part-I: Unsupervised Learning Network

* Introduction :=

* Definition of unsupervised learning network

Unsupervised learning in artificial neural networks (ANNs) is a type of machine learning that uses unlabeled data to discover patterns and build models.

* How it works :=

In unsupervised learning, ANNs are given unlabeled data sets and left to find patterns and build models without human intervention. ANNs learn to categorize data by exploiting the distance between clusters within.

* Applications :=

1) Image recognition.
2) Customer segmentation.
3) Cross-selling strategies.
4) Handwriting recognition.
5) Speech recognition.

* fixed weight competitive Nets :=

1) During training process also the weights remains fixed in these competitive networks.

2) The idea of competition is used among neurons for enhancement of contrast in their activation functions.

* Types of fixed weight Competitive Networks :=

① Maxnet :=

1) Maxnet network was developed by Lippmann in 1987.

2) The Maxner serves as a subnet for picking the node whose input is larger.

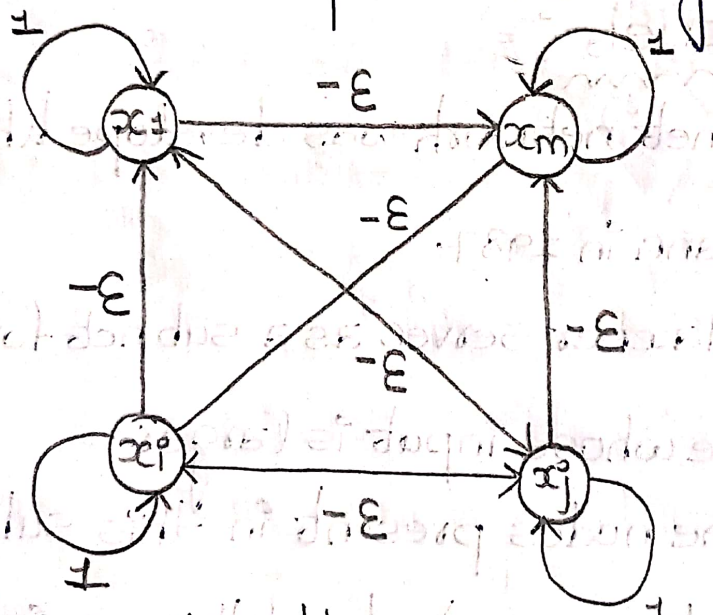
3) All the nodes present in this subnet are fully interconnected and there exist symmetrical weights in all these weighted interconnections.

* Architecture of Maxnet :=

1) The architecture of Maxnet is a fixed symmetrical weights are present over the weighted interconnections.

2) The weights between the neurons are inhibitory and fixed.

3) The Maxnet with this structure can be used as a subnet to select a particular node whose net input is the largest.



* Training Algorithm :=

1) Winner-Takes-All (WTA) : The neuron with the highest activation value wins the competition and is updated.

2) Lateral Inhibition : The winning neuron inhibits the other neurons.

preventing them from winning.

3) Weights Updating :- The weights are updated based on the winning neuron."

② Hamming Network :-

1) The Hamming network is a two-layer feedforward neural network for classification of binary bipolar n-tuple input vectors using minimum Hamming distance denoted as DH (Lippmann, 1987).

2) Hamming network finds the similarities between the input pattern and the weight vectors of all neurons.

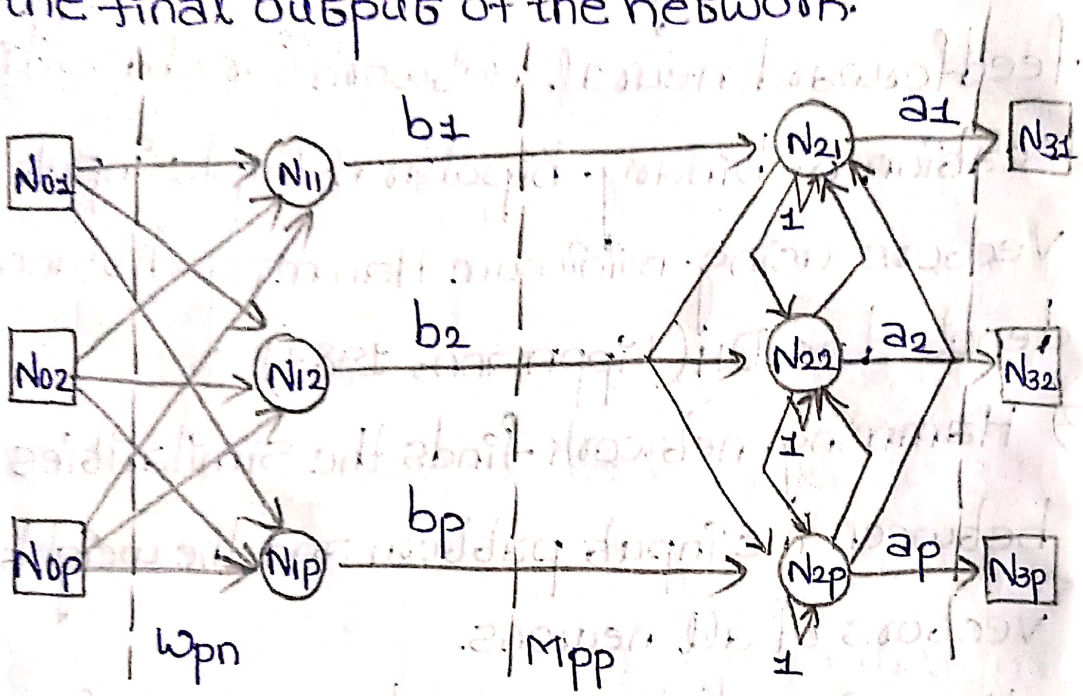
* Hamming distance :- = The number of different bits in two binary (or) bipolar vectors x_1 and x_2 is called the hamming distance between the vectors and is denoted by $H[X_1, X_2]$.

* ~~Architecture~~ of Hamming Network :-

1) Input Layer :- The input layer receives the input patterns.

2) Hidden Layer: The hidden layer consists of neurons that compute the Hamming distance between the input pattern and the stored patterns.

3) Output Layer: The output layer produces the final output of the network.



* Training :=

1) Unsupervised Learning: Hamming networks are trained using unsupervised learning, where the network learns to recognize patterns without labeled data.

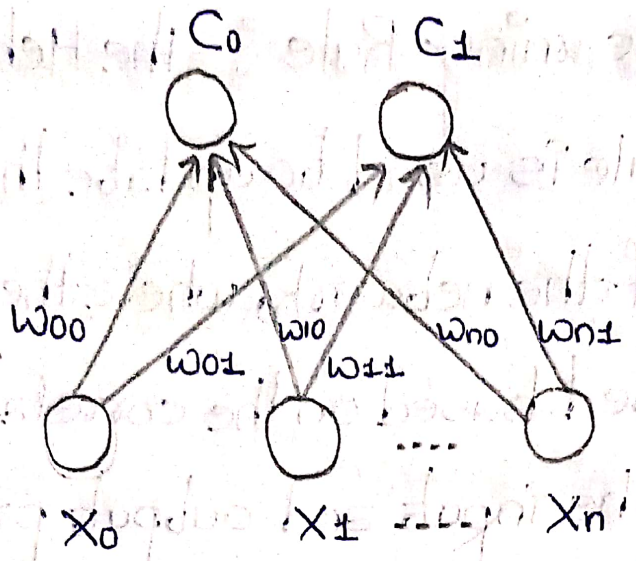
2) Hebbian Learning Rule :- The Hebbian learning rule is used to update the weights of the network, where the weights are adjusted based on the correlation between the input and output patterns.

* Kohonen self organizing feature maps :-

1) Kohonen self-organizing feature Maps (SOFMs) are a type of neural network that uses unsupervised learning to map high-dimensional input data to a lower-dimen-sional representation.

2) Kohonen self organizing feature map (KSOM) refers to a neural network, which is trained using competitive learning.

* Architecture of KSOFMs :- The architecture of the self organizing feature map with two clusters and n input features of any sample is given below :-



* How it works :=

1) Let's say an input data of size(m,n) where m is the number of training examples and n is the number of features in each example.
2) First, it initializes the weights of size (n,c) where C is the number of clusters.
3) Then iterating over the input data, for each training example, it updates the winning vector.
4) Weight updation rule is given by :

$$w_{ij} = w_{ij}(old) + alpha(t) * (x_i^k - w_{ij}(old))$$

where,

⇒ Alpha := Learning rate at time t
⇒ j := winning vector
⇒ i := ith feature of training example.
⇒ k := kth training example from the input data.

5) After training the KSOFM network, trained weights are used for clustering new examples, a new example falls in the cluster of winning vectors.

* Training Algorithm :-

1) Initialization : The weights are initialized randomly.

2) Competition : The neurons in the competitive layer compete to represent the input data.

3) Cooperation : The winning neuron and its neighbours are updated to represent the input data.

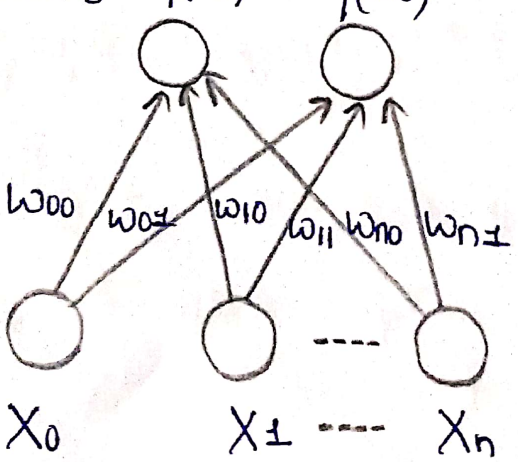
4) Convergence : The network converges to a stable, where the neurons represent the input data.

* Learning Vector Quantization (LVQ) :=

1) Learning Vector Quantization (LVQ) is a type of Artificial Neural Network which also inspired by biological models of neural systems.

2) It is based on prototype supervised learning classification algorithm and trained its network through a competitive learning algorithm similar to self Organizing Map.

* Architecture of Learning Vector Quantization (LVQ) :- The architecture of the learning vector quantization with the number of classes in an input data and n number of input features for any sample is given below :-



* How it works :-

1) Let's say that an input data of size(m,n) where m is the number of training examples and n is the number of features in each example and a label vector of size(m, 1).

2) First, it initializes the weights of size (n,c) from the first c number of training samples with different labels and should be discarded from all training samples.

3) Here, c is the number of classes, then iterate over the remaining input data, for each training example, it updates the winning vector.

4) The weight updation rule is given by :-

if correctly-classified :

$$w_{ij}(new) = w_{ij}(old) + alpha(t) * (x_i^k - w_{ij}(old))$$

else :

$$w_{ij}(new) = w_{ij}(old) - alpha(t) * (x_{ik} - w_{ij}(old))$$

where,

⇒ alpha :- Learning rate at time t

⇒ j :- winning vector

⇒ i :- ith feature of training example

⇒ k :- kth training example from the input data.

5) After training the LVQ network, trained weights are used for classifying new examples, a new example is labelled with the class of the winning vector.

* Training Algorithm :=

1) Initialization : Initialize the weight vectors randomly (or) using a predefined initialization method.
2) Training Data presentation : present the training data to the network, one sample at a time.
3) Winner selection : select the winner using a distance metric such as Euclidean distance.
4) Weight Update : Update the winner's weight vector using the LVQ learning rule.
5) Repetition : Repeat steps 2-4 for all training samples.
6) Convergence : stop training when convergence is reached, typically when the weight updates become negligible.

* Counter propagation Networks :=

1) Counter-propagation networks are multilayer networks based on a combination of input, clustering and output layers.
2) It can be used to compress data, to approximate functions (or) to associate patterns.

* Types of counter propagation network :=

① Full counter propagation network :=

1) Full counter propagation network efficiently represents a large number of vector pair x:y by adaptively constructing a look-up-table.
2) The full counter propagation Network works best if the inverse function exists and is continuous.
3) The vector x and y propagate through the network in a counterflow manner to yield output vector x* and y*.

② Forward-only counter propagation network :=

1) A simplified version of full counter propaga-tion network is the forward-only CPN.
2) Forward-only CPN uses only the x vector to form the cluster on the kohonen units during phase I training.
3) In case of forward-only CPN, first input vectors are presented to the input units.
4) First, the weights between the input layer and cluster layer are trained.
5) then the weights between the cluster layer and output layer are trained.
6) this is a specific competitive network, with target known.

* Architecture :=

1) Input Layer : The input layer receives the input data.
2) Kohonen Layer : The kohonen layer is a self-organizing map that clusters the input data.

3) Grossberg Layer: The Grossberg layer is a multilayer perceptron that learns to map the clustered data to the output.

* Training Algorithm :=

1) Initialization: Initialize the weights of the kohonen and Grossberg layers.
2) Kohonen Learning: Train the kohonen layer using the self-organizing map algorithm.
3) Grossberg Learning: Train the Grossberg layer using the multilayer perceptron algorithm.

* Features :=

1) Hybrid Architecture: CPNs combine the features of SOMs and MLPs.
2) Unsupervised Learning: CPNs use unsupervised learning in the kohonen layer.
3) Supervised learning: CPNs use Supervised learning in the Grossberg layer.

* Adaptive Resonance Theory (ART) :=

1) The term "adaptive" and "resonance" used in this suggests that they are open to new learning (i.e. adaptive) without discarding the previous (or) the old informa-tion (i.e. resonance).

2) The ART networks are known to solve the stability-plasticity dilemma i.e., stability refers to their nature of memorizing the learning and plasticity refers to the fact that they are flexible to gain new informa-tion.

* Types of ART :=

① ART1 := It is the simplest and the basic ART architecture. It is capable of clustering binary input values.

② ART2 := It is extension of ART1 that is capable of clustering continuous-valued input data.

③ fuzzy ART :- It is the augmentation of fuzzy logic and ART.

④ ARTMAP :- It is a supervised form of ART learning where one ART learns based on the previous ART module, it is also known as predictive ART.

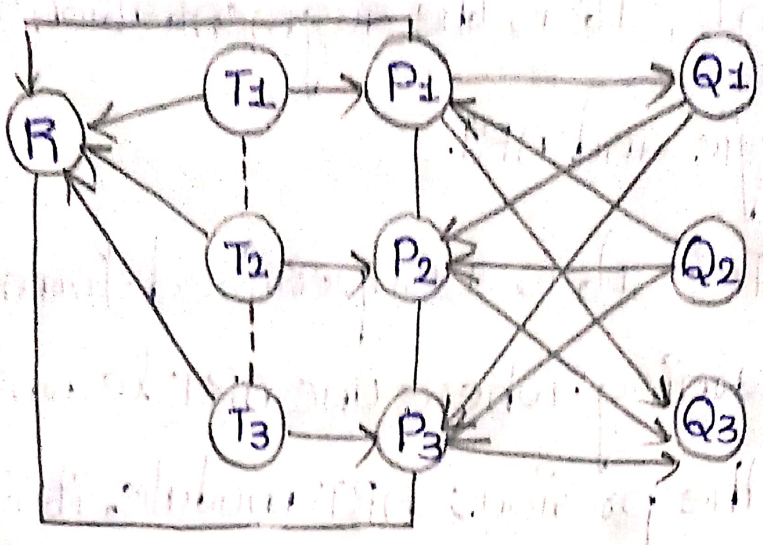
⑤ FARTMAP : this is a supervised ART architecture with fuzzy logic included.

* Basics of Adaptive Resonance Theory (ART) :-

1) f_1 Layer :- f_1 layer (or) the comparsion field (where the inputs are processed.

2) f_2 Layer :- f_2 layer (or) the recognition field (which consists of the clustering units)

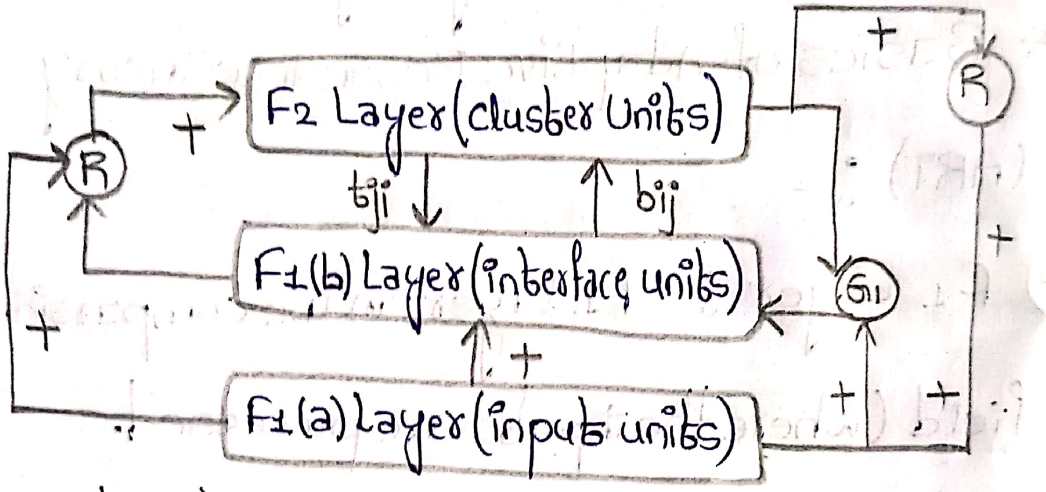
3) The Reset Module :- The Reset Module (that acts as a control mechanism).



F1(a) Layer inputs portion

F1(b) Layer interface portion

F2(c) Layer cluster Unit



* Advantages :-

1) It exhibits stability and is not disturbed by a wide variety of inputs provided to its network.

2) It can be integrated and used with various other techniques to give more good results.

* Special Networks :=

① Bayesian belief network :- Bayesian belief network describes the probability distribution governing a set of variables by specifying a set of conditional independence assumptions along with a set of conditional probabilities.

② Radial basis function network :=

1) Radial basis function (RBF) network is an artificial neural network that uses radial basis functions as activation functions

2) The output of the network is a linear combination of radial basis functions of the inputs and neuron parameters.

3) RBF networks form a special class of neural networks, which consists of three layers.

4) The input layer is used only to connect the network to its environment.

5) The output layer is linear and serves as a summation unit.