Adaptive Resonance Theory : Soft Computing Course Lecture 25 - 28, notes, slides

www.myreaders.info/ , RC Chakraborty, e-mail rcchak@gmail.com , Dec. 01, 2010

http://www.myreaders.info/html/soft_computing.html

Adaptive Resonance Theory (ART)

Soft Computing

Adaptive Resonance Theory, topics : Why ART? Recap - supervised, unsupervised, back-prop algorithms, competitive learning, stability-plasticity dilemma (SPD). ART networks : unsupervised ARTs, supervised ART, basic ART structure - comparison field, recognition field, vigilance parameter, reset module; simple ART network, general ART architecture, important ART networks, unsupervised ARTs - discovering cluster structure. Iterative clustering : non-neural approach, distance functions, vector quantization clustering, algorithm for vector quantization and examples. Unsupervised ART clustering : ART1 architecture, ART1 model description, ART1 algorithm, clustering procedure and ART2.

Adaptive Resonance Theory (ART)

Soft Computing

Topics

(Lectures 25, 26, 27, 28 4 hours)

Adaptive Resonance Theory (ART)

What is ART ?

- ART stands for "Adaptive Resonance Theory", invented by Stephen Grossberg in 1976. ART represents a family of neural networks.

- ART encompasses a wide variety of neural networks.
  The basic ART System is an unsupervised learning model.

- The term "resonance" refers to resonant state of a neural network in which a category prototype vector matches close enough to the current input vector. ART matching leads to this resonant state, which permits learning. The network learns only in its resonant state.

- ART neural networks are capable of developing stable clusters of arbitrary sequences of input patterns by self-organizing.
  ART-1 can cluster binary input vectors.
  ART-2 can cluster real-valued input vectors.

- ART systems are well suited to problems that require online learning of large and evolving databases.

1. Adaptive Resonance Theory (ART)

Real world is faced with a situations where data is continuously changing. In such situation, every learning system faces plasticity-stability dilemma.

This dilemma is about *:*

"A system that must be able to learn to adapt to a changing environment (ie it must be plastic) but the constant change can make the system unstable, because the system may learn new information only by forgetting every thing it has so far learned."

This phenomenon, a contradiction between plasticity and stability, is called **plasticity - stability dilemma.**

The back-propagation algorithm suffer from such stability problem.

- Adaptive Resonance Theory (ART) is a new type of neural network, designed by Grossberg in 1976 to solve plasticity-stability dilemma.
- ART has a self regulating control structure that allows autonomous recognition and learning.
- ART requires no supervisory control or algorithmic implementation.

- **Recap** *(Supervised , Unsupervised and BackProp Algorithms)*

Neural networks learn through supervised and unsupervised means. The hybrid approaches are becoming increasingly common as well.

- In **supervised learning**, the input and the expected output of the system are provided, and the ANN is used to model the relationship between the two. Given an input set **x**, and a corresponding output set **y**, an optimal rule is determined such that: **y = f(x) + e** where, **e** is an approximation error that needs to be minimized. Supervised learning is useful when we want the network to reproduce the characteristics of a certain relationship.

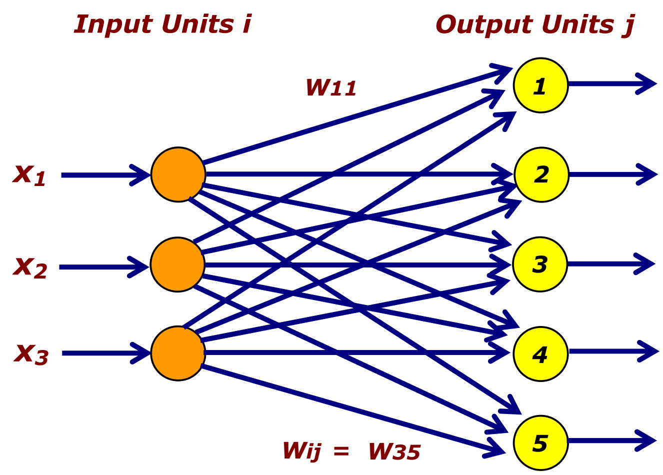
- In **unsupervised learning**, the data and a cost function are provided. The ANN is trained to minimize the cost function by finding a suitable input-output relationship. Given an input set **x**, and a cost function **g(x, y)** of the input and output sets, the goal is to minimize the cost function through a proper selection of **f** (the relationship between **x**, and **y**). At each training iteration, the trainer provides the input to the network, and the network produces a result. This result is put into the cost function, and the total cost is used to update the weights. Weights are continually updated until the system output produces a minimal cost. Unsupervised learning is useful in situations where a cost function is known, but a data set is not know that minimizes that cost function over a particular input space.

- In **backprop network learning,** a set of input-output pairs are given and the network is able to learn an appropriate mapping. Among the supervised learning, BPN is most used and well known for its ability to attack problems which we can not solve explicitly. However there are several technical problems with back-propagation type algorithms. They are not well suited for tasks where input space changes and are often slow to learn, particularly with many hidden units. Also the semantics of the algorithm are poorly understood and not biologically plausible, restricting its usefulness as a model of neural learning.

Most learning in brains is completely unsupervised.

• Competitive Learning Neural Networks

While no information is available about desired outputs the network updated weights only on the basis of input patterns. The Competitive Learning network is unsupervised learning for categorizing inputs. The neurons (or units) compete to fire for particular inputs and then learn to respond better for the inputs that activated them by adjusting their weights. An example of competitive learning network is shown below.



Competitive Learning Network

- All input units ***i*** are connected to all output units ***j*** with weights ***Wij*** .
- Number of inputs is the input dimension and the number of outputs is equal to the number of clusters that data are to be divided into.
- The network indicates that the 3-D input data are divided into **5** clusters.
- A cluster center position is specified by the weight vector connected to the corresponding output unit.
- The clusters centers, denoted as the weights, are updated via the competitive learning rule.

[Continued in next slide]

[Continued from previous slide]

- For an output unit j , the input vector $X = [x_1 , x_2 , x_3]^T$ and the weight vector $W_j = [w_{1j} , w_{1j} , w_{1j}]^T$ are normalized to unit length.
- The activation value a_j of the output unit j is calculated by the inner product of the weight vectors

$$a_j = \sum_{i=1}^{3} x_i \; w_{ij} = X^T W_j = W_j \; X^T$$

and then the output unit with the highest activation is selected for further processing; **this implied competitive**.

- Assuming that output unit k has the maximal activation, the weights leading to this unit are updated according to the competitive, called **winner-take-all (WTA)** learning rule

$$w_k (t + 1) = \frac{w_k (t) + \eta \{x (t) + w_k (t)\}}{||w_k (t) + \eta \{x (t) + w_k (t)\}||}$$

which is normalized to ensure that $w_k (t + 1)$ is always of unit length; only the weights at the winner output unit **k** are updated and all other weights remain unchanged.

- Alternately, Euclidean distance as a dissimilarity measure is a more general scheme of competitive learning, in which the activation of output unit j is as

$$a_j = \{ \sum_{i=1}^{3} (x_i - w_{ij})^2 \}^{1/2} = || \; x_i - w_{ij} \; ||$$

the weights of the output units with the smallest activation are updated according to

$$w_k (t + 1) = w_k (t) + \eta \{x (t) + w_k (t)\}$$

A competitive network, on the input patterns, performs an on-line clustering process and when complete the input data are divided into disjoint clusters such that similarity between individuals in the same cluster are larger than those in different clusters. Stated above, two metrics of similarity measures: one is Inner product and the other Euclidean distance. Other metrics of similarity measures can be used. The selection of different metrics lead to different clustering.

The limitations of competitive learning are stated in the next slide.

[Continued from previous slide]

Limitations of Competitive Learning :

- Competitive learning lacks the capability to add new clusters when deemed necessary.
- Competitive learning does not guarantee stability in forming clusters.
  - If the learning rate η is constant, then the winning unit that responds to a pattern may continue changing during training.
  - If the learning rate η is decreasing with time, it may become too small to update cluster centers when new data of different probability are presented.

Carpenter and Grossberg (1998) referred such occurrence as the stability-plasticity dilemma which is common in designing intelligent learning systems. In general, a learning system should be plastic, or adaptive in reacting to changing environments, and should be stable to preserve knowledge acquired previously.

- **Stability-Plasticity Dilemma (SPD)**

Every learning system faces the plasticity-stability dilemma.

The plasticity-stability dilemma poses few questions :

- How can we continue to quickly learn new things about the environment and yet not forgetting what we have already learned?
- How can a learning system remain plastic (adaptive) in response to significant input yet stable in response to irrelevant input?
- How can a neural network can remain plastic enough to learn new patterns and yet be able to maintain the stability of the already learned patterns?
- How does the system know to switch between its plastic and stable modes.
- What is the method by which the system can retain previously learned information while learning new things.

Answer to these questions, about plasticity-stability dilemma in learning systems is the Grossberg's Adaptive Resonance Theory (ART).

- ART has been developed to avoid stability-plasticity dilemma in competitive networks learning.
- The stability-plasticity dilemma addresses how a learning system can preserve its previously learned knowledge while keeping its ability to learn new patterns.
- ART is a family of different neural architectures. ART architecture can self-organize in real time producing stable recognition while getting input patterns beyond those originally stored.

2. Adaptive Resonance Theory (ART) Networks

An adaptive clustering technique was developed by Carpenter and Grossberg in 1987 and is called the Adaptive Resonance Theory (ART) .

The Adaptive Resonance Theory (ART) networks are self-organizing competitive neural network. ART includes a wide variety of neural networks. ART networks follow both supervised and unsupervised algorithms.

- The **unsupervised ARTs** named as ART1, ART2 , ART3, . . and are similar to many iterative clustering algorithms where the terms "nearest" and "closer" are modified by introducing the concept of "resonance". Resonance is just a matter of being within a certain threshold of a second similarity measure.
- The **supervised ART** algorithms that are named with the suffix "MAP", as ARTMAP. Here the algorithms cluster both the inputs and targets and associate two sets of clusters.

The basic ART system is unsupervised learning model. It typically consists of

- a comparison field and a recognition field composed of neurons,
- a vigilance parameter, and
- a reset module

Each of these are explained in the next slide.

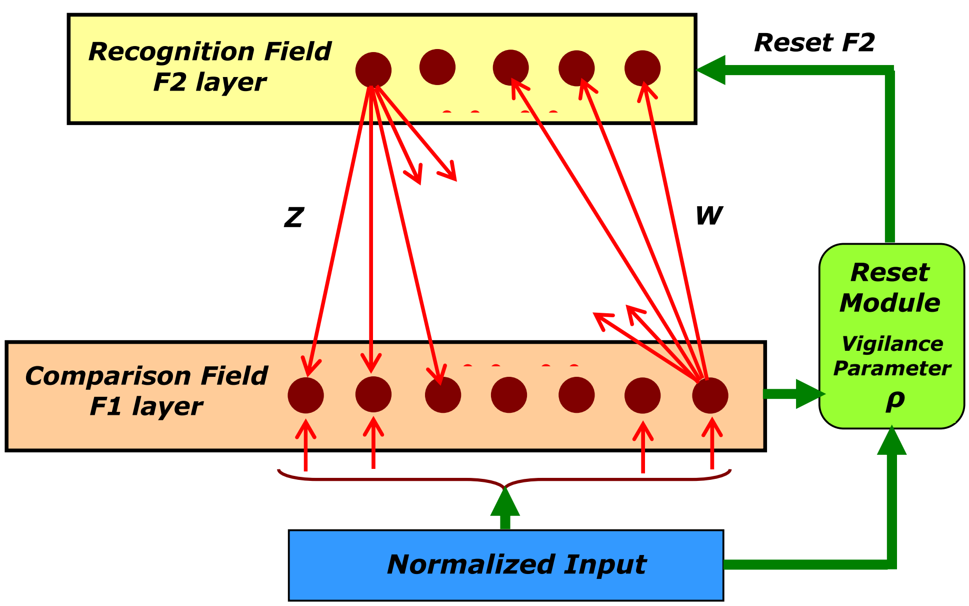


Fig Basic ART Structure

- **Comparison field**

  The comparison field takes an input vector (a one-dimensional array of values) and transfers it to its best match in the recognition field. Its best match is the single neuron whose set of weights (weight vector) most closely matches the input vector.

- **Recognition field**

  Each recognition field neuron, outputs a negative signal proportional to that neuron's quality of match to the input vector to each of the other recognition field neurons and inhibits their output accordingly. In this way the recognition field exhibits lateral inhibition, allowing each neuron in it to represent a category to which input vectors are classified.

- **Vigilance parameter**

  After the input vector is classified, a reset module compares the strength of the recognition match to a vigilance parameter. The vigilance parameter has considerable influence on the system:

  – Higher vigilance produces highly detailed memories (many, fine-grained categories), and

  – Lower vigilance results in more general memories (fewer, more-general categories).

- **Reset Module**

The reset module compares the strength of the recognition match to the vigilance parameter.

  - If the vigilance threshold is met, then training commences.
  - Otherwise, if the match level does not meet the vigilance parameter, then the firing recognition neuron is inhibited until a new input vector is applied;

Training commences only upon completion of a search procedure.
In the search procedure, the recognition neurons are disabled one by one by the reset function until the vigilance parameter is satisfied by a recognition match.

If no committed recognition neuron's match meets the vigilance threshold, then an uncommitted neuron is committed and adjusted towards matching the input vector.

2.1 Simple ART Network

ART includes a wide variety of neural networks. ART networks follow both supervised and unsupervised algorithms. The unsupervised ARTs as ART1, ART2, ART3, ... are similar to many iterative clustering algorithms.

The simplest ART network is a vector classifier. It accepts as input a vector and classifies it into a category depending on the stored pattern it most closely resembles. Once a pattern is found, it is modified (trained) to resemble the input vector. If the input vector does not match any stored pattern within a certain tolerance, then a new category is created by storing a new pattern similar to the input vector. Consequently, no stored pattern is ever modified unless it matches the input vector within a certain tolerance.

This means that an ART network has

- both plasticity and stability;
- new categories can be formed when the environment does not match any of the stored patterns, and
- the environment cannot change stored patterns unless they are sufficiently similar.

2.2 General ART Architecture

The general structure, of an ART network is shown below.

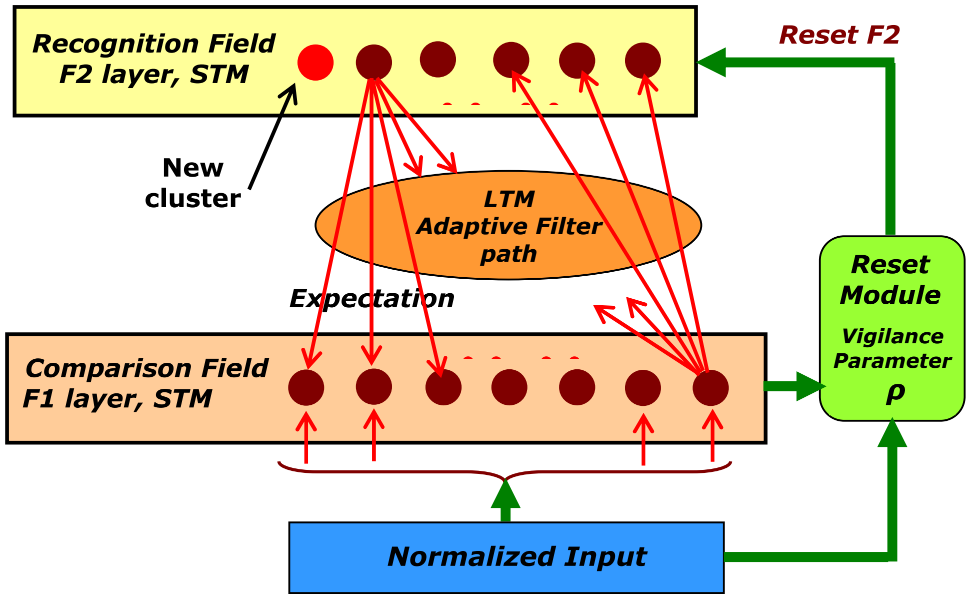


Fig Simplified ART Architecture

There are two layers of neurons and a reset mechanism.

- **F1 layer** : an input processing field; also called **comparison layer**.
- **F2 layer** : the cluster units ; also called **competitive layer**.
- **Reset mechanism** : to control the degree of similarity of patterns placed on the same cluster; takes decision whether or not to allow cluster unit to learn.

There are two sets of connections, each with their own weights, called :

- **Bottom-up weights** from each unit of layer **F1** to all units of layer **F2** .
- **Top-down weights** from each unit of layer **F2** to all units of layer **F1** .

2.3 Important ART Networks

The ART comes in several varieties. They belong to both unsupervised and supervised form of learning.

Unsupervised ARTs are named as ART1, ART2 , ART3, . . and are similar to many iterative clustering algorithms.

- ART1 model (1987) designed to cluster binary input patterns.
- ART2 model (1987) developed to cluster continuous input patterns.
- ART3 model (1990) is the refinement of these two models.

Supervised ARTs are named with the suffix "MAP", as ARTMAP, that combines two slightly modified ART-1 or ART-2 units to form a supervised learning model where the first unit takes the input data and the second unit takes the correct output data. The algorithms cluster both the inputs and targets, and associate the two sets of clusters.

Fuzzy ART and **Fuzzy ARTMAP** are generalization using fuzzy logic.

A taxonomy of important ART networks are shown below.

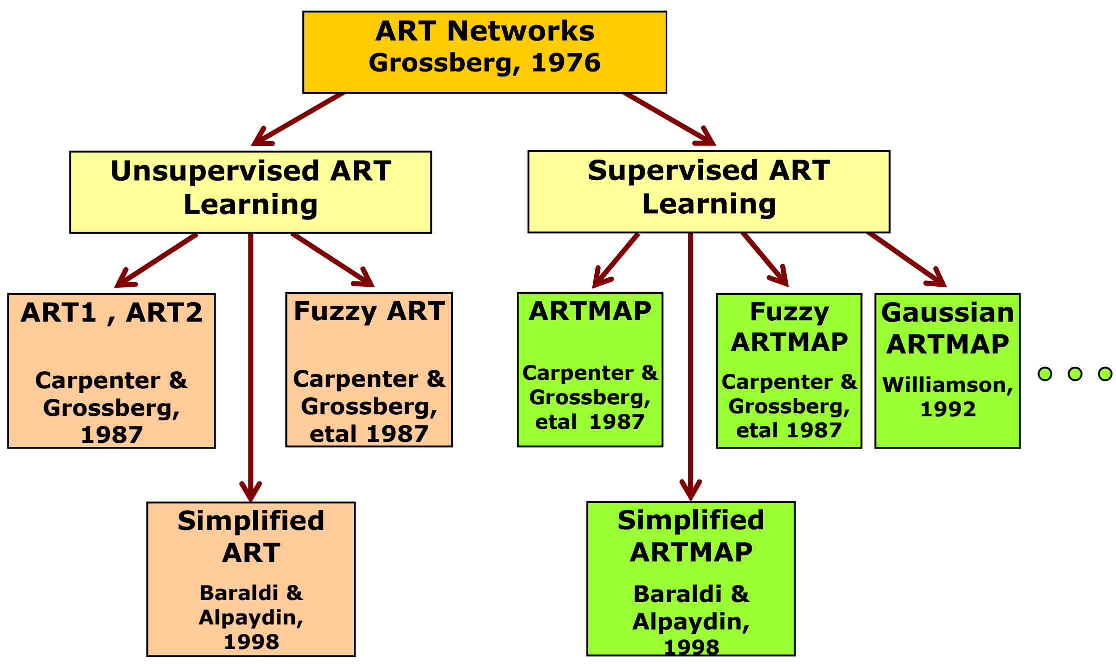


Fig. Important ART Networks

Note : Only the unsupervised ARTs are presented in what follows in the remaining slides.

2.4 Unsupervised ARTs – Discovering Cluster Structure

Human has ability to learn through classification. Human learn new concepts by relating them to existing knowledge and if unable to relate to something already known, then creates a new structure. The unsupervised ARTs named as ART1, ART2 , ART3, . . represent such human like learning ability.

ART is similar to many iterative clustering algorithms where each pattern is processed by

- Finding the "nearest cluster" seed/prototype/template to that pattern and then updating that cluster to be "closer" to the pattern;
- Here the measures "nearest" and "closer" can be defined in different ways in n-dimensional Euclidean space or an n-space.

How ART is different from most other clustering algorithms is that it is capable of determining number of clusters through adaptation.

- ART allows a training example to modify an existing cluster only if the cluster is sufficiently close to the example (the cluster is said to "resonate" with the example); otherwise a new cluster is formed to handle the example
- To determine when a new cluster should be formed, ART uses a vigilance parameter as a threshold of similarity between patterns and clusters.

ART networks can "discover" structure in the data by finding how the data is clustered. The ART networks are capable of developing stable clusters of arbitrary sequences of input patterns by self-organization.

Note : For better understanding, in the subsequent sections, first the iterative clustering algorithm (a non-neural approach) is presented then the ART1 and ART2 neural networks are presented.

3. Iterative Clustering - Non Neural Approach

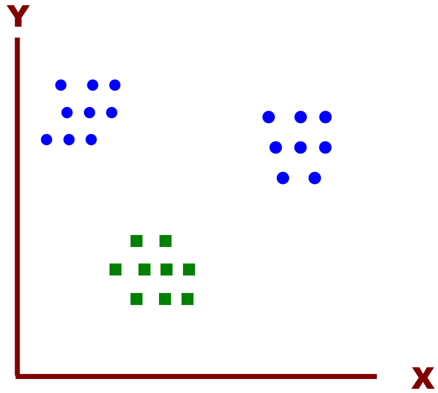
Organizing data into sensible groupings is one of the most fundamental mode of understanding and learning.

Clustering is a way to form 'natural groupings' or clusters of patterns. Clustering is often called an unsupervised learning.

- cluster analysis is the study of algorithms and methods for grouping, or clustering, objects according to measured or perceived intrinsic characteristics or similarity.
- Cluster analysis does not use category labels that tag objects with prior identifiers, i.e., class labels.
- The absence of category information, distinguishes the clustering (unsupervised learning) from the classification or discriminant analysis (supervised learning).
- The aim of clustering is exploratory in nature to find structure in data.

- **Example :**

Three natural groups of data points, that is three natural clusters.



In clustering, the task is to learn a classification from the data represented in an n-dimensional Euclidean space or an n-space.

– the data set is explored to find some intrinsic structures in them;

– no predefined classification of patterns are required;

The K-mean, ISODATA and Vector Quantization techniques are some of the decision theoretic approaches for cluster formation among unsupervised learning algorithms.

(Note : a recap of distance function in n-space is first mentioned and then vector quantization clustering is illustrated.)

Recap : Distance Functions

Vector Space Operations

Let **R** denote the field of real numbers.

For any non-negative integer **n**, the space of all n-tuples of real numbers forms an n-dimensional vector space over ***R***, denotes R^n.

An element of R^n is written as $X = (x_1, x_2, \ldots x_i \ldots, x_n)$, where x_i is a real number. Similarly the other element $Y = (y_1, y_2, \ldots y_i \ldots, y_n)$

The vector space operations on R^n are defined by

$X + Y = (x_1 + y_1, X_2 + y_2, \ldots, x_n + y_n)$ and

$aX = (ax_1, ax_2, \ldots, ax_n)$

The standard inner product, called dot product, on R_n, is given by

$X \bullet Y = \sum_{i=1}^{n} (x_1 y_1 + x_2 y_2 + \ldots + xn\ yn)$ is a real number.

The dot product defines a **distance function** (or metric) on R^n by

$$d(X, Y) = ||X - Y|| = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

The (interior) angle θ between **x** and **y** is then given by

$$\theta = \cos^{-1}\left(\frac{X \bullet Y}{||X||\ ||Y||} \right)$$

The dot product of **X** with itself is always non negative, is given by

$$||X|| = \sqrt{\sum_{i=1}^{n} (x_i - x_i)^2}$$

[Continued in next slide]

- **Euclidean Distance**

  It is also known as Euclidean metric, is the "ordinary" distance between two points that one would measure with a ruler.

  The Euclidean distance between two points

  $\mathbf{P} = (p_1, p_2, \ldots p_i \ldots, x_n)$ and

  $\mathbf{Q} = (q_1, q_2, \ldots q_i \ldots, q_n)$

  in Euclidean n-space , is defined as :

$$\sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \ldots + (p_n - q_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}$$

  Example : Three-dimensional distance

  For two 3D points,

  $\mathbf{P} = (p_x, p_y, \ldots p_z)$ and

  $\mathbf{Q} = (q_x, q_y, \ldots q_z)$

  The Euclidean 3-space , is computed as :

$$\sqrt{(p_x - q_x)^2 + (p_y - q_y)^2 + (p_z - q_z)^2}$$

3.1 Vector Quantization Clustering

The goal is to "discover" structure in the data by finding how the data is clustered. One method for doing this is called vector quantization for grouping feature vectors into clusters.

The Vector Quantization (VQ) is non-neural approach to dynamic allocation of cluster centers.

VQ is a non-neural approach for grouping feature vectors into clusters. It works by dividing a large set of points (vectors) into groups having approximately the same number of points closest to them. Each group is represented by its centroid point, as in k-means and some other clustering algorithms.

- **Algorithm for vector quantization**
  - To begin with, in VQ no cluster has been allocated; first pattern would hold itself as the cluster center.
  - When ever a new input vector X^p as p^{th} pattern appears, the Euclidean distance d between it and the j^{th} cluster center C_j is calculated as

$$d = | X^p - C_j | = \left[\sum_{i=1}^{N} (X_i^p - C_{ji})^2 \right]^{1/2}$$

  - The cluster closest to the input is determined as

$$| X^p - C_k | < | X^p - C_j | = \begin{cases} j = 1, \ldots, M \\ \textit{minimum} \\ j \neq k \end{cases}$$

    where M is the number of allotted clusters.

  - Once the closest cluster k is determined, the distance $| X^p - C_k |$ must be tested for the threshold distance ρ as
    1. $| X^p - C_k | < \rho$ pattern assigned to k^{th} cluster
    2. $| X^p - C_k | > \rho$ a new cluster is allocated to pattern p
  - update that cluster centre where the current pattern is assigned

$$C_x = (1/N_x) \sum_{x \in S_n} X$$

    where set X represents all patterns coordinates (x, y) allocated to that cluster (ie S_n) and N is number of such patterns.

- **Example 1 :** *(Ref previous slide)*

Consider 12 numbers of pattern points in Euclidean space.

Their coordinates **(X , Y)** are indicated in the table below.

Table Input pattern - coordinates of 12 points

Points	X	Y	Points	X	Y
1	2	3	7	6	4
2	3	3	8	7	4
3	2	6	9	2	4
4	3	6	10	3	4
5	6	3	11	2	7
6	7	3	12	3	7

Determine clusters using VQ, assuming the threshold distance = 2.0.

- Take a new pattern, find its distances from all the clusters identified,
- Compare distances w.r.t the threshold distance and accordingly decide cluster allocation to this pattern,
- Update the cluster center to which this new pattern is allocated,
- Repeat for next pattern.

Computations to form clusters

Input Pattern	Determining cluster closest to input pattern: Cluster 1 Distance	Cluster 1 center	Cluster 2 Distance	Cluster 2 center	Cluster 3 Distance	Cluster 3 center	Cluster no assigned to i/p pattern
1, (2,3)	0	(2 , 3)					1
2, (3,3)	1	(2.5 , 3)					1
3, (2,6)	3.041381		0	(2 , 6)			2
4, (3,6)	3.041381		1	(2.5 , 6)			2
5, (6,3)	4.5		5.408326		0	(6 , 3)	3
6, (7,3)	5.5		6.264982		1	(6.5 , 3)	3
7, (6,4)	3.640054		4.031128		1.118033	(6.333333, 3.3333)	3
8, (7,4)	4.609772		4.924428		0.942809	(6.5, 3.5)	3
9, (2,4)	1.11803	(2.33333, 3.33333)	2.06155		4.527692		1
10, (3,4)	0.942808	(2.5, 3.5)	2.0615528		3.5355339		1
11, (2,7)	3.5355339		1.1180339	(2.333333, 6.333333)	7.2629195		2
12, (3,7)	3.5707142		0.9428089	(2.5, 6.5)	4.9497474		2

The computations illustrated in the above table indicates :

- No of clusters **3**
- Cluster centers **C_1 = (2.5, 3.5) ; C_2 = (2.5, 6.5); C_3 = (6.5, 3.5).**
- Clusters Membership **S(1) = {P1, P2, P9, P10}; S(2) = {P3, P4, P11, P12};**
  S(3) = {P5, P6, P7, P8};

These results are graphically represented in the next slide

[Continued from previous slide]

Graphical Representation of Clustering

(Ref - Example -1 in previous slide)

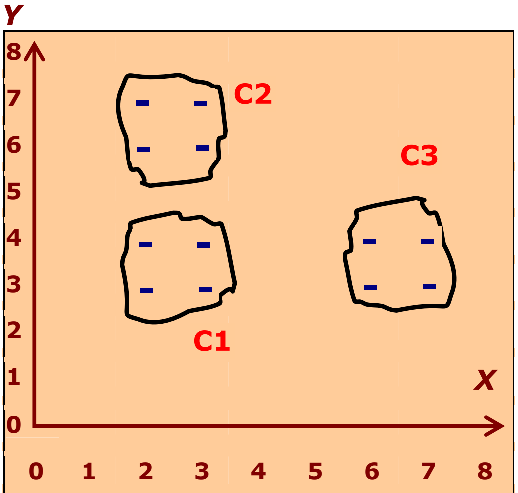


Fig (a) Input pattern for VQ , Threshold distance =2.0

Fig Clusters formed

Results of vector quantization :

Clusters formed

- Number of input patterns : 12
- Threshold distance assumed : 2.0
- No of clusters : 3
- Cluster centers :
  - C1 = (2.5, 3.5) ;
  - C2 = (2.5, 6.5);
  - C3 = (6.5, 3.5).
- Clusters Membership :
  - S(1)= {P1, P2, P9, P10};
  - S(2) = {P3, P4, P11, P12};
  - S(1) = {P5, P6, P7, P8};

Note : About threshold distance

- large threshold distance may obscure meaningful categories.
- low threshold distance may increase more meaningful categories.
- See next slide, clusters for threshold distances as **3.5** and **4.5** .

- **Example 2**

The input patterns are same as of Example 1.

Determine the clusters, assuming the threshold distance = 3.5 and 4.5.

- follow the same procedure as of Example 1 ;
- do computations to form clusters, assuming the threshold distances as **3.5** and **4.5**.
- The results are shown below.

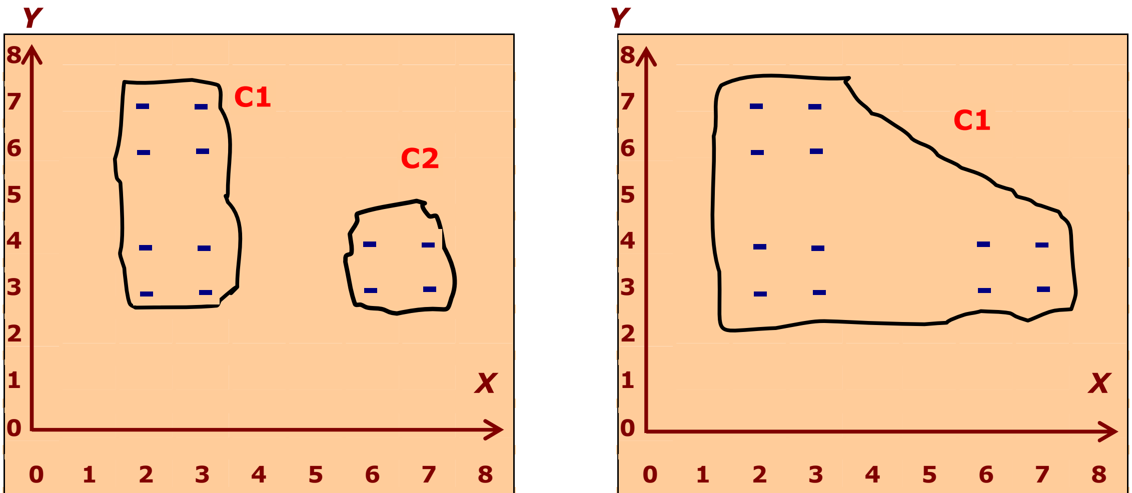


Fig (b) Input pattern for VQ , Threshold distance = 3.5

Fig (c) Input pattern for VQ , Threshold distance = 4.5

Fig Clusters formed

- Fig (b) for the threshold distance = **3.5** , two clusters formed.
- Fig (c) for the threshold distance = **4.5** , one cluster formed.

4. Unsupervised ART Clustering

The taxonomy of important ART networks, the basic ART structure, and the general ART architecture have been explained in the previous slides. Here only Unsupervised ART (ART1 and ART2) Clustering are presented.

ART1 is a clustering algorithm can learn and recognize binary patterns. Here

– similar data are grouped into cluster
– reorganizes clusters based upon the changes
– creates new cluster when different data is encountered

ART2 is similar to ART1, can learn and recognize arbitrary sequences of analog input patterns.

The ART1 architecture, the model description, the pattern matching cycle, and the algorithm - clustering procedure, and a numerical example is presented in this section.

4.1 ART1 Architecture

The Architecture of ART1 neural network consist of two layers of neurons.

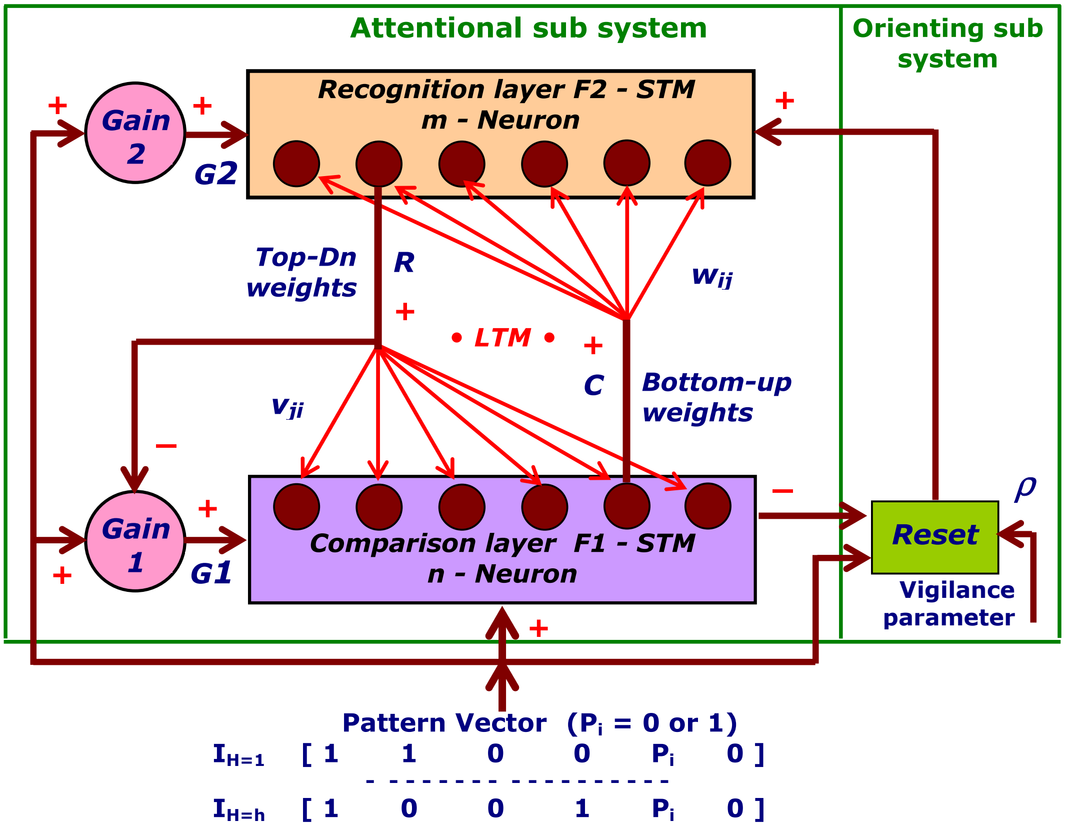


Fig. ART1 Network architecture

ATR1 model consists an "Attentional" and an "Orienting" subsystem.

The **Attentional sub-system** consists of :

- two competitive networks, as Comparison layer **F1** and Recognition layer **F2**, fully connected with top-down and bottom-up weights;
- two control gains, as **Gain1** and **Gain2.**

The **pattern vector** is input to comparison layer **F1**

The **Orienting sub-system** consists of :

- Reset layer for controlling the attentional sub-system overall dynamics based on vigilance parameter.
- Vigilance parameter ρ determines the degree of mismatch to be tolerated between the input pattern vectors and the weights connecting **F1** and **F2.**

The nodes at **F2** represent the clusters formed. Once the network stabilizes, the top-down weights corresponding to each node in **F2** represent the prototype vector for that node.

4.2 ART1 Model Description

The ART1 system consists of two major subsystem, an attentional subsystem and an orienting subsystem, described below. The system does pattern matching operation during which the network structure tries to determine whether the input pattern is among the patterns previously stored in the network or not.

- **Attentional Subsystem**

(a) **F1 layer** of neurons/nodes called or input layer or comparison layer; short term memory (**STM**).

(b) **F2 layer** of neurons/nodes called or output layer or recognition layer; short term memory (**STM**).

(c) **Gain control unit** , Gain1 and Gain2, one for each layer.

(d) **Bottom-up connections** from F1 to F2 layer ; traces of long term memory (**LTM**).

(e) **Top-down connections** from F2 to F1 layer; traces of long term memory (**LTM**).

(f) **Interconnections** among the nodes in each layer are not shown.

(g) **Inhibitory connection** (-ve weights) from F2 layer to gain control.

(h) **Excitatory connection** (+ve weights) from gain control to F1 and F2.

- **Orienting Subsystem**

(h) **Reset layer** for controlling the attentional subsystem overall dynamics.

(i) **Inhibitory connection** (-ve weights) from F1 layer to Reset node.

(j) **Excitatory connection** (+ve weights) from Reset node to F2 layer

- **Comparison F1 and Recognition F2 layers**

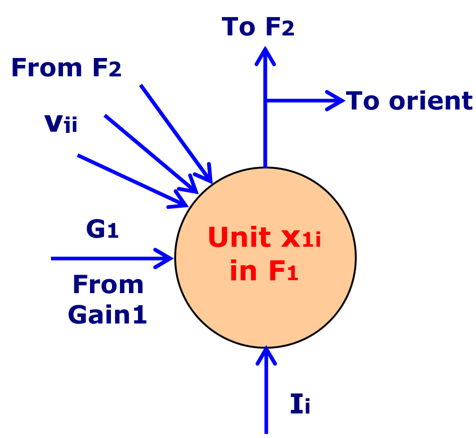
The comparison layer **F1** receives the binary external input, then **F1** passes the external input to recognition layer **F2** for matching it to a classification category. The result is passed back to **F1** to find :

If the category matches to that of input, then

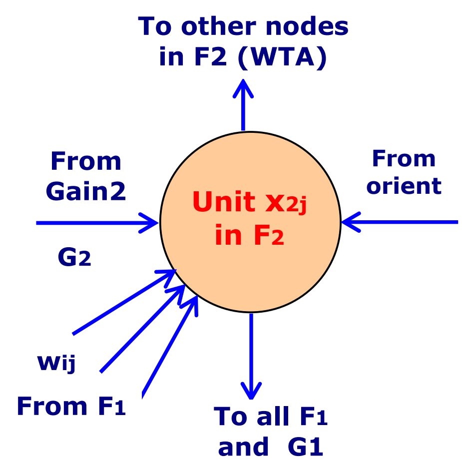
- If Yes (match) then a new input vector is read and the cycle starts again
- If No (mismatch) then the orienting system inhibits the previous category to get a new category match in **F2** layer.

The two gains, control the activities of **F1** and **F2** layer, respectively.

Processing element x_{1i} in layer F1



Processing element x_{2i} in layer F2



1. A processing element X_{1i} in F_1 receives input from three sources:
(a) External input vector I_i,
(b) Gain control signal G_1
(c) Internal network input v_{ji} made of the output from F_2 multiplied appropriate connections weights.

2. There is no inhibitory input to the neuron

3. The output of the neuron is fed to the F_2 layer as well as the orienting sub-system.

1. A processing element X_{2j} in F_2 receives input from three sources:
(a) Orienting sub-system,
(b) Gain control signal **G2**
(c) Internal network input w_{ij} made of the output from F_1 multiplied appropriate connections weights.

2. There is no inhibitory input to the neuron.

3. The output of the neuron is fed to the F_1 layer as well as G_1 control.

4.3 ART1 Pattern Matching Cycle

The ART network structure does pattern matching and tries to determine whether an input pattern is among the patterns previously stored in the network or not.

Pattern matching consists of : Input pattern presentation, Pattern matching attempts, Reset operations, and the Final recognition.

The step-by-step pattern matching operations are described below.

- **Fig (a) show the input pattern presentation.**

The sequence of effects are :

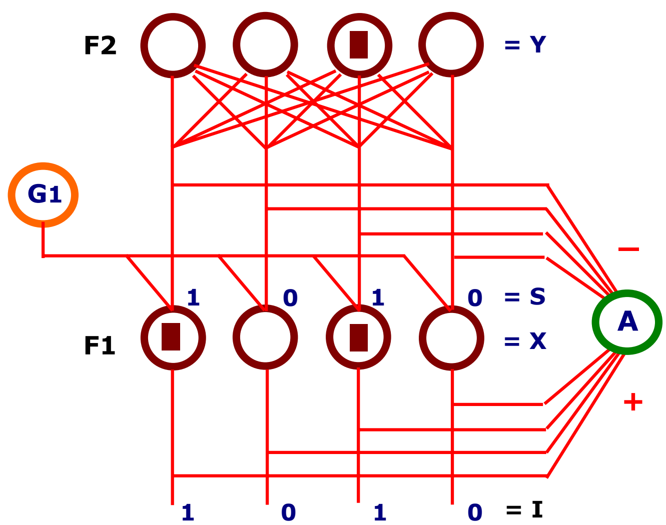


Fig (a) Input Pattern

► Input pattern **I** presented to the units in **F1** layer. A pattern of activation **X** is produced across **F1**.

► Same input pattern **I** also excites the orientation sub-system **A** and gain control **G1**.

► Output pattern **S** (which is inhibitory signal) is sent to **A**. It cancels the excitatory effect of signal **I** so that **A** remains inactive.

► Gain control **G1** sends an excitatory signal to **F1**. The same signal is applied to each node in **F1** layer. It is known as nonspecific signal.

► Appearance of **X** on **F1** results an output pattern **S**. It is sent through connections to **F_2** which receives entire output vector **S**.

► Net values calculated in the **F2** units, as the sum the product of the input values and the connection weights.

► Thus, in response to inputs from **F1**, a pattern of activity **Y** develops across the nodes of **F2** which is a competitive layer that performs a contrast enhancement on the input signal.

- **Fig (b) show the Pattern Matching Attempts.**

The sequence of operations are :

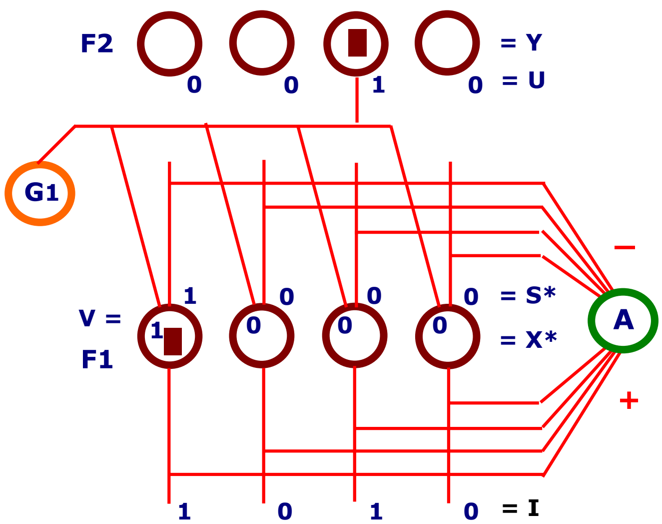


Fig (b) Pattern matching

► Pattern of activity **Y** results an output **U** from **F2** which is an inhibitory signal sent to **G1**. If it receives any inhibitory signal from **F2**, it ceases activity.

► Output **U** becomes second input pattern for **F1** units. Output **U** is transformed to pattern **V**, by **LTM** traces on the top-down connections from **F2** to **F1**.

► Activities that develop over the nodes in **F1** or **F2** layers are the **STM** traces not shown in the fig.

- **The 2/3 Rule**

► Among the three possible sources of input to **F1** or **F2**, only two are used at a time. The units on **F1** and **F2** can become active only if two out of the possible three sources of input are active. This feature is called the 2/3 rule.

► Due to the 2/3 rule, only those **F1** nodes receiving signals from both **I** and **V** will remain active. So the pattern that remains on **F1** is **I** ∩ **V** .

► The Fig shows patterns mismatch and a new activity pattern **X*** develops on **F1**. As the new output pattern **S*** is different from the original **S** , the inhibitory signal to **A** no longer cancels the excitation coming from input pattern **I**.

- **Fig (c) show the Reset Operations.**

The sequence of effects are :

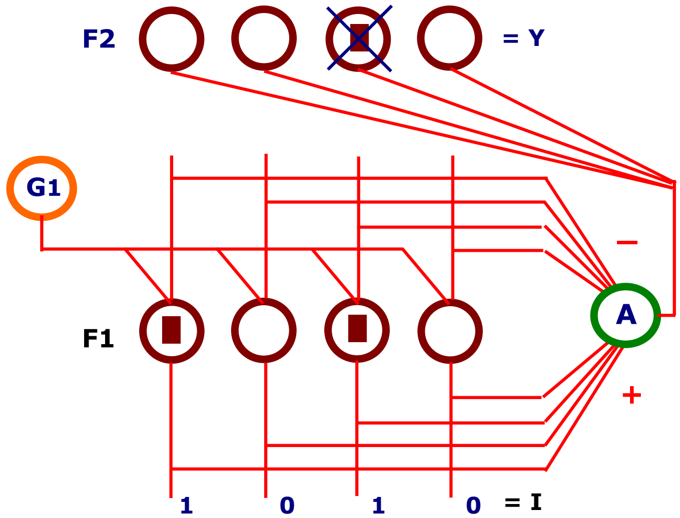


Fig (c) Reset

► Orientation sub-system **A** becomes active due to mismatch of patterns on **F1**.

► Sub-system **A** sends a non-specific reset signal to all nodes on **F2**.

► Nodes on **F2** responds according to their present state. If nodes are inactive, nodes do not respond; If nodes are active, nodes become inactive and remain for an extended period of time. This sustained inhibition prevents the same node winning the competition during the next cycle.

► Since output **Y** no longer appears, the top-down output and the inhibitory signal to **G1** also disappears.

- **Fig (d) show the final Recognition.**

The sequence of operations are :

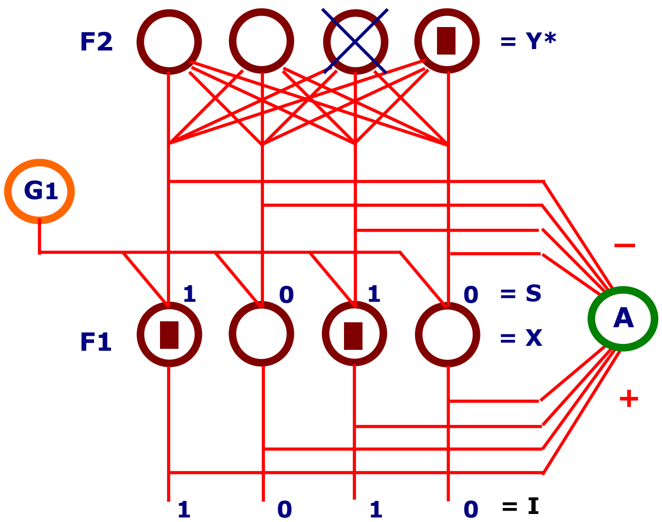


Fig (d) Final

► The original pattern **X** is reinstated on **F1** and a new cycle of pattern matching begins. Thus a new pattern **Y*** appears on **F2**.

► The nodes participating in the original pattern **Y** remains inactive due to long term effects of the reset signal from **A**.

► This cycle of pattern matching will continue until a match is found, or until **F2** runs out of previously stored patterns. If no match is found, the network will assign some uncommitted node or nodes on **F2** and will begin to learn the new pattern. Learning takes through **LTM** traces (modification of weights). This learning process does not start or stop but continue while the pattern matching process takes place. When ever signals are sent over connections, the weights associated with those connections are subject to modification.

► The mismatches do not result in loss of knowledge or learning of incorrect association because the time required for significant changes in weights is very large compared to the time required for a complete matching cycle. The connection participating in mismatch are not active long enough to effect the associated weights seriously.

► When a match occurs, there is no reset signal and the network settles down into a resonate state. During this stable state, connections remain active for sufficiently long time so that the weights are strengthened. This resonant state can arise only when a pattern match occurs or during enlisting of new units on **F2** to store an unknown pattern.

4.4 ART1 Algorithm - Clustering Procedure

(Ref: Fig ART1 Architecture, Model and Pattern matching explained before)

- **Notations**

  - $I(X)$ is input data set ( ) of the form $I(X) = \{ x(1), x(2), \ldots, x(t) \}$ where t represents time or number of vectors. Each $x(t)$ has n elements; Example $t = 4$, $x(4) = \{1 \;\; 0 \;\; 0\}^T$ is the 4th vector that has 3 elements .

  - $W(t) = (w_{ij}(t))$ is the Bottom-up weight matrix of type $n \times m$ where $i = 1, n$; $j = 1, m$; and its each column is a column vector of the form $w_j(t) = [(w_{1j}(t) \ldots\ldots w_{ij}(t) \ldots w_{nj}(t)]^T$, T is transpose; Example : Each column is a column vectors of the form

$$W(t) = (w_{ij}(t)) = \begin{array}{c} \begin{array}{cc} W_{j=1} & W_{j=2} \\ \downarrow & \downarrow \end{array} \\ \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \\ W_{31} & W_{32} \end{bmatrix} \end{array}$$

    $V(t) = (v_{ji}(t))$ is the Top-down weight matrix of type $m \times n$ where $j = 1, m$; $i = 1, n$; and its each line is a column vector of the form $v_j(t) = [(v_{j1}(t) \ldots\ldots v_{ji}(t) \ldots v_{jn}(t)]^T$, T is transpose; Example : Each line is a column vector of the form

$$v(t) = (v_{ji}(t)) = \begin{array}{c} \begin{array}{cc} v_{j=1} & v_{j=2} \end{array} \\ \begin{bmatrix} v_{11} & v_{12} & v_{13} \\ v_{21} & v_{22} & v_{23} \end{bmatrix} \end{array}$$

  - For any two vectors u and v belong to the same vector space R, say $u, v \in R$ the notation $< u, v > = u \cdot v = u^T \cdot v$ is scalar product; *and* $u \times v = (u_1 v_1, \ldots u_i v_i \ldots u_n v_n)^T \in R$, is piecewise product , that is component by component.

  - The $u \wedge v \in R^n$ means component wise minimum, that is the minimum on each pair of components $\min \{ u_i ; v_i \}$, $i = 1, n$;

  - The 1-norm of vector u is $||u||_1 = ||u|| = \sum_{i=1}^{n} | u_i |$

  - The vigilance parameter is real value $\rho \in (0, 1)$, The learning rate is real value $\alpha \in (0, 1)$,

- **Step-by-Step Clustering Procedure**

Input: Feature vectors

- Feature vectors $\mathbf{I}_{H=1 \text{ to } h}$, each representing input pattern to layer $\mathbf{F_1}$.
- Vigilance parameter ρ ; select value between **0.3** and **0.5**.

Assign values to control gains G_1 and G_2

$$G_1 = \begin{cases} 1 & \text{if input } I_H \neq 0 \text{ and output from } F_2 \text{ layer} = 0 \\ 0 & \text{otherwise} \end{cases}$$

$$G_2 = \begin{cases} 1 & \text{if input } I_H \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

Output: Clusters grouped according to the similarity is determined by ρ. Each neuron at the output layer represents a cluster, and the top-down (or backward) weights represents temp plates or prototype of the cluster.

Step - 1 (Initialization)

- Initially, no input vector **I** is applied, making the control gains, **G1 = 0, G2 = 0.** Set nodes in **F1** layer and **F2** layer to zero.

- Initialize bottom-up $w_{ij}(t)$ and top-down $v_{ji}(t)$ weights for time **t.**
  Weight w_{ij} is from neuron **i** in **F1** layer to neuron **j** in **F2** layer; where $i = 1, n;\ j = 1, m;$ and weight matrix $W(t) = (w_{ij}(t))$ is of type $n \times m$.
  Each column in $W(t)$ is a column vector $w_j(t),\ j = 1, m;$
  $w_j(t) = [(w_{1j}(t) \ldots \ldots w_{ij}(t) \ldots w_{nj}(t)]^T$, **T** is transpose and $w_{ij} = 1/(n+1)$ where **n** is the size of input vector;
  Example : If $n = 3$; then $w_{ij} = 1/4$

  column vectors

$$W(t) = (w_{ij}(t)) = \begin{array}{c} \begin{array}{cc} W_{j=1} & W_{j=2} \\ \downarrow & \downarrow \end{array} \\ \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \\ W_{31} & W_{32} \end{bmatrix} \end{array}$$

  The v_{ji} is weight from neuron **j** in **F2** layer to neuron **i** in **F1** layer; where $j = 1, m;\ i = 1, n;$ Weight matrix $V(t) = (v_{ji}(t))$ is of type $m \times n$.
  Each line in $V(t)$ is a column vector $V_j(t),\ j = 1, m;$
  $v_j(t) = [(v_{j1}(t) \ldots \ldots v_{ji}(t) \ldots v_{jn}(t)]^T$, **T** is transpose and $v_{ji} = 1$.

  Each line is a column vector

$$v(t) = (v_{ji}(t)) = \begin{array}{cc} v_{j=1} \rightarrow & \multirow{2}{*}{$\begin{bmatrix} v_{11} & v_{12} & v_{13} \\ v_{21} & v_{22} & v_{23} \end{bmatrix}$} \\ v_{j=2} \rightarrow & \end{array}$$

- Initialize the vigilance parameter, usually $0.3 \le \rho \le 0.5$

- Learning rate $\alpha = 0, 9$

- Special Rule : Example
  "While indecision, then the winner is second between equal".

Step - 2 (Loop back from step 8)

Repeat steps 3 to 10 for all input vectors I_H presented to the **F1** layer; that is **I(X) = { x(1), x(2), x(3), . . . , x(t), }**

Step – 3 (Choose input pattern vector)

Present a randomly chosen input data pattern, in a format as input vector.

Time t =1,

The First the binary input pattern say **{ 0 0 1 }** is presented to the network. Then

- As input **I ≠ 0 ,** therefore node **G1 = 1** and thus activates all nodes in **F1**.
- Again, as input **I ≠ 0** and from **F2** the output $X_2 = 0$ means not producing any output, therefore node **G2 = 1** and thus activates all nodes in **F2**, means recognition in **F2** is allowed.

Step - 4 (Compute input for each node in F2)

Compute input y_j for each node in **F2** layer using :

$$y_j = \sum_{i=1}^{n} I_i \times w_{ij}$$, If $j = 1, 2$ then $y_{j=1}$ and $y_{j=1}$ are

$$y_{j=1} = \begin{bmatrix} I_1 & I_2 & I_3 \end{bmatrix} \begin{bmatrix} W_{11} \\ W_{21} \\ W_{31} \end{bmatrix} \qquad y_{j=2} = \begin{bmatrix} I_1 & I_2 & I_3 \end{bmatrix} \begin{bmatrix} W_{12} \\ W_{22} \\ W_{32} \end{bmatrix}$$

Step – 5 (Select Winning neuron)

Find **k** , the node in **F2**, that has the largest y_k calculated in step 4.

$$y_k = \sum_{j=1}^{\text{no of nodes in } F_2} \max (y_j)$$

If an Indecision tie is noticed, then follow note stated below.
Else go to step 6.

Note :

Calculated in step 4, $y_{j=1} = 1/4$ and $y_{j=2} = 1/4$, are equal, means an indecision tie then go by some defined special rule.

Let us say the winner is the second node between the equals, i.e., $k = 2$.

Perform vigilance test, for the $F2_k$ output neuron , as below:

$$r = \frac{< V_k \ , \ X(t) >}{||X(t)||} = \frac{V_k^T \cdot X(t)}{||X(t)||}$$

If $r > \rho = 0.3$, means resonance exists and learning starts as :

The input vector $X(t)$ is accepted by $F2_{k=2}$.

Go to step 6.

Step – 6 (Compute activation in F1)

For the winning node **K** in **F2** in step 5, compute activation in **F1** as

$x_k^* = (x_1^* , x_2^* , \cdot \cdot \cdot , x_{i=n}^*)$ where $x_i^* = v_{ki} \times I_i$ is the piecewise product component by component and $i = 1, 2, . . . , n.$; i.e.,

$$x_K^* = (v_{k1} I_1 , . . . , v_{ki} I_i . ., v_{kn} I_n)^T$$

Step – 7 (Similarity between activation in F1 and input)

Calculate the similarity between x_k^* and input I_H using :

$$\frac{\| x_k^* \|}{\| I_H \|} = \frac{\sum_{i=1}^{n} x_i^*}{\sum_{i=1}^{n} I_i}$$

Example : If $x_{K=2}^* = \{0\ 0\ 1\}$, $I_{H=1} = \{0\ 0\ 1\}$

then similarity between x_k^* and input I_H is

$$\frac{\| x_{K=2}^* \|}{\| I_{H=1} \|} = \frac{\sum_{i=1}^{n} x_i^*}{\sum_{i=1}^{n} I_i} = 1$$

Step – 8 (Test similarity with vigilance parameter)

Test the similarity calculated in Step 7 with the vigilance parameter:

The similarity $\left(\dfrac{\| X^{*}_{K=2} \|}{\| I_{H=1} \|} \right) = 1$ is $> \rho$

It means the similarity between $X^{*}_{K=2}$, $I_{H=1}$ is **true.** Therefore,

Associate Input $I_{H=1}$ with **F2** layer node **m = k**

(a) Temporarily disable node k by setting its activation to **0**

(b) Update top-down weights , $v_j\,(t)$ of node $j = k = 2$, from **F2** to **F1**

$$v_{k\,i}\,(new) = v_{k\,i}\,(t) \times I_i \quad \text{where } i = 1, 2, \ldots, n \ ,$$

(c) Update bottom-up weights , $w_j\,(t)$ of node $j = k$, from **F2** to **F1**

$$w_{k\,i}\,(new) = \frac{v_{k\,i}\,(new)}{0.5 + \| v_{k\,i}\,(new) \|} \quad \text{where } i = 1, 2, \ldots, n$$

(d) Update weight matrix $W(t)$ and $V(t)$ for next input vector, time $t = 2$

$$v(t) = v(2) = (v_{ji}\,(2)) = \begin{array}{c} \\ \end{array}\begin{bmatrix} v11 & v12 & v13 \\ v21 & v22 & v23 \end{bmatrix} \begin{array}{l} \leftarrow vj{=}1 \\ \leftarrow vj{=}2 \end{array}$$

$$W(t) = W(2) = (w_{ij}\,(t)) = \begin{array}{cc} Wj{=}1 & Wj{=}2 \\ \downarrow & \downarrow \end{array} \atop \begin{bmatrix} W11 & W12 \\ W21 & W22 \\ W31 & W32 \end{bmatrix}$$

If done with all input pattern vectors t (1, n) then **STOP.**

else Repeat step 3 to 8 for next Input pattern

4.5 ART1 Numerical Example

- **Example : Classify in even or odd the numbers 1, 2, 3, 4, 5, 6, 7**

Input:

The decimal numbers **1, 2, 3, 4, 5, 6, 7** given in the BCD format.
This input data is represented by the set() of the form

I(X) = { x(1), x(2), x(3), x(4), x(5), x(6), x(7) } where

Decimal nos	BCD format	Input vectors x(t)
1	0 0 1	$x(1) = \{ 0\ 0\ 1\}^T$
2	0 1 0	$x(2) = \{ 0\ 1\ 0\}^T$
3	0 1 1	$x(3) = \{ 0\ 1\ 1\}^T$
4	1 0 0	$x(4) = \{ 1\ 0\ 0\}^T$
5	1 0 1	$x(5) = \{ 1\ 0\ 1\}^T$
6	1 1 0	$x(6) = \{ 1\ 1\ 0\}^T$
7	1 1 1	$x(7) = \{ 1\ 1\ 1\}^T$

- The variable **t** is time, here the natural numbers which vary from 1 to 7, is expressed as **t = 1 , 7** .
- The **x(t)** is input vector; **t = 1, 7** represents **7** vectors.
- Each **x(t)** has 3 elements, hence input layer **F1** contains **n= 3** neurons;
- let class A_1 contains even numbers and A_2 contains odd numbers, this means , two clusters, therefore output layer **F2** contains **m = 2** neurons.

Step - 1 (Initialization)

- Initially, no input vector **I** is applied, making the control gains, **G1 = 0, G2 = 0.** Set nodes in **F1** layer and **F2** layer to zero.

- Initialize bottom-up $w_{ij}(t)$ and top-down $v_{ji}(t)$ weights for time **t.**
  Weight w_{ij} is from neuron i in **F1** layer to neuron j in **F2** layer;
  where $i = 1, n$; $j = 1, m$; and
  weight matrix $W(t) = (w_{ij}(t))$ is of type **n x m.**
  Each column in $W(t)$ is a column vector $w_j(t)$, $j = 1, m$;
  $w_j(t) = [(w_{1j}(t) \ldots . w_{ij}(t) \ldots w_{nj}(t)]^T$, **T** is transpose and
  $w_{ij} = 1/(n+1)$ where n is the size of input vector;
  here $n = 3$; so $w_{ij} = 1/4$

  column vectors

$$W(t) = (w_{ij}(t)) = \begin{array}{cc} W_{j=1} & W_{j=2} \\ \downarrow & \downarrow \end{array} \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \\ W_{31} & W_{32} \end{bmatrix} = \begin{bmatrix} 1/4 & 1/4 \\ 1/4 & 1/4 \\ 1/4 & 1/4 \end{bmatrix} \quad \text{where } t=1$$

  The v_{ji} is weight from neuron j in **F2** layer to neuron i in **F1** layer;
  where $j = 1, m$; $i = 1, n$;
  weight matrix $V(t) = (v_{ji}(t))$ is of type **m x n.**
  Each line in $V(t)$ is a column vector $v_j(t)$, $j = 1, m$;
  $v_j(t) = [(v_{j1}(t) \ldots . v_{ji}(t) \ldots v_{jn}(t)]^T$, **T** is transpose and $v_{ji} = 1$.

  Each line is a column vector

$$v(t) = (v_{ji}(t)) = \begin{array}{c} v_{j=1} \rightarrow \\ v_{j=2} \rightarrow \end{array} \begin{bmatrix} v_{11} & v_{12} & v_{13} \\ v_{21} & v_{22} & v_{23} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad \text{where } t=1$$

- Initialize the vigilance parameter $\rho = 0.3$, usually $0.3 \le \rho \le 0.5$

- Learning rate $\alpha = 0, 9$

- Special Rule : While indecision , then the winner is second between equal.

Step - 2 (Loop back from step 8)

Repeat steps 3 to 10 for all input vectors $I_{H = 1 \text{ to } h=7}$ presented to the **F1** layer; that is $I(X) = \{ x(1), x(2), x(3), x(4), x(5), x(6), x(7) \}$

Step – 3 (Choose input pattern vector)

Present a randomly chosen input data in B C D format as input vector. Let us choose the data in natural order, say $x(t) = x(1) = \{ 0 \ 0 \ 1 \}^T$

Time t =1, the binary input pattern **{ 0 0 1 }** is presented to network.

- As input $I \neq 0$, therefore node $G1 = 1$ and thus activates all nodes in **F1**.
- Again, as input $I \neq 0$ and from **F2** the output $X_2 = 0$ means not producing any output, therefore node $G2 = 1$ and thus activates all nodes in **F2**, means recognition in **F2** is allowed.

Step - 4 (Compute input for each node in F2)

Compute input y_j for each node in **F2** layer using :

$$y_j = \sum_{i=1}^{n} I_i \times W_{ij}$$

$$y_{j=1} = \begin{bmatrix} I_1 & I_2 & I_3 \end{bmatrix} \begin{bmatrix} W_{11} \\ W_{21} \\ W_{31} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \end{bmatrix} = 1/4$$

$$y_{j=2} = \begin{bmatrix} I_1 & I_2 & I_3 \end{bmatrix} \begin{bmatrix} W_{12} \\ W_{22} \\ W_{32} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \end{bmatrix} = 1/4$$

Step – 5 (Select winning neuron)

Find k, the node in $F2$, that has the largest y_k calculated in step 4.

$$y_k = \sum_{j=1}^{\text{no of nodes in } F_2} \max(y_j)$$

If an indecision tie is noticed, then follow note stated below.

Else go to step 6.

Note :

Calculated in step 4, $y_{j=1} = 1/4$ and $y_{j=2} = 1/4$, are equal, means an indecision tie. *[Go by Remarks mentioned before, how to deal with the tie]*.

Let us say the winner is the second node between the equals, i.e., $k = 2$.

Perform vigilance test, for the $F2_k$ output neuron , as below:

$$r = \frac{< V_k , X(t) >}{||X(t)||} = \frac{V_k^T \cdot X(t)}{||X(t)||} = \frac{\begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}}{\sum_{i=1}^{n} |X(t)|} = \frac{1}{1} = 1$$

Thus $r > \rho = 0.3$, **means resonance exists and learning starts as** :

The input vector $X(t=1)$ is accepted by $F2_{k=2}$, ie $x(1) \in$ **A2** cluster.

Go to Step 6.

Step – 6 (Compute activation in F1)

For the winning node **K** in **F2** in step 5, compute activation in **F1** as

$x^*_k = (x^*_1 , x^*_2 , \cdot \cdot \cdot , x^*_{i=n})$ where $x^*_i = v_{ki} \times I_i$ is the piecewise product component by component and $i = 1, 2, . . . , n.$; i.e.,

$$x^*_K = (v_{k1} I_1 , . . . , v_{ki} I_i . ., v_{kn} I_n)^T$$

Accordingly $x^*_{K=2} = \{1\ 1\ 1\} \times \{0\ 0\ 1\} = \{0\ 0\ 1\}$

Step – 7 (Similarity between activation in F1 and input)

Calculate the similarity between x^*_k and input I_H using :

$$\frac{\| x^*_k \|}{\| I_H \|} = \frac{\sum_{i=1}^{n} x^*_i}{\sum_{i=1}^{n} I_i} \qquad \text{here } n = 3$$

Accordingly , while $x^*_{K=2} = \{0\ 0\ 1\}$, $I_{H=1} = \{0\ 0\ 1\}$

Similarity between x^*_k and input I_H is

$$\frac{\| x^*_{K=2} \|}{\| I_{H=1} \|} = \frac{\sum_{i=1}^{n} x^*_i}{\sum_{i=1}^{n} I_i} = 1$$

Step – 8 (Test similarity with vigilance parameter)

Test the similarity calculated in Step 7 with the vigilance parameter:

The similarity $\left(\dfrac{\| X^*_{K=2} \|}{\| I_{H=1} \|}\right) = 1$ is $> \rho$

It means the similarity between $X^*_{K=2}$, $I_{H=1}$ is **true**. Therefore,

Associate Input $I_{H=1}$ with **F2** layer node $m = k = 2$, i.e., **Cluster 2**

(a) Temporarily disable node k by setting its activation to **0**

(b) Update top-down weights , $v_j\ (t)$ of node $j = k = 2$, from **F2** to **F1**

$$v_{k\,i}\ (new) = v_{k\,i}\ (t=1) \times I_i \quad \text{where } i = 1, 2, \ldots, n = 3 ,$$

$$v_{k=2}\ (t=2) = v_{k=2,\,i}\ (t=1) \times I_i = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$$

(c) Update bottom-up weights , $w_j\ (t)$ of node $j = k = 2$, from **F2** to **F1**

$$w_{k\,i}\ (new) = \frac{v_{k\,i}\ (new)}{0.5 + \| v_{k\,i}\ (new) \|} \quad \text{where } i = 1, 2, \ldots, n = 3 ,$$

$$w_{k=2}\ (t=2) = \frac{v_{k=2,\,i}\ (t=2)}{0.5 + \| v_{k=2,\,i}\ (t=2) \|} = \frac{\begin{bmatrix} 0 & 0 & 1 \end{bmatrix}}{0.5 + \left\| \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \right\|}$$

$$= \begin{bmatrix} 0 & 0 & 2/3 \end{bmatrix}^T$$

(d) Update weight matrix $W(t)$ and $V(t)$ for next input vector, time $t = 2$

$$v(t) = v(2) = (v_{ji}\ (2)) = \begin{array}{c} v_{j=1} \\ v_{j=2} \end{array} \begin{bmatrix} v_{11} & v_{12} & v_{13} \\ v_{21} & v_{22} & v_{23} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

$$W(t) = W(2) = (w_{ij}\ (t)) = \begin{array}{c} \begin{array}{cc} W_{j=1} & W_{j=2} \end{array} \\ \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \\ W_{31} & W_{32} \end{bmatrix} \end{array} = \begin{bmatrix} 1/4 & 0 \\ 1/4 & 0 \\ 1/4 & 2/3 \end{bmatrix}$$

If done with all input pattern vectors t (1, 7) then stop.

else Repeat step 3 to 8 for next input pattern

- **Present Next Input Vector and Do Next Iteration (step 3 to 8)**

Time $t = 2$; $I_H = I_2 = \{0\ 1\ 0\}$;

$$W(t=2) = \begin{array}{cc} W_{j=1} & W_{j=2} \\ \downarrow & \downarrow \end{array} \begin{bmatrix} 1/4 & 0 \\ 1/4 & 0 \\ 1/4 & 2/3 \end{bmatrix} \qquad v(t=2) = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \begin{array}{l} v_{j=1} \\ v_{j=2} \end{array}$$

$$y_{j=1} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \end{bmatrix} = 1/4 = 0.25$$

$$y_{j=2} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 2/3 \end{bmatrix} = 0 = 0$$

Find winning neuron, in node in F2 that has $max(y_{j=1,m})$;

Assign $k = j$; i.e., $y_k = y_j = max(1/4, 0)$;

Decision $y_{j=1}$ is maximum, so $K = 1$

Do vigilance test , for output neuron $F2_{k=1}$,

$$r = \frac{V^T_{k=1} \cdot X(t=2)}{||X(t=2)||} = \frac{\begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}}{\sum_{i=1}^{n} |X(t=2)|} = \frac{1}{1} = 1$$

Resonance , since $r > \rho = 0.3$, resonance exists ; So **Start learning**;

Input vector $x(t=2)$ is accepted by $F2_{k=1}$, means $x(2) \in$ **A1 Cluster.**

Compute activation in **F1,** for winning node $k = 1$, piecewise product component by component

$$X^*_{K=1} = V_{k=1,i} \times I_{H=2,i} = (v_{k1} I_{H1}, \ldots v_{ki} I_{Hi} \ldots, v_{kn} I_{Hn})^T$$

$$= \{1\ 1\ 1\} \times \{0\ 1\ 0\} = \{0\ 1\ 0\}$$

Find similarity between $X^*_{K=1} = \{0\ 1\ 0\}$ and $I_{H=2} = \{0\ 1\ 0\}$ as

$$\frac{\| X^*_{K=1} \|}{\| I_{H=2} \|} = \frac{\sum_{i=1}^{n} X^*_i}{\sum_{i=1}^{n} I_{H=2,i}} = 1$$

[Continued in next slide]

[Continued from previous slide : Time t =2]

Test the similarity calculated with the vigilance parameter:

Similarity $\left(\dfrac{\| X^*_{K=1} \|}{\| I_{H=2} \|}\right) = 1$ is $> \rho$

It means the similarity between $X^*_{K=1}$, $I_{H=2}$ is **true.**

So Associate input $I_{H=2}$ with **F2** layer node $m = k = 1$, i.e., **Cluster 1**

(a) Temporarily disable node $k = 1$ by setting its activation to **0**

(b) Update top-down weights, $v_j\ (t=2)$ of node $j = k = 1$, from **F2** to **F1**

$$v_{k=1,\ i}\ (new) = v_{k=1,\ i}\ (t=2) \times I_{H=2,\ i} \quad \text{where } i = 1, 2, \ldots, n = 3,$$

$$v_{k=1,}\ (t=3) = v_{k=1,\ i}\ (t=2) \times I_{H=2,\ i} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T$$

(c) Update bottom-up weights, $w_j\ (t=2)$ of node $j = k = 1$, from **F1** to **F2**

$$w_{k=1,\ i}\ (new) = \frac{v_{k=1,\ i}\ (new)}{0.5 + ||\ v_{k=1,\ i}\ (new)\ ||} \quad \text{where } i = 1, 2, \ldots, n = 3,$$

$$w_{k=1,}\ (t=3) = \frac{v_{k=1,\ i}\ (t=3)}{0.5 + ||v_{k=1,\ i}\ (t=3)||} = \frac{\begin{bmatrix} 0 & 1 & 0 \end{bmatrix}}{0.5 + \left\| \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \right\|}$$

$$= \begin{bmatrix} 0 & 2/3 & 0 \end{bmatrix}^T$$

(d) Update weight matrix $W(t)$ and $V(t)$ for next input vector, time $t = 3$

$$V(t) = v(3) = (v_{ji}\ (3)) = \begin{array}{c} \\ v_{j=1} \\ v_{j=2} \end{array} \begin{bmatrix} v_{11} & v_{12} & v_{13} \\ v_{21} & v_{22} & v_{23} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$W(t) = W(3) = (w_{ij}\ (3)) = \begin{array}{cc} W_{j=1} & W_{j=2} \\ \downarrow & \downarrow \end{array} \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \\ W_{31} & W_{32} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 2/3 & 0 \\ 0 & 2/3 \end{bmatrix}$$

If done with all input pattern vectors t (1, 7) then stop.

else Repeat step 3 to 8 for next input pattern

- **Present Next Input Vector and Do Next Iteration (step 3 to 8)**

Time $t = 3$; $I_H = I_3 = \{ 0 \ 1 \ 1 \}$;

$$W(t=3) = \begin{array}{cc} W_{j=1} \downarrow & W_{j=2} \downarrow \\ \end{array} \begin{bmatrix} 0 & 0 \\ 2/3 & 0 \\ 0 & 2/3 \end{bmatrix} \qquad v(t=3) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{array}{l} \leftarrow v_{j=1} \\ \leftarrow v_{j=2} \end{array}$$

$$y_{j=1} = \begin{bmatrix} 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 2/3 \\ 0 \end{bmatrix} = 2/3 = 0.666$$

$$y_{j=2} = \begin{bmatrix} 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 2/3 \end{bmatrix} = 2/3 = 0.666$$

Find winning neuron, in node in F2 that has $max(y_{j=1,m})$;

Assign $k = j$; i.e., $y_k = y_j = max(2/3, \ 2/3)$; indecision tie;

take winner as second; $j = K = 2$

Decision $K = 2$

Do vigilance test , for output neuron $F2_{k=2}$,

$$r = \frac{V^T_{k=2} \cdot X(t=3)}{||X(t=3)||} = \frac{\begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}}{\sum_{i=1}^{n} |X(t=3)|} = \frac{1}{2} = 0.5$$

Resonance , since $r > \rho = 0.3$, resonance exists ; So **Start learning**;

Input vector $x(t=3)$ is accepted by $F2_{k=2}$, means $x(3) \in$ **A2 Cluster**.

Compute activation in **F1,** for winning node $k = 2$, piecewise product component by component

$$X^*_{K=2} = V_{k=2,\,i} \ x \ I_{H=3,\,i} = (v_{k1} I_{H1} , . . v_{ki} I_{Hi} . , v_{kn} I_{Hn})^T$$

$$= \{0 \ 0 \ 1\} \ x \ \{0 \ 1 \ 1\} = \{ \ 0 \ \ 0 \ \ 1 \ \}$$

Find similarity between $X^*_{K=2} = \{0 \ 1 \ 0\}$ and $I_{H=3} = \{0 \ 1 \ 1\}$ as

$$\frac{\| X^*_{K=2} \|}{\| I_{H=3} \|} = \frac{\sum_{i=1}^{n} X^*_i}{\sum_{i=1}^{n} I_{H=3,\,i}} = 1/2 = 0.5$$

[Continued in next slide]

[Continued from previous slide : Time t =3]

Test the similarity calculated with the vigilance parameter:

Similarity $\left(\dfrac{\| X^{*}_{K=1} \|}{\| I_{H=2} \|} \right) = 0.5$ is $> \rho$

It means the similarity between $X^{*}_{K=2}$, $I_{H=3}$ is **true**.

So Associate input $I_{H=3}$ with **F2** layer node $m = k = 2$, i.e., **Cluster 2**

(a) Temporarily disable node $k = 2$ by setting its activation to **0**

(b) Update top-down weights, $v_j\ (t=3)$ of node $j = k = 2$, from **F2** to **F1**

$$v_{k=2,\ i}\ (new) = v_{k=2,\ i}\ (t=3) \times I_{H=3,\ i} \quad \text{where } i = 1, 2, \ldots, n = 3,$$

$$v_{k=2,}\ (t=4) = v_{k=2,\ i}\ (t=3) \times I_{H=3,\ i} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$$

(c) Update bottom-up weights, $w_j\ (t=3)$ of node $j = k = 2$, from $F1$ to $F2$

$$w_{k=2,\ i}\ (new) = \frac{v_{k=2,\ i}\ (new)}{0.5 + \|\ v_{k=2,\ i}\ (new)\ \|} \quad \text{where } i = 1, 2, \ldots, n = 3,$$

$$w_{k=2,}\ (t=4) = \frac{v_{k=2,\ i}\ (t=4)}{0.5 + \|v_{k=2,\ i}\ (t=4)\|} = \frac{\begin{bmatrix} 0 & 0 & 1 \end{bmatrix}}{0.5 + \|\begin{bmatrix} 0 & 0 & 1 \end{bmatrix}\|}$$

$$= \begin{bmatrix} 0 & 0 & 2/3 \end{bmatrix}^T$$

(d) Update weight matrix $W(t)$ and $V(t)$ for next input vector, time $t = 4$

$$V(t) = v(3) = (v_{ji}\ (3)) = \begin{array}{c} \quad v_{j=1} \quad v_{j=2} \\ \begin{bmatrix} v_{11} & v_{12} & v_{13} \\ v_{21} & v_{22} & v_{23} \end{bmatrix} \end{array} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$W(t) = W(3) = (w_{ij}\ (3)) = \begin{array}{c} W_{j=1} \ \ W_{j=2} \\ \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \\ W_{31} & W_{32} \end{bmatrix} \end{array} = \begin{bmatrix} 0 & 0 \\ 2/3 & 0 \\ 0 & 2/3 \end{bmatrix}$$

If done with all input pattern vectors t (1, 7) then stop.

else Repeat step 3 to 8 for next input pattern

- **Present Next Input Vector and Do Next Iteration (step 3 to 8)**

Time $t = 4$; $I_H = I_4 = \{ 1 \; 0 \; 0 \}$;

$$W(t=3) = \begin{array}{cc} W_{j=1} & W_{j=2} \\ \downarrow & \downarrow \end{array} \quad \begin{bmatrix} 0 & 0 \\ 2/3 & 0 \\ 0 & 2/3 \end{bmatrix} \qquad v(t=3) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{array}{l} \leftarrow v_{j=1} \\ \leftarrow v_{j=2} \end{array}$$

$$y_{j=1} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 2/3 \\ 0 \end{bmatrix} = 0 \qquad y_{j=2} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 2/3 \end{bmatrix} = 0$$

Find winning neuron, in node in F2 that has $max(y_{j=1,m})$;

Assign $k = j$ for $y_j = max(0, \; 0)$; Indecision tie; Analyze both cases

Case 1 : Take winner as first; $j = K = 1$; Decision $K = 1$

Do vigilance test , for output neuron $F2_{k=1}$,

$$r = \frac{V^T_{k=1} \cdot X(t=4)}{||X(t=4)||} = \frac{\begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}}{\sum_{i=1}^{n} |X(t=4)|} = \frac{0}{1} = 0$$

Resonance , since $r < \rho = 0.3$, no resonance exists ;

Input vector $x(t=4)$ is not accepted by $F2_{k=1}$, means $x(4) \notin$ **A1** Cluster.

Put Output $O_1(t = 4) = 0$.

Case 2 : Take winner as second ; $j = K = 2$; Decision $K = 2$

Do vigilance test , for output neuron $F2_{k=2}$,

$$r = \frac{V^T_{k=2} \cdot X(t=4)}{||X(t=4)||} = \frac{\begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}}{\sum_{i=1}^{n} |X(t=4)|} = \frac{0}{1} = 0$$

Resonance , since $r < \rho = 0.3$, no resonance exists ;

Input vector $x(t=4)$ is not accepted by $F2_{k=2}$, means $x(4) \notin$ **A2** Cluster.

Put Output $O_2(t = 4) = 0$.

Thus Input vector $x(t=4)$ is **Rejected** by **F2** layer.

[Continued in next slide]

[Continued from previous slide : Time t =4]

Update weight matrix $W(t)$ and $V(t)$ for next input vector, time $t=5$

$$W(4) = W(3)\ ;\quad V(4) = V(3)\ ;\quad O_{(t=4)} = \{\,1 \quad 1\,\}^T$$

$$\begin{array}{c} \\ V(t) = v(4) = (v_{ji}(4)) \end{array} = \begin{array}{c} v_{j=1} \quad v_{j=2} \\ \begin{bmatrix} v_{11} & v_{12} & v_{13} \\ v_{21} & v_{22} & v_{23} \end{bmatrix} \end{array} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$W(t) = W(4) = (w_{ij}(4)) = \begin{array}{cc} W_{j=1} & W_{j=2} \\ \multicolumn{2}{c}{\begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \\ W_{31} & W_{32} \end{bmatrix}} \end{array} = \begin{bmatrix} 0 & 0 \\ 2/3 & 0 \\ 0 & 2/3 \end{bmatrix}$$

If done with all input pattern vectors t (1, 7) then stop.

else Repeat step 3 to 8 for next input pattern

- **Present Next Input Vector and Do Next Iteration (step 3 to 8)**

Time t =5; $I_H = I_5 = \{ 1 \ 0 \ 1 \}$;

$$W(t=5) = \begin{array}{cc} W_{j=1} & W_{j=2} \\ \downarrow & \downarrow \end{array} \begin{bmatrix} 0 & 0 \\ 2/3 & 0 \\ 0 & 2/3 \end{bmatrix} \qquad v(t=5) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{array}{l} \leftarrow v_{j=1} \\ \leftarrow v_{j=2} \end{array}$$

$$y_{j=1} = \begin{bmatrix} 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 2/3 \\ 0 \end{bmatrix} = 0 \qquad y_{j=2} = \begin{bmatrix} 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 2/3 \end{bmatrix} = 2/3$$

Find winning neuron, in node in F2 that has $max(y_{j=1,m})$;

Assign $k = j$; i.e., $y_k = y_j = \max(0, 2/3)$

Decision $y_{j=2}$ is maximum, so **K = 2**

Do vigilance test , for output neuron $F2_{k=2}$,

$$r = \frac{V^T_{k=2} \cdot X(t=5)}{||X(t=5)||} = \frac{\begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}}{\sum_{i=1}^{n} |X(t=5)|} = \frac{1}{2} = 0.5$$

Resonance , since $r > \rho = 0.3$, resonance exists ; So **Start learning**;

Input vector $x(t=5)$ is accepted by $F2_{k=2}$, means $x(5) \in$ **A2 Cluster.**

Compute activation in **F1,** for winning node **k = 2**, piecewise product component by component

$$X^*_{K=2} = V_{k=2,\,i} \ \text{x} \ I_{H=5,\,i} = (v_{k1} I_{H1} , . . v_{ki} I_{Hi} . , v_{kn} I_{Hn})^T$$

$$= \{0\ 0\ 1\} \ \text{x} \ \{1\ 0\ 1\} = \{ \ 0 \ 0 \ 1 \ \}$$

Find similarity between $X^*_{K=2} = \{0\ 0\ 1\}$ and $I_{H=5} = \{1\ 0\ 1\}$ as

$$\frac{\| X^*_{K=2} \|}{\| I_{H=5} \|} = \frac{\sum_{i=1}^{n} X^*_i}{\sum_{i=1}^{n} I_{H=5,\,i}} = 1/2 = 0.5$$

[Continued in next slide]

[Continued from previous slide : Time t =5]

Test the similarity calculated with the vigilance parameter:

Similarity $\left(\dfrac{\| X^*_{K=1} \|}{\| I_{H=2} \|}\right) = 0.5$ **is** $> \rho$

It means the similarity between $X^*_{K=2}$, $I_{H=5}$ is **true**.

So Associate input $I_{H=5}$ with **F2** layer node $m = k = 2$, i.e., **Cluster 2**

(a) Temporarily disable node $k = 2$ by setting its activation to **0**

(b) Update top-down weights, $v_j\ (t=5)$ of node $j = k = 2$, from **F2** to **F1**

$$v_{k=2,\ i}\ (new) = v_{k=2,\ i}\ (t=5) \times I_{H=5,\ i} \quad \text{where } i = 1, 2, \ldots, n = 3,$$

$$v_{k=2,}\ (t=6) = v_{k=2,\ i}\ (t=5) \times I_{H=5,\ i} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$$

(c) Update bottom-up weights, $w_j\ (t=5)$ of node $j = k = 2$, from $F1$ to $F2$

$$w_{k=2,\ i}\ (new) = \frac{v_{k=2,\ i}\ (new)}{0.5 + \|\ v_{k=2,\ i}\ (new)\ \|} \quad \text{where } i = 1, 2, \ldots, n = 3,$$

$$w_{k=2,}\ (t=6) = \frac{v_{k=2,\ i}\ (t=6)}{0.5 + \| v_{k=2,\ i}\ (t=5) \|} = \frac{\begin{bmatrix} 0 & 0 & 1 \end{bmatrix}}{0.5 + \left\| \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \right\|}$$

$$= \begin{bmatrix} 0 & 0 & 2/3 \end{bmatrix}^T$$

(d) Update weight matrix $W(t)$ and $V(t)$ for next input vector, time $t = 6$

$$V(t) = v(6) = (v_{ji}\ (6)) = \begin{array}{c} v_{j=1} \\ v_{j=2} \end{array} \begin{bmatrix} v11 & v12 & v13 \\ v21 & v22 & v23 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$W(t) = W(6) = (w_{ij}\ (6)) = \begin{array}{c} \begin{array}{cc} W_{j=1} & W_{j=2} \end{array} \\ \begin{bmatrix} W11 & W12 \\ W21 & W22 \\ W31 & W32 \end{bmatrix} \end{array} = \begin{bmatrix} 0 & 0 \\ 2/3 & 0 \\ 0 & 2/3 \end{bmatrix}$$

If done with all input pattern vectors t (1, 7) then stop.

else Repeat step 3 to 8 for next input pattern

- **Present Next Input Vector and Do Next Iteration (step 3 to 8)**

Time $t = 6$; $I_H = I_6 = \{ 1 \ 1 \ 0 \}$;

$$W(t=6) = \begin{array}{cc} W_{j=1} & W_{j=2} \\ \left[\begin{array}{cc} 0 & 0 \\ 2/3 & 0 \\ 0 & 2/3 \end{array}\right] \end{array} \qquad v(t=6) = \begin{array}{l} \\ \left[\begin{array}{ccc} 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right] \begin{array}{l} v_{j=1} \\ v_{j=2} \end{array} \end{array}$$

$$y_{j=1} = \begin{bmatrix} 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 2/3 \\ 0 \end{bmatrix} = 2/3 \qquad y_{j=2} = \begin{bmatrix} 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 2/3 \end{bmatrix} = 0$$

Find winning neuron, in node in F2 that has $\max(y_{j=1,m})$;

Assign $k = j$; i.e., $y_k = y_j = \max(2/3, \ 0)$

Decision $y_{j=1}$ is maximum, so $K = 1$

Do vigilance test, for output neuron $F2_{k=1}$,

$$r = \frac{V^T_{k=1} \cdot X(t=6)}{||X(t=6)||} = \frac{\begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}}{\sum_{i=1}^{n} |X(t=6)|} = \frac{1}{2} = 0.5$$

Resonance, since $r > \rho = 0.3$, resonance exists ; So **Start learning**;

Input vector $x(t=6)$ is accepted by $F2_{k=1}$, means $x(6) \in$ **A1 Cluster.**

Compute activation in **F1,** for winning node $k = 1$, piecewise product component by component

$$X^*_{K=1} = V_{k=1,\,i} \ \times \ I_{H=6,\,i} = (v_{k1} I_{H1}, \ldots v_{ki} I_{Hi}, \ldots, v_{kn} I_{Hn})^T$$

$$= \{0\ 1\ 0\} \ \times \ \{1\ 1\ 0\} = \{ \ 0 \ 1 \ 0 \ \}$$

Find similarity between $X^*_{K=1} = \{0\ 1\ 0\}$ and $I_{H=6} = \{1\ 1\ 0\}$ as

$$\frac{\| X^*_{K=1} \|}{\| I_{H=6} \|} = \frac{\sum_{i=1}^{n} X^*_i}{\sum_{i=1}^{n} I_{H=5,\,i}} = 1/2 = 0.5$$

[Continued in next slide]

[Continued from previous slide : Time t =6]

Test the similarity calculated with the vigilance parameter:

Similarity $\left(\dfrac{\| X^*_{K=1} \|}{\| I_{H=6} \|}\right) = 0.5$ **is** $> \rho$

It means the similarity between $X^*_{K=1}$, $I_{H=6}$ is **true**.

So Associate input $I_{H=6}$ with **F2** layer node $m = k = 1$, i.e., **Cluster 1**

(a) Temporarily disable node $k = 1$ by setting its activation to **0**

(b) Update top-down weights, $v_j\ (t=6)$ of node $j = k = 2$, from **F2** to **F1**

$$v_{k=1,\ i}\ (new) = v_{k=1,\ i}\ (t=6) \times I_{H=6,\ i} \quad \text{where } i = 1, 2, \ldots, n = 3\ ,$$

$$v_{k=1,}\ (t=7) = v_{k=1,\ i}\ (t=6) \times I_{H=6,\ i} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T$$

(c) Update bottom-up weights, $w_j\ (t=2)$ of node $j = k = 1$, from $F1$ to $F2$

$$w_{k=1,\ i}\ (new) = \frac{v_{k=1,\ i}\ (new)}{0.5 + \|\ v_{k=1,\ i}\ (new)\ \|} \quad \text{where } i = 1, 2, \ldots, n = 3\ ,$$

$$w_{k=1,}\ (t=7) = \frac{v_{k=1,\ i}\ (t=7)}{0.5 + \|v_{k=1,\ i}\ (t=7)\|} = \frac{\begin{bmatrix} 0 & 1 & 0 \end{bmatrix}}{0.5 + \left\|\begin{bmatrix} 0 & 1 & 0 \end{bmatrix}\right\|}$$

$$= \begin{bmatrix} 0 & 2/3 & 0 \end{bmatrix}^T$$

(d) Update weight matrix $W(t)$ and $V(t)$ for next input vector, time $t = 7$

$$V(t) = v(7) = (v_{ji}\ (7)) = \begin{array}{c} \\ v_{j=1} \\ v_{j=2} \end{array} \begin{bmatrix} v_{11} & v_{12} & v_{13} \\ v_{21} & v_{22} & v_{23} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$W(t) = W(7) = (w_{ij}\ (7)) = \begin{array}{cc} W_{j=1} & W_{j=2} \\ \downarrow & \downarrow \end{array} \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \\ W_{31} & W_{32} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 2/3 & 0 \\ 0 & 2/3 \end{bmatrix}$$

If done with all input pattern vectors t (1, 7) then stop.

else Repeat step 3 to 8 for next input pattern

- **Present Next Input Vector and Do Next Iteration (step 3 to 8)**

Time $t = 7$; $I_H = I_7 = \{1 \;\; 1 \;\; 1\}$;

$$W(t=7) = \begin{array}{cc} W_{j=1} & W_{j=2} \\ \downarrow & \downarrow \end{array} \begin{bmatrix} 0 & 0 \\ 2/3 & 0 \\ 0 & 2/3 \end{bmatrix} \qquad v(t=7) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{array}{l} \leftarrow v_{j=1} \\ \leftarrow v_{j=2} \end{array}$$

$$y_{j=1} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 2/3 \\ 0 \end{bmatrix} = 2/3 \qquad y_{j=2} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 2/3 \end{bmatrix} = 2/3$$

Find winning neuron, in node in F2 that has $max(y_{j=1,m})$;

Assign $k = j$; i.e., $y_k = y_j = \max(2/3 , 2/3)$; indecision tie;

take winner as second; $j = K = 2$

Decision $K = 2$

Do vigilance test , for output neuron $F2_{k=1}$,

$$r = \frac{V^T_{k=2} \cdot X(t=7)}{||X(t=7)||} = \frac{\begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}}{\sum_{i=1}^{n} |X(t=7)|} = \frac{1}{3} = 0.333$$

Resonance , since $r > \rho = 0.3$, resonance exists ; So **Start learning**;

Input vector $x(t=7)$ is accepted by $F2_{k=2}$, means $x(7) \in$ **A2 Cluster.**

Compute activation in **F1,** for winning node $k = 2$, piecewise product component by component

$$X^*_{K=2} = V_{k=2,\,i} \times I_{H=7,\,i} = (v_{k1} I_{H1} , . . v_{ki} I_{Hi} . , v_{kn} I_{Hn})^T$$

$$= \{0 \; 0 \; 1\} \times \{1 \; 1 \; 1\} = \{ \; 0 \;\; 0 \;\; 1 \; \}$$

Find similarity between $X^*_{K=2} = \{0 \; 0 \; 1\}$ and $I_{H=7} = \{1 \; 1 \; 1\}$ as

$$\frac{\| X^*_{K=2} \|}{\| I_{H=7} \|} = \frac{\sum_{i=1}^{n} X^*_i}{\sum_{i=1}^{n} I_{H=7,\,i}} = 1/3 = 0.333$$

[Continued in next slide]

[Continued from previous slide : Time t =7]

Test the similarity calculated with the vigilance parameter:

Similarity $\left(\dfrac{\| X^*_{K=2} \|}{\| I_{H=7} \|}\right) = 0.333$ **is** $> \rho$

It means the similarity between $X^*_{K=2}$, $I_{H=7}$ is **true**.

So Associate input $I_{H=7}$ with **F2** layer node $m = k = 2$, i.e., **Cluster 2**

(a) Temporarily disable node $k = 2$ by setting its activation to **0**

(b) Update top-down weights, $v_j\ (t=7)$ of node $j = k = 2$, from **F2** to **F1**

$$v_{k=2,\ i}\ (new) = v_{k=2,\ i}\ (t=7) \times I_{H=7,\ i} \quad \text{where } i = 1, 2, \ldots, n = 3,$$

$$v_{k=2,}\ (t=8) = v_{k=2,\ i}\ (t=7) \times I_{H=7,\ i} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$$

(c) Update bottom-up weights, $w_j\ (t=7)$ of node $j = k = 1$, from $F1$ to $F2$

$$w_{k=2,\ i}\ (new) = \frac{v_{k=2,\ i}\ (new)}{0.5 + \|\ v_{k=2,\ i}\ (new)\ \|} \quad \text{where } i = 1, 2, \ldots, n = 3,$$

$$w_{k=2,}\ (t=8) = \frac{v_{k=2,\ i}\ (t=8)}{0.5 + \|v_{k=2,\ i}\ (t=8)\|} = \frac{\begin{bmatrix} 0 & 0 & 1 \end{bmatrix}}{0.5 + \left\|\begin{bmatrix} 0 & 0 & 1 \end{bmatrix}\right\|}$$

$$= \begin{bmatrix} 0 & 0 & 2/3 \end{bmatrix}^T$$

(d) Update weight matrix $W(t)$ and $V(t)$ for next input vector, time $t = 8$

$$V(t) = v(8) = (v_{ji}\ (8)) = \begin{array}{c} \\ v_{j=1} \\ v_{j=2} \end{array} \begin{bmatrix} v_{11} & v_{12} & v_{13} \\ v_{21} & v_{22} & v_{23} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$W(t) = W(8) = (w_{ij}\ (8)) = \begin{array}{cc} W_{j=1} & W_{j=2} \end{array} \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \\ W_{31} & W_{32} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 2/3 & 0 \\ 0 & 2/3 \end{bmatrix}$$

If done with all input pattern vectors t (1, 7) then STOP.

else Repeat step 3 to 8 for next input pattern

[Continued from previous slide]

- **Remarks**

  The decimal numbers **1, 2, 3, 4, 5, 6, 7** given in the BCD format (patterns) have been classified into two clusters (classes) as even or odd.

  Cluster Class **A_1 = { X(t=2), X(t=2) }**

  Cluster Class **A_2 = { X(t=1), X(t=3) , X(t=3) , X(t=3) }**

  The network failed to classify **X(t=4)** and rejected it.

  The network has learned by the :

  - Top down weight matrix ***V(t)*** and
  - Bottom up weight matrix ***W(t)***

  These two weight matrices, given below, were arrived after all, 1 to 7, patterns were one-by-one input to network that adjusted the weights following the algorithm presented.

$$V(t) = v(8) = (v_{ji}\,(8)) = \begin{array}{c} \\ v_{j=1} \\ v_{j=2} \end{array}\begin{bmatrix} v_{11} & v_{12} & v_{13} \\ v_{21} & v_{22} & v_{23} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$W(t) = W(8) = (w_{ij}\,(8)) = \begin{array}{cc} W_{j=1} & W_{j=2} \\ \downarrow & \downarrow \end{array}\begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \\ W_{31} & W_{32} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 2/3 & 0 \\ 0 & 2/3 \end{bmatrix}$$

4.3 ART2

The Adaptive Resonance Theory (ART) developed by Carpenter and Grossberg designed for clustering binary vectors, called ART1 have been illustrated in the previous section.

They later developed ART2 for clustering continuous or real valued vectors. The capability of recognizing analog patterns is significant enhancement to the system. The differences between ART2 and ART1 are :

- The modifications needed to accommodate patterns with continuous-valued components.
- The F1 field of ART2 is more complex because continuous-valued input vectors may be arbitrarily close together. The F1 layer is split into several sunlayers.
- The F1 field in ART2 includes a combination of normalization and noise suppression, in addition to the comparison of the bottom-up and top-down signals needed for the reset mechanism.
- The orienting subsystem also to accommodate real-valued data.

The learning laws of ART2 are simple though the network is complicated.

5. References : Textbooks

1. *"Neural Network, Fuzzy Logic, and Genetic Algorithms - Synthesis and Applications", by S. Rajasekaran and G.A. Vijayalaksmi Pai, (2005), Prentice Hall, Chapter 5, page 117-154.*

2. *"Elements of Artificial Neural Networks", by Kishan Mehrotra, Chilukuri K. Mohan and Sanjay Ranka, (1996), MIT Press, Chapter 5, page 157-197.*

3. *"Fundamentals of Neural Networks: Architecture, Algorithms and Applications", by Laurene V. Fausett, (1993), Prentice Hall, Chapter 5, page 218-288.*

4. *"Neural Network Design", by Martin T. Hagan, Howard B. Demuth and Mark Hudson Beale, (1996) , PWS Publ. Company, Chapter 16-18, page 16-1 to 18-40.*

5. *"Pattern Recognition Using Neural and Functional Networks", by Vasantha Kalyani David, Sundaramoorthy Rajasekaran, (2008), Springer, Chapter 4, page 27-49*

6. *Related documents from open source, mainly internet. An exhaustive list is being prepared for inclusion at a later date.*