alpha > 0$, then $\boldsymbol{w}_J = (w_{J1}, w_{J2}, ..., w_{JM})$ approach X^*. Usually we take $\alpha = 1$, then $w_{Ji} = 1$ if $x_i = 1$. The above equations are called *fast learning* rules because the system reaches equilibrium (stable point) by one step only. For the weights in R, the learning rule of *outstar* is used :

$$\dot{z}_J = (-dz_J + eIx_J)f(x_J)$$

where z_J is the weight from F2 to R, x_J is the activity of v_J in F2, and d, e are system parameters.

For fast learning, we assume the speed of the weight change to reach its asymptotic value is much faster than the decay of the excitation from input I. Therefore,

At equilibrium, $\dot{z}_J = 0$

=> $0 = -dz_J + eIx_J$

But $x_J = 1$ (since F2 is a binary-value layer), therefore,

$z_J = hI$ where $h = e/d$.

In our application, we just set $h = 1$ for simplicity. As a result, $z_J = I$; the weight captures the expected value from the teacher.

4. Match Tracking and Character Recognition

During the training process, when a committed node in R is activated by an input pattern but the value stored in the weight of R is different from the expected result, i.e. $z_J \neq I$, a reset signal, together with a control strategy called *match tracking*, is sent to the orienting subsystem of the ART module to reset the F2 node. *Match tracking*, proposed by Carpenter and Grossberg, in response to a predictive error on training trail, is a process which increases the vigilance parameter of ART by the minimum amount needed to abort the resonant state and to drive a new search for a new category that can establish a correct prediction. An example by Carpenter and Grossberg has shown that without match tracking ARTMAP may produce incorrect result[5]. The same phenomenon occurs in SART.

In order to verify the capability and usefulness of SART, two simulations have been carried out. The first one was to recognize the twenty six alphabets which were 4x4 binary patterns. The patterns are listed in appendix

and the testing sets are listed in table 1. There were totally six testing sets which classified the twenty six alphabets into two to seven groups. The numbers under the alphabets indicated to which group they belong. The patterns were grouped into different categories randomly except test #1 in which the two groups divided are linearly separable[7]. The input patterns were repeatedly presented to the system until 100% accuracy is achieved. Table 2 shows the number of nodes committed in R with different values of ρ_o (baseline vigilance parameter) in the ART module. The stability-plasticity property and the representation power of SART was then verified.

Test #	# of group	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	2	1	0	0	0	1	1	0	1	1	0	1	1	1	1	0	0	0	0	0	1	0	1	1	1	1	1
2	3	1	0	2	0	0	1	2	2	0	1	1	2	0	1	2	0	2	0	1	1	2	2	0	0	1	2
3	4	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1
4	5	4	1	2	3	3	0	4	4	1	2	1	2	3	0	0	1	2	4	4	0	0	1	2	3	0	4
5	6	0	0	0	1	1	1	2	2	2	3	3	3	4	4	4	5	5	5	0	1	2	3	4	5	0	1
6	7	0	1	2	3	1	4	5	6	0	1	2	3	2	5	5	6	0	1	2	3	5	4	5	6	3	5

Table 1. Testing Set in Alphabet Recognition

Test #	# of nodes in R ($\rho_o = 0.0$)	# of nodes in R ($\rho_o = 0.5$)	# of nodes in R ($\rho_o = 0.75$)	# of nodes in R ($\rho_o = 0.9$)
1	11	11	18	22
2	19	19	19	24
3	22	22	22	Unstable
4	22	22	23	25
5	22	22	25	25
6	24	24	24	26

Table 2. Results of Alphabet Recognition

It can be concluded that when ρ_o is large, more categories will be formed. It is reasonable because we want to classify the twenty six alphabets more finer; as a result, generalization will be small. The case when $\rho_o = 0.0$ is called *forced choice* which means after the first choice in F2, the ART module will reach resonant state for any input. Reset occurs only for mismatch between expected result and the value of R.

It was noted that there was an unstable case occurred in test #3. The following section discusses the reason why instability occurs and suggests the solution by using complement coding.

5. Instability and Complement Coding

Firstly, we investigate why instability occurred in test #3 when $\rho_o = 0.9$. The letters 'E' and 'Q' were both classified into the same group (group 0) while the letter 'G' was classified into group 2. During the first round training, 'E' and 'Q' fired the same node in F2 while 'G' occupied another node in F2. But we can see that the intersection of the binary patterns 'E' and 'Q' is the same as that of 'G'.

1111		1111		1111
1000	$\cap$	1001	=	1000
1111		1011		1011
1111		1111		1111
'E'		'Q'		'G'

Unfortunately, at the second round training, when 'G' was presented to the system, the node with template formed by 'E' and 'Q' was selected first. Reset and match tracking took place because of the mismatch between the teacher and R. And then the value of ρ becomes greater than 1.0. Therefore the node formed by 'G' would never be selected and an uncommitted node in F2 was selected. This process repeated indefinitely until all the nodes in F2 were exhausted. Same phenomenon happens in ARTMAP.

This problem occurs because the system has lost some information during learning. The intersect operation in learning makes the ART system stable because it is monotonic decreasing[4]; however, information has been lost. Considering the partial truth table of the intersect operation shown in table 3, all these three combinations of $A \cap B$ yield the same result 0, but they carry different information about A and B. As we can see, the non-existence of a feature in the process of pattern recognition may sometimes contribute to the decision making. Therefore, it may be better to represent both the existence and non-existence of an attribute.

A	B	$A \cap B$
0	0	0
1	0	0
0	1	0

Table 3. Partial truth table of $\cap$

In order to solve this problem, *complement coding* was used[8]. Complement coding represents both the on-response and off-response of an input pattern. Let **a** be the input pattern, then the complement of **a**, denoted $\mathbf{a}^c$, represents the off-response, where $a_i^c \equiv 1 - a_i$. Then the pair $(\mathbf{a}, \mathbf{a}^c)$ forms the complement coding of **a**.

Using complement coding, the binary pattern of the node formed by the intersection of 'E' and 'Q' is

	1111		0000
	1000		0110
(a)	1011	($\mathbf{a}^c$)	0000
	1111		0000

and the template formed by 'G' only is

	1111		0000
	1000		0111
(a)	1011	($\mathbf{a}^c$)	0100
	1111		0000

The two patterns are different in the complement part although their original parts are the same. Table 4 shows the result of alphabet recognition of SART with complement coding.

Comparing tables 2 and 4 gives the phenomenon that the number of categories formed with complement coding is less than that without complement coding. As more information is provided to SART for classification, generalization is usually increased. Complement coding increases generalization is shown in more details in the second simulation.

Test #	# of nodes in R ($\rho_o = 0.0$)	# of nodes in R ($\rho_o = 0.5$)	# of nodes in R ($\rho_o = 0.75$)	# of nodes in R ($\rho_o = 0.9$)
1	8	8	14	24
2	16	16	18	24
3	22	22	23	26
4	19	19	21	26
5	20	20	23	25
6	25	25	25	26

Table 4. Results of Alphabet Recognition with Complement Coding

6. Distinguishing Edible and Poisonous Mushrooms

ARTMAP was tested on a benchmark machine learning database that partitioned a set of observable features of a mushroom as a binary vector, and each mushroom is classified as either edible or poisonous[9]. This data set includes descriptions of 8,124 hypothetical samples corresponding to 23 species of gilled mushrooms in the Agaricus and Lepoiota Family. Among the 8,124 samples, 4,208 (51.8%) samples are edible and 3,916 (48.2%) are poisonous. Each sample describes 22 observable features, which have totally 126 different values, of a mushroom. The result was satisfactory as over 90% correct classification could be obtained when the training set size is over 250 out of 8,124[5]. In order to compare the performance and result with ARTMAP implemented by Carpenter and Grossberg, the same benchmark was also tested by SART. Complement coding has not been applied to ARTMAP but it is applied in SART so as to provide more information. Consequently, the input patterns to SART were 252-element binary vectors, each vector having 126 1's and 126 0's, instead of 126-element binary vectors in ARTMAP.

In this simulation, SART performed only *off-line learning* in which a fixed training set was repeatedly presented to the system until 100% accuracy was achieved on that set. Usually 2 to 3 iterations were needed to stabilize the network. The training sets were ranging in size from 15 to 4,000 samples. System performance was then measured on the test set, which consisted of all the others 8,124 samples not included in the training set. No learning occurred in testing. Two set of testings were performed, using two different baseline setting of vigilance parameter $\rho_o = 0.0$ (forced choice condition) and $\rho_o = 0.7$ (conservative condition). In each set of testing, 10 independent simulations were carried out and the results are listed in tables 5 and 6.

Training Set Size	Average % Correct (Test Set)	Average % Incorrect (Test Set)	Number of SART Categories
15	80.3	19.7	2-4
125	97.7	2.3	4-6
1000	99.6	0.4	4-7
4000	99.9	0.1	5-7

Table 5. Off-line Forced Choice (ρ_o = 0.0) SART Performance

Training Set Size	Average % Correct (Test Set)	Average % Incorrect (Test Set)	Average % No-Response (Test Set)	Number of SART Categories
15	53.5	6.3	40.2	4-5
125	75.9	0.6	23.5	13-17
1000	98.6	0.1	1.3	17-35
4000	99.9	0	0.1	20-42

Table 6. Off-line Conservative (ρ_o = 0.7) SART Performance

Training Set Size	ρ_o = 0.0		ρ_o = 0.7	
	ARTMAP	SART	ARTMAP	SART
15	4-6	2-4	8-10	4-5
125	5-14	4-6	33-37	13-17
1000	7-18	4-7	53-66	17-35
4000	11-22	5-7	61-73	20-42

Table 7. Number of Categories formed by ARTMAP and SART

The fourth column in table 6 shows the average percentage of *No-Response* which means the matching between the input sample and all the patterns learnt in SART does not exceed the baseline vigilance value (0.7). Therefore, SART answers 'Sorry, I don't know'.

The number of categories formed in the forced choice mode is much less than that in the conservative mode, especially when the size of the training set is large. This result is reasonable because in the conservative cases, the similarity between the input pattern and the selected template should not be less than the baseline vigilance value ρ_o (0.7). If the difference between them is large, a new category will be formed to recognize this novel input pattern. As a result, in general, more categories will be generated when the baseline vigilance value is greater. For instance, when the training set size was 4000, one simulation gave 42 categories in conservative mode but 5 only in forced choice mode.

The results shown above are similar to that in ARTMAP by Carpenter and Grossberg except the number of categories formed by SART is much smaller than that in ARTMAP. Table 7 compares the difference. The difference in the number of categories formed is due to the incorporation of complement coding in SART. Complement coding can provide more information to SART so that more generalisation is resulted; therefore, less categories are formed. Without complement coding, the number of categories formed in SART is almost the same as that in ARTMAP. As the number of categories formed is smaller, the learning rate, in the simulation, is faster because the time needed to establish resonant state is shortened, even though the input pattern size is twice of that without complement coding, which takes 30% longer in learning rate. The slower in learning rate is due to the searching of the appropriate template from a larger category set in F2 to match the input pattern. But it may have no influence if the searching is done in parallel mechanism.

7. Conclusions

The self-organising and self-stabilizing capabilities of ART make it a fantastic model, in spite of the complexity of its structure. The ARTMAP provides supervised learning but its structure is more complicated. SART is a simpler supervised ART model than ARTMAP. The supervised learning capability is desirable as it can tell the ART module what the actual concept learnt is and distinguish a subset pattern from the template if they are classified into different groups but their difference is within the tolerant level. The expected value from the teacher in SART is used to differentiate them. ART cannot do that as it self-organizes or classifies the patterns according to their similarity only. Incorporating SART with match tracking and complement coding, the classification ability and system stability can be improved. Besides, the system generalization is enhanced.

References

[1] S. Grossberg, "On the Development of Feature Detectors in the Visual Cortex with Applications to Learning and Reaction-Diffusion Systems", *Biol. Cybernetics*, **21**, 145-159, (1976).

[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, "The ART of Adaptive Pattern Recognition by a Self-Organizing Neural Network", *IEEE Computer*, March, 77-88, (1988).

[4] B. Moore, "ART 1 and Pattern Clustering", *Proc. of the 1988 Connectionist Models Summer School*, San Mateo, CA: Morgan Kaufman Publshers, 174-184, (1988).

[5] G.A. Carpenter and S. Grossberg, "ARTMAP: Supervised Real-Time Learning and Classification of Nonstationary Data by a Self-Organizing Neural Network", *Neural Networks*, **4**, 565-588, (1991).

[6] S. Grossberg, editor, *Studies of Mind and Brain*, volume 70 of *Boston Studies in Philosophy of Science*, D. Reidel Publishing Co : Boston, (1982).

[7] J. Sklansky and G.N. Wassel, *Pattern Classifiers and Trainable Machine*, Springer-Verlag, New York, (1981).

[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] J.S. Schlimmer, *Mushroom database*, UCI Repository of Machine Learning Databases, (1987). (aha@csi.uottawa.ca)

Appendix

The twenty six 4x4 binary alphabet patterns. The two groups separated by the dashed line are linearly separable.