ART:

Adaptive resonance theory (ART) is one kind of neural networks which uses unsupervised learning techniques. It is developed by Stephen Grossberg and Gail Carpenter in 1987. The term “adaptive” and “resonance” used in the name refers that ART networks are open to new learning without losing old information. ART networks implement a clustering algorithm. Input is presented to the network and the algorithm checks whether it fits into one of the already stored clusters. If it fits, then the input is added to the cluster that matches the most else a new cluster is formed.

Operating Principal:

ART operating principal can be classified into following phases:

- **Recognition Phase:** The input vector is compared with the classification presented at every node in the output layer. The output of the neuron becomes “1” if it becomes the best match with the classification applied, otherwise it becomes “0”.
- **Comparison Phase:** In this phase, a comparison of the input vector to the comparison layer vector is done. The condition for reset is that the degree of similarity would be less than vigilance parameter.
- **Search Phase:** In this phase, the network will search for reset as well as the match done in the above phases. Hence, if there would be no reset and the match is quite good, then the classification is over. Otherwise, the process would be repeated, and the other stored pattern must be sent to find the correct match.

Types of ART:

The ARTs can be classified as follows:

- **ART 1:** Simple and basic ART architecture and deals with binary input values.
- **ART 2:** Extended version of ART 1 and works with continuous-valued (analog) input data.
- **Fuzzy ART:** Combination of fuzzy logic and ART.
- **ARTMAP:** Supervised form of ART which is also known as predictive ART as it learns from the previous module.
- **FARTMAP:** Supervised ART including fuzzy logic.

Advantage of ART:

- ART can be used integrated with other techniques to obtain more good results.
- ART networks tackle the stability-plasticity dilemma:

Plasticity: They can always adapt to unknown inputs (by creating a new cluster with new weight vector) if the given input can not be classified by existing clusters.

Stability: Existing clusters are not deleted by the introduction of new inputs (new clusters will just be created in addition to the old ones).

Application of ART:

Mobile robot control, face recognition, land cover classification, target recognition, medical diagnosis, signature verification, clustering web users.

Limitations of ART:

Some ART does not main the same throughout as they depend on the order of the training data or the learning rate.

ART 1:

ART 1 network and its architecture have been discussed here which is designed to cluster binary input patters.

Architecture of ART 1:

The main architecture of ART 1 consists of the following two units:

1. Computational Unit – This unit is made up of the following layers:

a) Input unit (F1 layer) – It is also known as the comparison field (where the inputs are processed). The F1 layer accepts the inputs and performs some processing and transfers it to the F2 layer that best matches with the classification factor. There exist two sets of weighted interconnections for controlling the degree of similarity between the units in the F1 and the F2 layer. It further has the following two portions −

i) F1 a layer Input portion − In ART 1, there would be no processing in this portion rather than having the input vectors only. It is connected to F1 b layer interface portion.

ii) F1 b layer Interface portion − This portion combines the signal from the input portion with that of F2 layer. F1 b layer is connected to F2 layer through bottom up weights $b_{i,j}$ and F2 layer is connected to F1 b layer through top down weights $t_{j,i}$. Bottom-up weights are used to determine output layer candidates that may best match the current input and Top-down weights represent the “prototype” for the cluster defined by each output neuron.

A close match between input and prototype is necessary for categorizing the input. Finding this match can require multiple signal exchanges between the two layers in both directions until "resonance" is established, or a new neuron is added.

b) Cluster Unit (F2 layer) − This is a competitive layer or the recognition field (which consists of the clustering units). The cluster unit with the large net input becomes the candidate to learn the input pattern first and the rest F2 units are ignored. The unit having the largest net input is selected to learn the input pattern. The activation of all other cluster units is set to 0.

c) Reset Mechanism − The work of this control mechanism is to decide whether the cluster unit can learn the input pattern or not depending on how similar its top-down weight vector to the input vector. This is called the vigilance test. Now, if the degree of this similarity is less than the vigilance parameter (ρ), then the cluster is not allowed to learn the pattern and a reset would happen. Thus, we can say that the vigilance parameter helps to incorporate new memories or new information. Higher vigilance produces more detailed memories, smaller clusters, and higher precision. On the other hand, lower vigilance produces more general memories.

Generally, two types of learning exist, slow learning and fast learning. In fast learning, weight update during resonance occurs rapidly. It is used in ART 1. In slow learning, the weight change occurs slowly relative to the duration of the learning trial. It is used in ART 2.

2. Supplement Unit − Actually the issue with Reset mechanism is that the layer F2 must have to be inhibited under certain conditions and must also be available when some learning happens. That is why two supplemental units namely, G1 and G2 is added along with reset unit, R. They are called gain control units. These units receive and send signals to the other units present in the network. '+' indicates an excitatory signal, while '−' indicates an inhibitory signal.

Algorithm:

- Initialize each top down weight $t_{j,i}(0) = 1$
- Initialize bottom up weight

$$b_{i,j}(0) = \frac{1}{n+1}\ \textbf{;} \textit{where n is the number of components in the input vector}\textbf{.}$$

While the network has not stabilized, **do**

1. Present a randomly chosen pattern $x = (x_1 + x_2 + \dots\dots + x_n)$ for learning

2. Let the active set A contain all nodes; calculate

$y_j = b_{1,j}.x_1 + \ldots\ldots\ldots + b_{n,j}.x_n \; for\ each\ node\ j \in A;$

3. **Repeat**

a) Let j* be a node in A with largest y_j with ties being broken arbitrarily;

b) Compute $s^* = (s_1^*, \ldots\ldots\ldots s_n^*)\ where,\ s_i^* = t_{j,i}^*.x_i$;

c) Compare similarity between s^* and x with the given vigilance parameter ρ:

if $\frac{\sum_{i=1}^{n} s_i^*}{\sum_{i=1}^{n} x_i} \le \rho$ **then** remove j* from set A **else** associate x with node j* and update weights:

$$b_{i,j}(new) = \frac{t_{j,i}(old).x_i}{0.5 + \sum_{i=i}^{n} t_{j,i}^*(old).x_i} \quad \text{and} \quad t_{j,i}^*(new) = t_{j,i}^*(old).x_i$$

Until A is empty or x has been associated with some node j.

4)if A is empty then create new node whose weight vector coincides with current input pattern x;

End while

<u>ART Example Computation</u>

Our input vectors are following:

RV1	0	1	0	1	1	1	0
RV2	1	0	1	0	0	0	0
RV3	1	1	0	0	0	0	1
RV4	0	1	0	1	0	1	0
RV5	1	0	0	0	0	0	1

And the vigilance parameter (Roh) $\rho = 0.4$

Solution:

For this example, let us assume that we have an ART – 1 network with 7 input neurons (n=7)

Initially, top down weights set to $t_{j,i}(0) = 1$ and bottom up weights to $b_{i,j}(0) = 1/8$

For the 1st input vector (RV1 0 1 0 1 1 1 0) we get:

$$y_1 = \frac{1}{8}.0 + \frac{1}{8}.1 + \frac{1}{8}.0 + \frac{1}{8}.1 + \frac{1}{8}.1 + \frac{1}{8}.1 + \frac{1}{8}.0 = ½$$

Clearly, y_1 is the winner (there are no competitors).

Since we have:

$$\frac{\sum_{i=1}^{7} t_{j,i}.x_i}{\sum_{i=7}^{7} x_i} = 1 > 0.4$$

The vigilance condition is satisfied, and we get the following new weights:

$$b_{1,2}(1) = b_{1,4}(1) = b_{1,5}(1) = b_{1,6}(1) = \frac{1}{0.5+4} = \frac{1}{4.5}$$

$$b_{1,1}(1) = b_{1,3}(1) = b_{1,7}(1) = 0$$

Also, we have, $t_{j,i}(1) = t_{j,i}(0).x_i$

B (1) = [0 $\frac{1}{4.5}$ 0 $\frac{1}{4.5}$ $\frac{1}{4.5}$ $\frac{1}{4.5}$ 0]

T (1) = [0 1 0 1 1 1 0]

For the 2nd input vector (RV2 1 0 1 0 0 0 0) we get:

y_1 = 0

Clearly, y_1 is still the winner.

However:

$$\frac{\sum_{i=1}^{7} t_{j,i}.x_i}{\sum_{i=7}^{7} x_i} = \frac{0}{2} = 0 < 0.4$$

This time we do not reach the vigilance threshold. This means we must generate a second node in the output layer that represents the current input.

Therefore, top down weights of the new node will be identical to the current input vector.

The new bottom up weights are set to zero in the positions where the input has zeros as well. The remaining weights are set to 1/ (0.5+1+0+1+0+0+0+0) = $\frac{1}{2.5}$

$$B(2) = \begin{bmatrix} 0 & \frac{1}{4.5} & 0 & \frac{1}{4.5} & \frac{1}{4.5} & \frac{1}{4.5} & 0 \\ \frac{1}{2.5} & 0 & \frac{1}{2.5} & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$T(2) = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

For the 3rd input vector (RV3 1 1 0 0 0 0 1) we get:

$$y_1 = \frac{1}{4.5}\ ,\ y_2 = \frac{1}{2.5}$$

Clearly, y_2 is the clear winner.

However:

$$\frac{\sum_{i=1}^{7} t_{j,i} . x_i}{\sum_{i=7}^{7} x_i} = \frac{1}{3} = 0.3 < 0.4$$

This time we do not reach the vigilance threshold. This means the active set is reduced to contain only the 1st node, which becomes the uncontested winner.

$$\frac{\sum_{i=1}^{7} t_{j,i} . x_i}{\sum_{i=7}^{7} x_i} = \frac{1}{3} = 0.3 < 0.4$$

The vigilance test fails for the 1st unit as well. We thus must create a third output neuron:

$$B(3) = \begin{bmatrix} 0 & \frac{1}{4.5} & 0 & \frac{1}{4.5} & \frac{1}{4.5} & \frac{1}{4.5} & 0 \\ \frac{1}{2.5} & 0 & \frac{1}{2.5} & 0 & 0 & 0 & 0 \\ \frac{1}{3.5} & \frac{1}{3.5} & 0 & 0 & 0 & 0 & \frac{1}{3.5} \end{bmatrix}$$

$$T(3) = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

For the 4th input vector (RV4 0 1 0 1 0 1 0) we get:

$$y_1 = \frac{1}{4.5}\ ,\ y_2 = \frac{1}{2.5},\ y_3 = \frac{1}{3.5}$$

Clearly, y_1 is the winner.

However:

$$\frac{\sum_{i=1}^{7} t_{j,i} . x_i}{\sum_{i=7}^{7} x_i} = \frac{3}{3} = 1 > 0.4$$

Therefore, we adapt the 1st node's weight. As usual each top down weight is multiplied by the corresponding element of the current input.

The new unit's bottom up weights are set to the top down weights divided by (0.5+0+1+0+1+0+1+0) = 3.5

The vigilance test fails for the 1st unit as well. We thus must create a third output neuron:

$$B(4)=\begin{bmatrix} 0 & \frac{1}{3.5} & 0 & \frac{1}{3.5} & 0 & \frac{1}{3.5} & 0 \\ \frac{1}{2.5} & 0 & \frac{1}{2.5} & 0 & 0 & 0 & 0 \\ \frac{1}{3.5} & \frac{1}{3.5} & 0 & 0 & 0 & 0 & \frac{1}{3.5} \end{bmatrix}$$

$$T(4)=\begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

For the 5th input vector (RV5 1 0 0 0 0 0 1) we get:

$$y_1 = 0\ ,\ y_2 = \frac{1}{2.5}\ ,\ y_3 = \frac{2}{3.5}$$

Clearly, y_3 is the clear winner.

However:

$$\frac{\sum_{i=1}^{7} t_{j,i}.x_i}{\sum_{i=7}^{7} x_i} = \frac{1}{2} = 0.5 > 0.4$$

Therefore, we adapt the 3rd node's weight.

$$B(5)=\begin{bmatrix} 0 & \frac{1}{3.5} & 0 & \frac{1}{3.5} & 0 & \frac{1}{3.5} & 0 \\ \frac{1}{2.5} & 0 & \frac{1}{2.5} & 0 & 0 & 0 & 0 \\ \frac{1}{2.5} & 0 & 0 & 0 & 0 & 0 & \frac{1}{2.5} \end{bmatrix}$$

$$T(4)=\begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

In the 2nd epoch the 1st i/p vector (RV1 0 1 0 1 1 1 0) gives us:

$y_1 = \frac{3}{3.5}$, y_2 =0, y_3 = 0

Here, y_1 is the winner.

Vigilance test

$$\frac{\sum_{i=1}^{7} t_{j,i}.x_i}{\sum_{i=7}^{7} x_i} = \frac{3}{4} = 0.75 > 0.4$$

No weight update happens.

2nd i/p vector (RV2 1 0 1 0 0 0 0) gives us:

$y_1 = 0$, $y_2 = \frac{2}{2.5}$, $y_3 = \frac{1}{2.5}$

Here, y_2 is the winner.

Vigilance test

$$\frac{\sum_{i=1}^{7} t_{j,i}.x_i}{\sum_{i=7}^{7} x_i} = \frac{2}{2} = 1 > 0.4$$

No weight update happens.

3rd i/p vector (RV3 1 1 0 0 0 0 1) gives us:

$y_1 = \frac{1}{3.5}$, $y_2 = \frac{1}{2.5}$, $y_3 = \frac{2}{2.5}$

Here, y_3 is the winner.

Vigilance test

$$\frac{\sum_{i=1}^{7} t_{j,i}.x_i}{\sum_{i=7}^{7} x_i} = \frac{2}{3} = 0.7 > 0.4$$

No weight update happens.

4th i/p vector (RV4 0 1 0 1 0 1 0) gives us:

$$y_1 = \frac{3}{3.5}, \; y_2 = 0, \; y_3 = 0$$

Here, y_1 is the winner.

Vigilance test

$$\frac{\sum_{i=1}^{7} t_{j,i} . x_i}{\sum_{i=7}^{7} x_i} = \frac{3}{3} = 1 > 0.4$$

No weight update happens.

5th i/p vector (RV5 1 0 0 0 0 0 1) gives us:

$$y_1 = 0, \; y_2 = \frac{1}{2.5}, \; y_3 = \frac{2}{2.5}$$

Here, y_3 is the winner.

Vigilance test

$$\frac{\sum_{i=1}^{7} t_{j,i} . x_i}{\sum_{i=7}^{7} x_i} = \frac{2}{2} = 1 > 0.4$$

No weight update happens.

The network has thus stabilized. The following clusters have been made:

Cluster 1 {RV1 RV4}

Cluster 2 {RV2}

Cluster 3 {RV3 RV5}