Artificial Neural Network(ANN)

Introduction

- Neural networks are parallel computing devices, which is basically an attempt to make a computer model of the brain.
- The main objective is to develop a system to perform various computational tasks faster than the traditional systems.

What is Artificial Neural Network?

- Artificial Neural Network (ANN) is an efficient computing system whose central theme is borrowed from the analogy of biological neural networks.
- ANNs are also named as "artificial neural systems," or "parallel distributed processing systems," or "connectionist systems."
- ANN acquires a large collection of units that are interconnected in some pattern to allow communication between the units.
- These units, also referred to as nodes or neurons, are simple processors which operate in parallel.
- Every neuron is connected with other neuron through a connection link.
- Each connection link is associated with a weight that has information about the input signal.
- This is the most useful information for neurons to solve a particular problem because the weight usually excites or inhibits the signal that is being communicated.
- Each neuron has an internal state, which is called an activation signal.
- Output signals, which are produced after combining the input signals and activation rule, may be sent to other units.

A Brief History of ANN

- The history of ANN can be divided into the following three eras −
- ANN during 1940s to 1960s

Some key developments of this era are as follows −

- 1943 − It has been assumed that the concept of neural network started with the work of physiologist, Warren McCulloch, and mathematician, Walter Pitts, when in 1943 they modeled a simple neural network using electrical circuits in order to describe how neurons in the brain might work.

- 1949 − Donald Hebb's book, *The Organization of Behavior*, put forth the fact that repeated activation of one neuron by another increases its strength each time they are used.
- 1956 − An associative memory network was introduced by Taylor.
- 1958 − A learning method for McCulloch and Pitts neuron model named Perceptron was invented by Rosenblatt.
- 1960 − Bernard Widrow and Marcian Hoff developed models called "ADALINE" and "MADALINE."

ANN during 1960s to 1980s

Some key developments of this era are as follows −

- 1961 − Rosenblatt made an unsuccessful attempt but proposed the "backpropagation" scheme for multilayer networks.
- 1964 − Taylor constructed a winner-take-all circuit with inhibitions among output units.
- 1969 − Multilayer perceptron (MLP) was invented by Minsky and Papert.
- 1971 − Kohonen developed Associative memories.
- 1976 − Stephen Grossberg and Gail Carpenter developed Adaptive resonance theory.

ANN from 1980s till Present

Some key developments of this era are as follows −

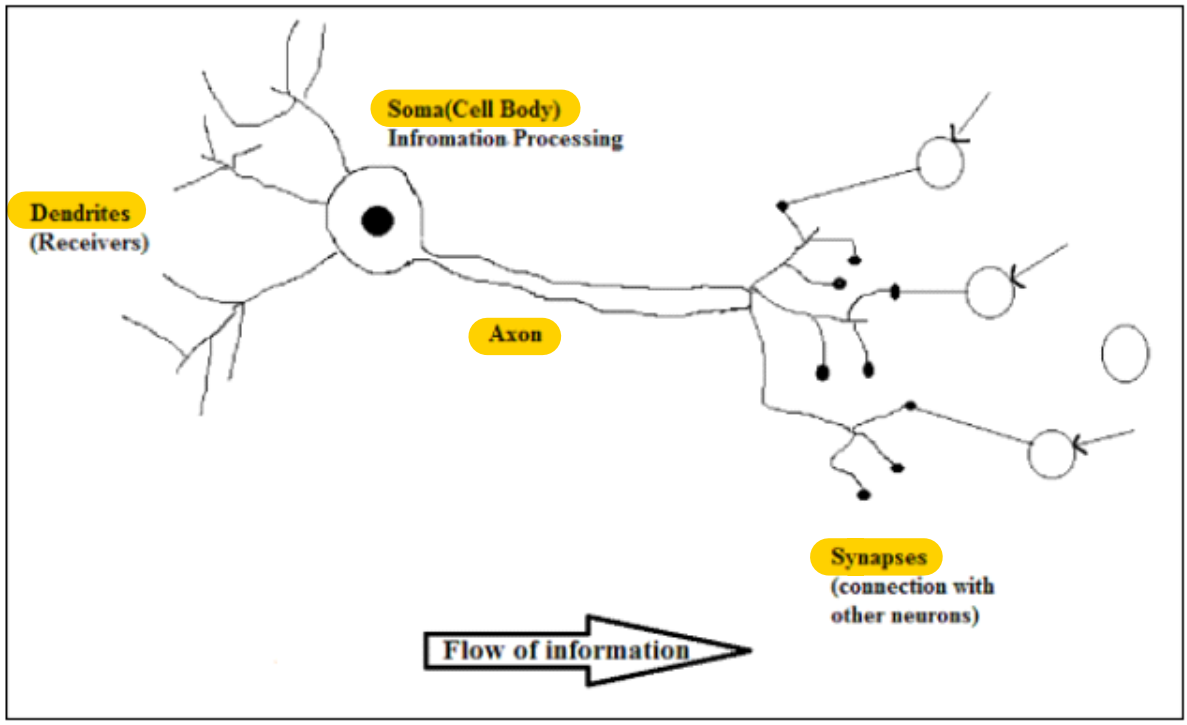
- 1982 − The major development was Hopfield's Energy approach.
- 1985 − Boltzmann machine was developed by Ackley, Hinton, and Sejnowski.
- 1986 − Rumelhart, Hinton, and Williams introduced Generalised Delta Rule.
- 1988 − Kosko developed Binary Associative Memory (BAM) and also gave the concept of Fuzzy Logic in ANN.

- The historical review shows that significant progress has been made in this field.
- Neural network based chips are emerging and applications to complex problems are being developed.
- Surely, today is a period of transition for neural network technology.

Biological Neuron

- A nerve cell (neuron) is a special biological cell that processes information.
- According to an estimation, there are huge number of neurons, approximately 10^{11} with numerous interconnections, approximately 10^{15}

Schematic Diagram



Working of a Biological Neuron

As shown in the above diagram, a typical neuron consists of the following four parts with the help of which we can explain its working −

- Dendrites − They are tree-like branches, responsible for receiving the information from other neurons it is connected to. In other sense, we can say that they are like the ears of neuron.
- Soma − It is the cell body of the neuron and is responsible for processing of information, they have received from dendrites.
- Axon − It is just like a cable through which neurons send the information.

- Synapses − It is the connection between the axon and other neuron dendrites.

ANN versus BNN

- Before taking a look at the differences between Artificial Neural Network (ANN) and Biological Neural Network (BNN), let us take a look at the similarities based on the terminology between these two.

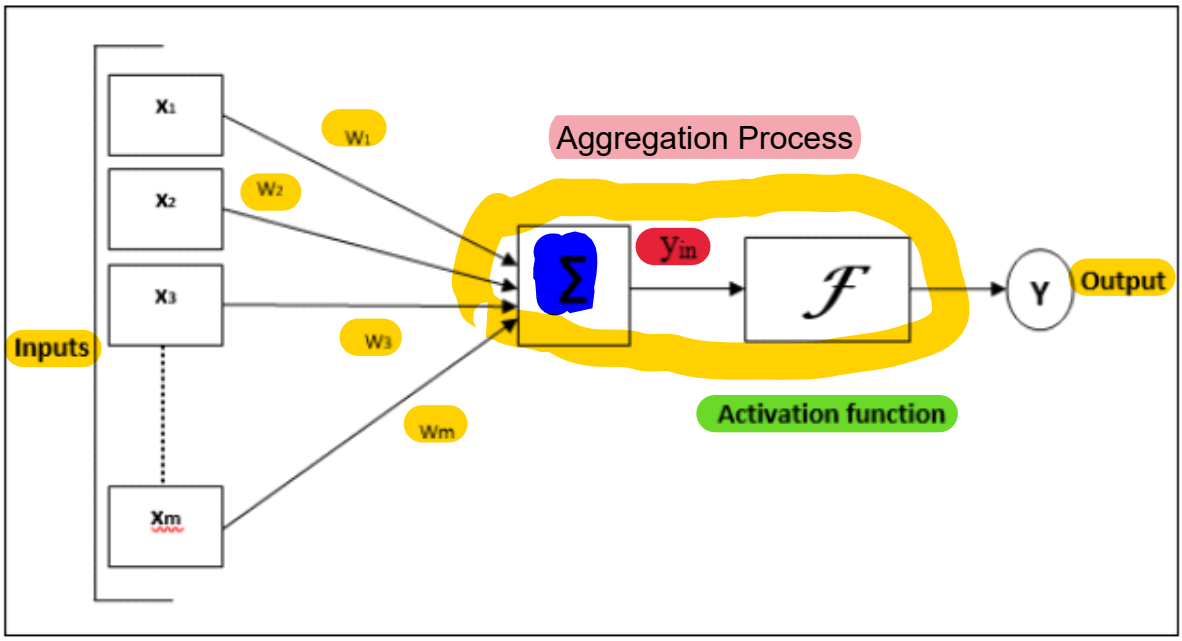
Biological Neural Network (BNN)	Artificial Neural Network (ANN)
Soma	Node
Dendrites	Input
Synapse	Weights or Interconnections
Axon	Output

The following table shows the comparison between ANN and BNN based on some criteria mentioned.

Criteria	BNN	ANN
Processing	Massively parallel, slow but superior than ANN	Massively parallel, fast but inferior than BNN
Size	10^{11} neurons and 10^{15}interconnections	10^2 to 10^4 nodes (mainly depends on the type of application and network designer)
Learning	They can tolerate ambiguity	Very precise, structured and formatted data is required to tolerate ambiguity
Fault tolerance	Performance degrades with even partial damage	It is capable of robust performance, hence has the potential to be fault tolerant
Storage capacity	Stores the information in the synapse	Stores the information in continuous memory locations

Model of Artificial Neural Network

The following diagram represents the general model of ANN followed by its processing.



- An artificial neuron is a mathematical function conceived as a model of biological neurons, a neural network.
- Artificial neurons are elementary units in an artificial neural network.
- The artificial neuron receives one or more inputs and sums them to produce an output.
- Neuron. Neurons are the basic unit of a neural network.
- In nature, neurons have a number of dendrites (inputs), a cell nucleus (processor) and an axon (output). ... neuron j:
- As you can see they have several inputs, for each input there's a weight (the weight of that specific connection)

For the above general model of artificial neural network, the net input can be calculated as follows −

$$y_{in}\:=\:x_{1}.w_{1}\:+\:x_{2}.w_{2}\:+\:x_{3}.w_{3}\:\dotso\: x_{m}.w_{m}$$

i.e., Net input $y_{in}\:=\:\sum_i^m\:x_{i}.w_{i}$

The output can be calculated by applying the activation function over the net input.

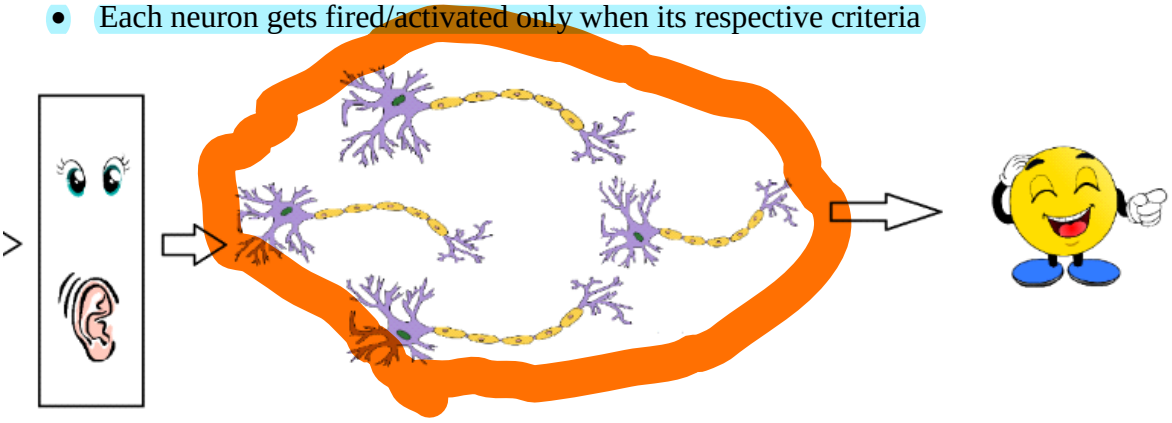
$$Y\:=\:F(y_{in}) $$

Output = function (net input calculated)

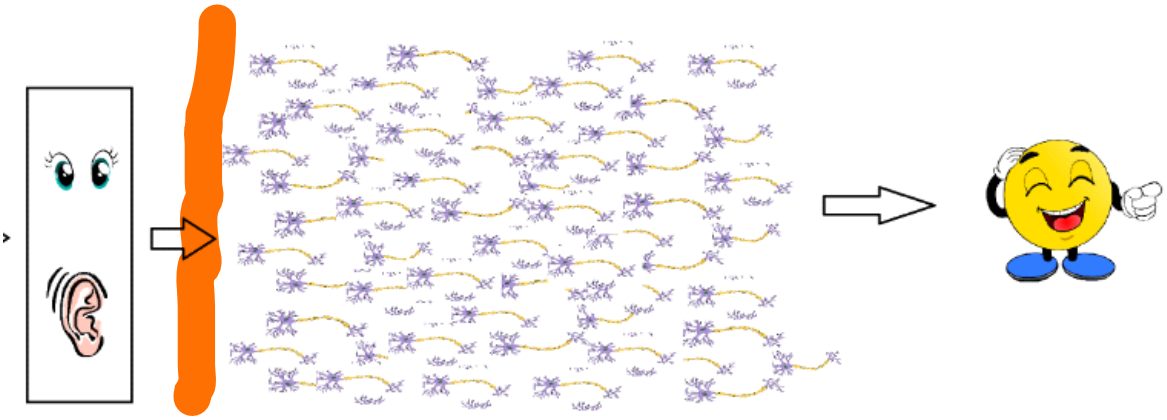
What is McCulloch Pitts neuron model?

Dendrite: Receives signals from other neurons
Soma: Processes the information
Axon: Transmits the output of this neuron
Synapse: Point of connection to other neurons.

- Basically, a neuron takes an input signal (dendrite), processes it like the CPU (soma), passes the output through a cable like structure to other connected neurons (axon to synapse to other neuron's dendrite).
- Our sense organs interact with the outer world and send the visual and sound information to the neurons.
- Example say you are watching Friends. Now the information your brain receives is taken in by the "laugh or not" set of neurons that will help you make a decision on whether to laugh or not.
- Each neuron gets fired/activated only when its respective criteria



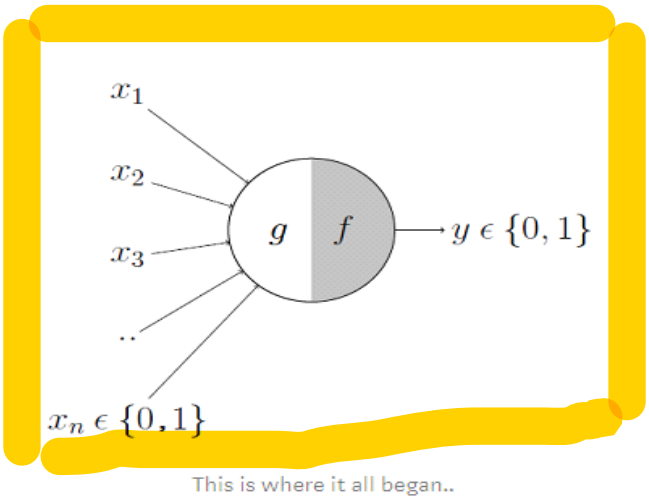
- In reality, it is not just a couple of neurons which would do the decision making.
- There is a massively parallel interconnected network of 10^{11} neurons (100 billion) in our brain and their connections are not as simple as I showed you above.
- It might look something like this:



- Now the sense organs pass the information to the first/lowest layer of neurons to process it.
- And the output of the processes is passed on to the next layers in a hierarchical manner, some of the neurons will fire and some won't and this process goes on until it results in a final response.

- The early model of an artificial neuron is introduced by Warren McCulloch and Walter Pitts in 1943.
- The McCulloch-Pitts neural model is also known as linear threshold gate. It is a neuron of a set of inputs $I_1, I_2, I_3, \ldots, I_m$ and one output y.
- The linear threshold gate simply classifies the set of inputs into two different classes. Thus the output y is binary. Such a function can be described mathematically using these equations:

The first computational model of a neuron was proposed by Warren MuCulloch (neuroscientist) and Walter Pitts (logician) in 1943.



This is where it all began..

It may be divided into 2 parts. The first part, g takes an input (ahem dendrite ahem), performs an aggregation and based on the aggregated value the second part, f makes a decision.

Linear separability

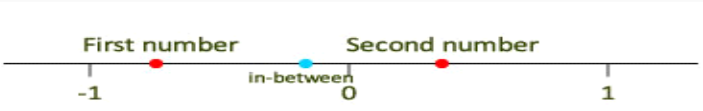
Linear separability is an important concept in neural networks.

The idea is to check if you can separate points in an n-dimensional space using only n-1 dimensions.

One Dimension

Consider you are on a number line. You take any two numbers. Now, there are two possibilities:

- You choose two different numbers
- You choose the same number
- If you choose two different numbers, you can always find another number between them. This number "separates" the two numbers you chose.



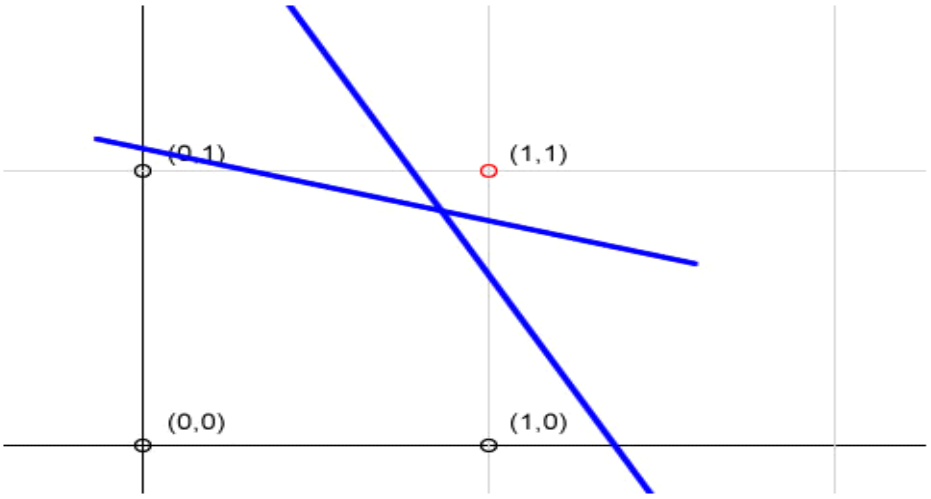
One dimensional separability

- So, you say that these two numbers are "linearly separable".
- But, if both numbers are the same, you simply cannot separate them. They're the same.

- So, they're "linearly inseparable". (Not just linearly, they're aren't separable at all. You cannot separate something from itself).

Two Dimensions

On extending this idea to two dimensions, some more possibilities come into existence. Consider the following:



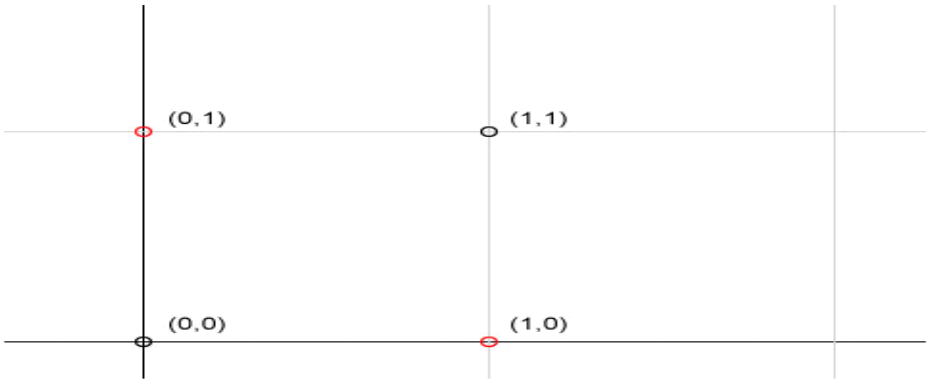
Two classes of points

Here, we're like to seperate the point (1,1) from the other points.

You can see that there exists a line that does this. In fact, there exist infinite such lines.

So, these two "classes" of points are linearly separable.

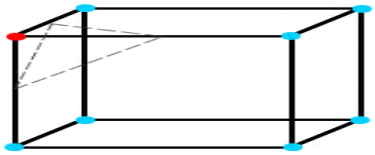
The first class consists of the point (1,1) and the other class has (0,1), (1,0) and (0,0).



Linearly inseparable

Three dimensions

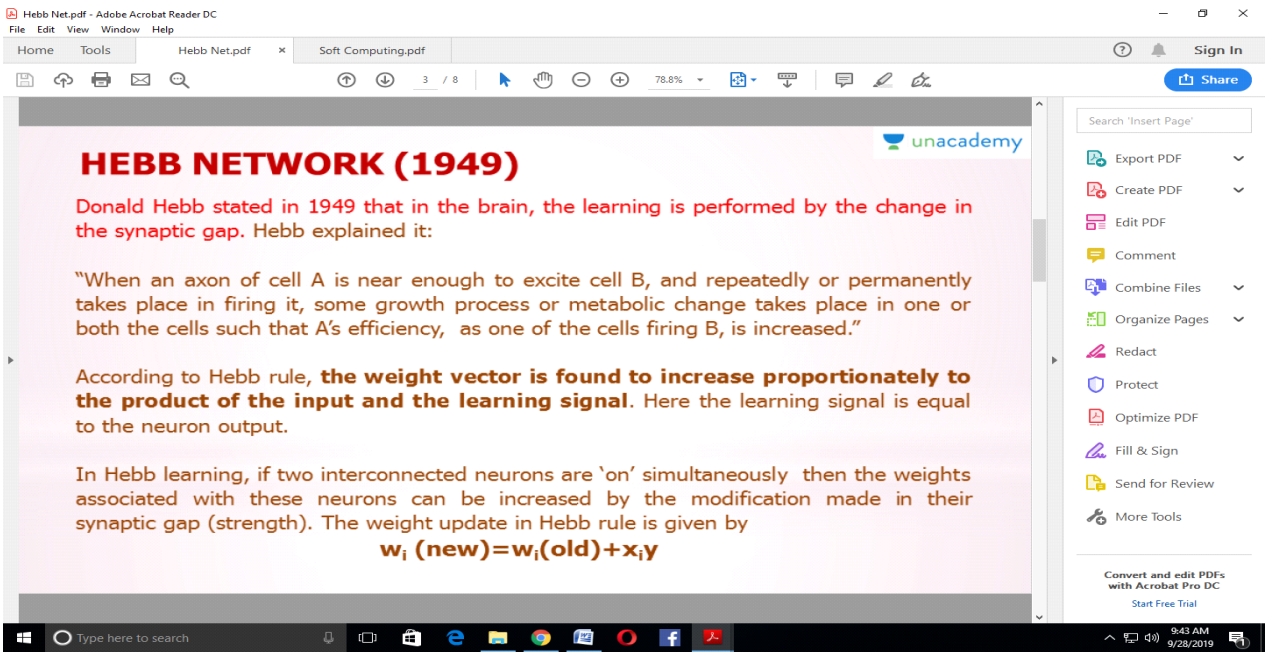
Extending the above example to three dimensions. You need a plane for separating the two classes.

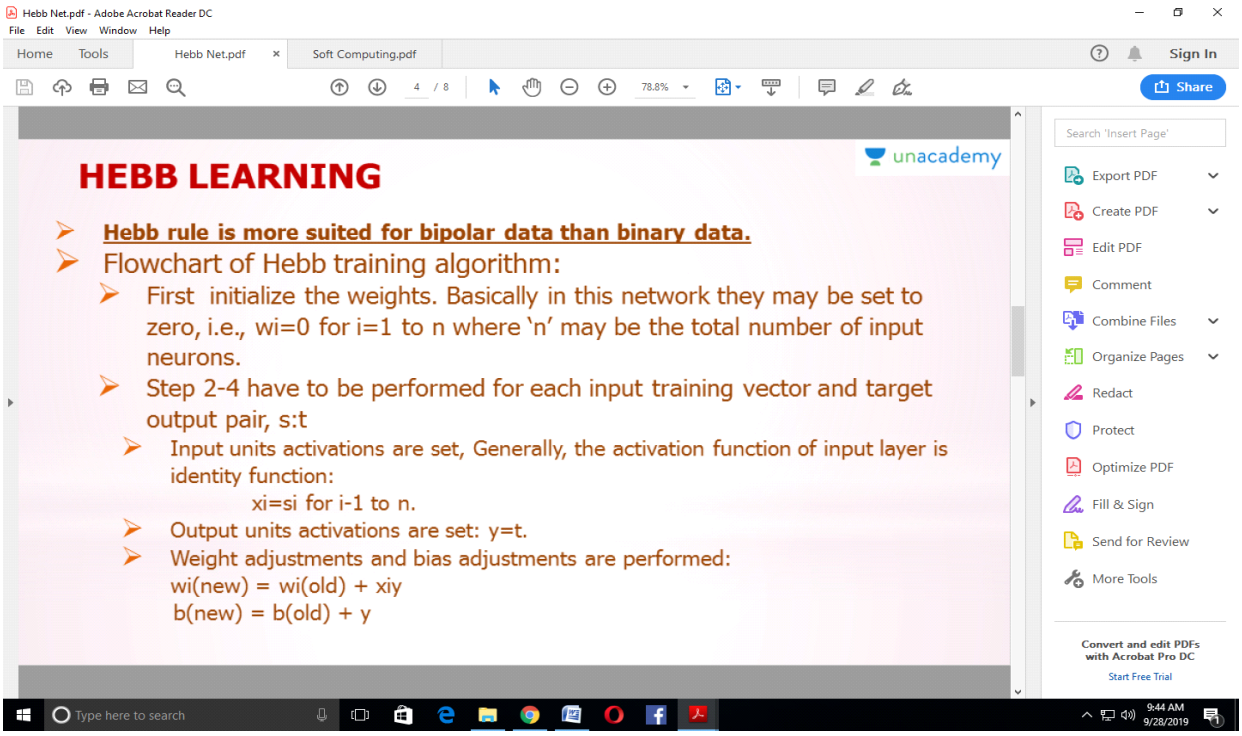


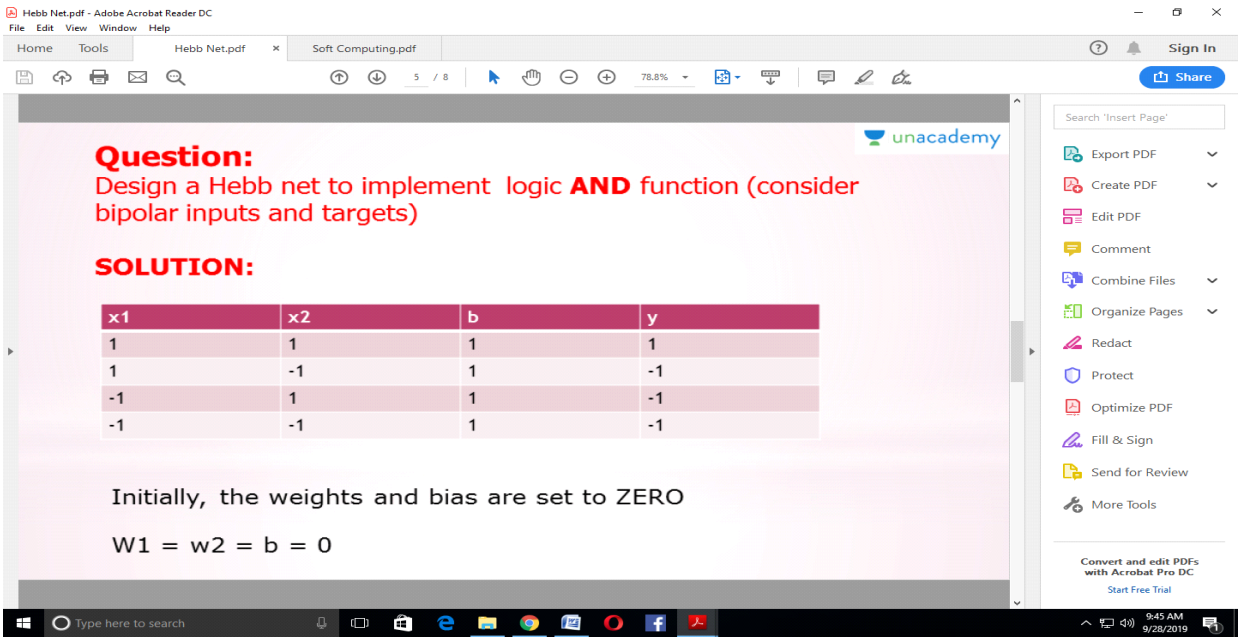
Linear separability in 3D space

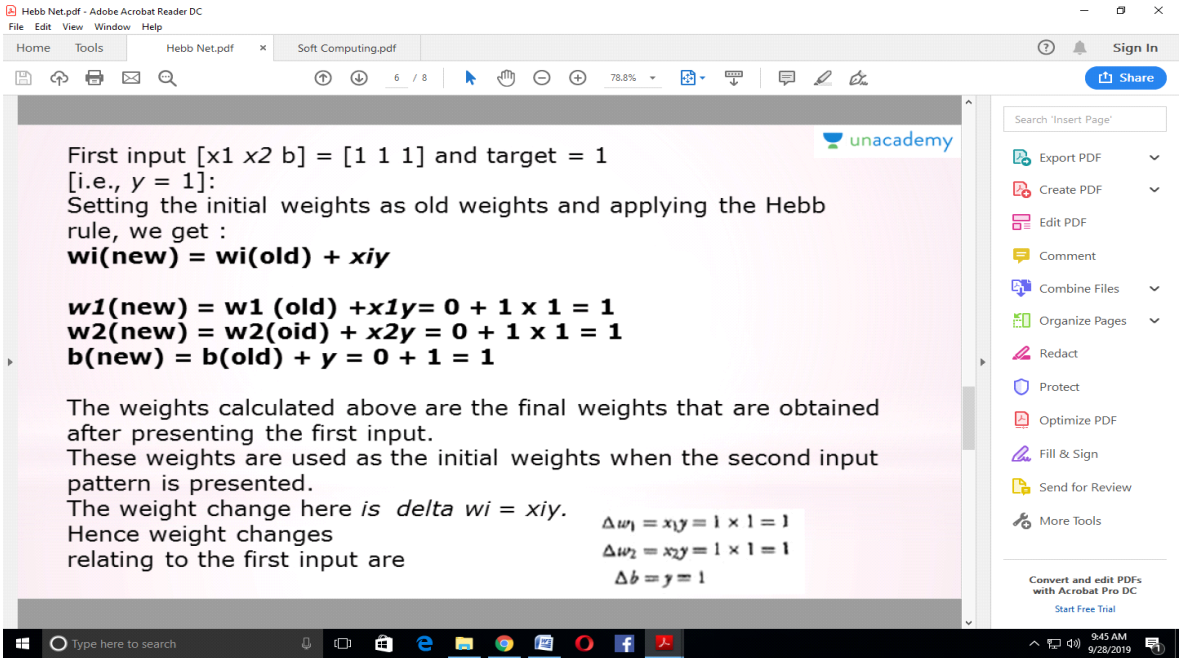
Hebb Network.

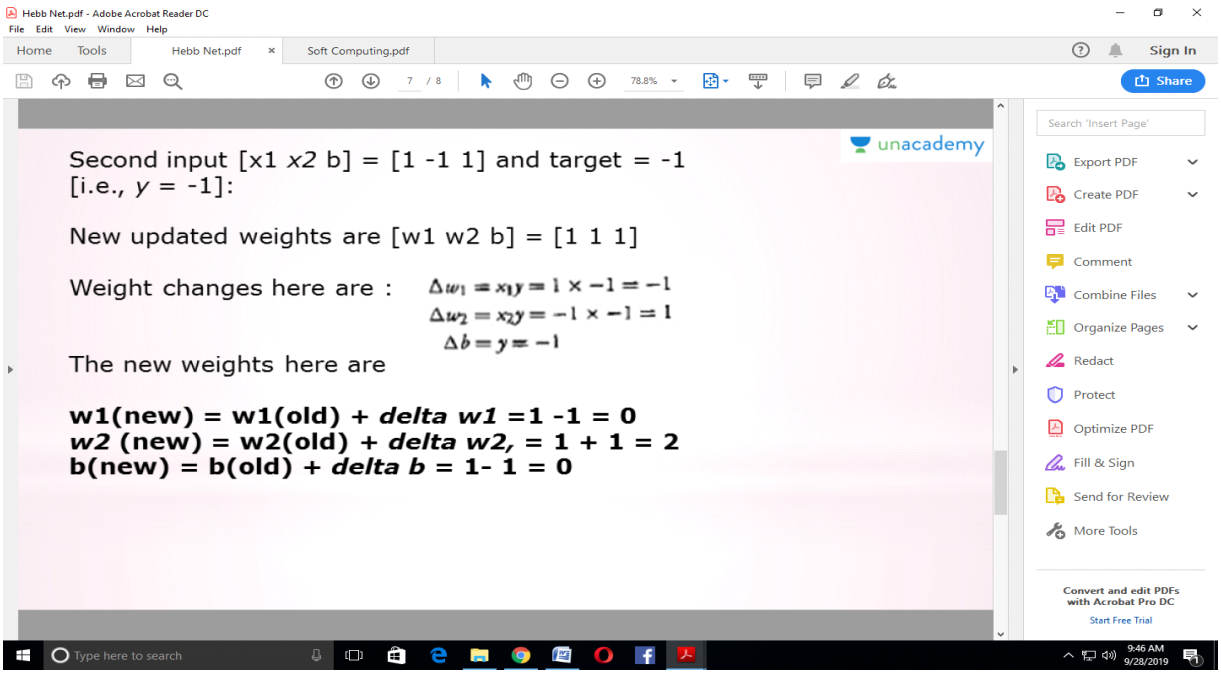
- Neural networks in which weights can store a set of associations of P pattern are called associative memory neural networks.
- A pair of vectors , with is an association. Every vector is n-tuples (n components), and every vector is m-tuples (m components).
- By using Hebb rule the weights can be calculated.
- What is Weight?
- The idea of weight is a foundational concept in artificial neural networks. A set of weighted inputs allows each artificial neuron or node in the system to produce related outputs.
- Weight is also known as synaptic weight.
- In an artificial neuron, a collection of weighted inputs is the vehicle through which the neuron engages in an activation function and produces a decision (either firing or not firing).
- Typical artificial neural networks have various layers including an input layer, hidden layers and an output layer.
- At each layer, the individual neuron is taking in these inputs and weighting them accordingly.
- This simulates the biological activity of individual neurons, sending signals with a given synaptic weight from the axon of a neuron to the dendrites of another neuron.

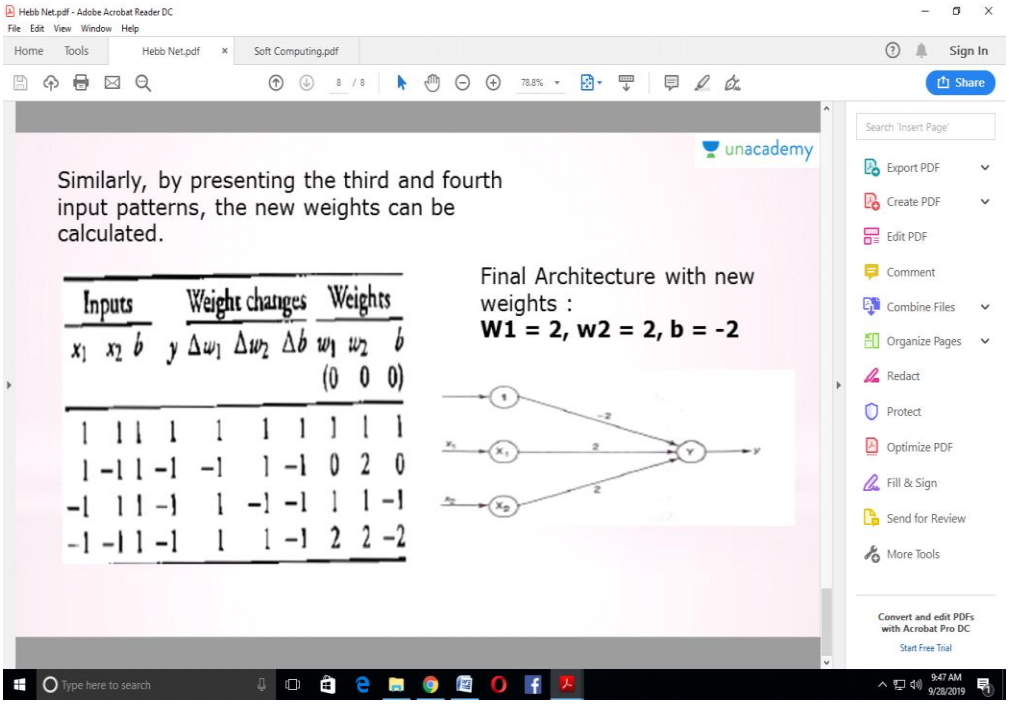

Hebb Net.pdf - Adobe Acrobat Reader DC
File Edit View Window Help
Home Tools Hebb Net.pdf Soft Computing.pdf
Sign In
Share
unacademy
HEBB NETWORK (1949)
Donald Hebb stated in 1949 that in the brain, the learning is performed by the change in the synaptic gap. Hebb explained it:
"When an axon of cell A is near enough to excite cell B, and repeatedly or permanently takes place in firing it, some growth process or metabolic change takes place in one or both the cells such that A's efficiency, as one of the cells firing B, is increased."
According to Hebb rule, the weight vector is found to increase proportionately to the product of the input and the learning signal. Here the learning signal is equal to the neuron output.
In Hebb learning, if two interconnected neurons are 'on' simultaneously then the weights associated with these neurons can be increased by the modification made in their synaptic gap (strength). The weight update in Hebb rule is given by
$w_i(new) = w_i(old) + x_i y$
Export PDF
Create PDF
Edit PDF
Comment
Combine Files
Organize Pages
Redact
Protect
Optimize PDF
Fill & Sign
Send for Review
More Tools
Convert and edit PDFs with Acrobat Pro DC
Start Free Trial
Type here to search
9:43 AM 9/28/2019


Hebb Net.pdf - Adobe Acrobat Reader DC
File Edit View Window Help
Home Tools Hebb Net.pdf Soft Computing.pdf
Sign In
Share
unacademy
HEBB LEARNING
Hebb rule is more suited for bipolar data than binary data.
Flowchart of Hebb training algorithm:
First initialize the weights. Basically in this network they may be set to zero, i.e., wi=0 for i=1 to n where 'n' may be the total number of input neurons.
Step 2-4 have to be performed for each input training vector and target output pair, s:t
Input units activations are set, Generally, the activation function of input layer is identity function:
xi=si for i-1 to n.
Output units activations are set: y=t.
Weight adjustments and bias adjustments are performed:
wi(new) = wi(old) + xiy
b(new) = b(old) + y
Export PDF
Create PDF
Edit PDF
Comment
Combine Files
Organize Pages
Redact
Protect
Optimize PDF
Fill & Sign
Send for Review
More Tools
Convert and edit PDFs with Acrobat Pro DC
Start Free Trial
Type here to search
9:44 AM 9/28/2019


Hebb Net.pdf - Adobe Acrobat Reader DC
File Edit View Window Help
Home Tools Hebb Net.pdf Soft Computing.pdf
Sign In
Share
unacademy
Question:
Design a Hebb net to implement logic AND function (consider bipolar inputs and targets)
SOLUTION:
x1 x2 b y
1 1 1 1
1 -1 1 -1
-1 1 1 -1
-1 -1 1 -1
Initially, the weights and bias are set to ZERO
W1 = w2 = b = 0
Export PDF
Create PDF
Edit PDF
Comment
Combine Files
Organize Pages
Redact
Protect
Optimize PDF
Fill & Sign
Send for Review
More Tools
Convert and edit PDFs with Acrobat Pro DC
Start Free Trial
Type here to search
9:45 AM 9/28/2019


Hebb Net.pdf - Adobe Acrobat Reader DC
File Edit View Window Help
Home Tools Hebb Net.pdf Soft Computing.pdf
Sign In
Share
unacademy
First input [x1 x2 b] = [1 1 1] and target = 1
[i.e., y = 1]:
Setting the initial weights as old weights and applying the Hebb rule, we get :
wi(new) = wi(old) + xiy
w1(new) = w1 (old) +x1y= 0 + 1 x 1 = 1
w2(new) = w2(oid) + x2y = 0 + 1 x 1 = 1
b(new) = b(old) + y = 0 + 1 = 1
The weights calculated above are the final weights that are obtained after presenting the first input.
These weights are used as the initial weights when the second input pattern is presented.
The weight change here is delta wi = xiy.
Hence weight changes relating to the first input are
Δw1 = x1y = 1 × 1 = 1
Δw2 = x2y = 1 × 1 = 1
Δb = y = 1
Export PDF
Create PDF
Edit PDF
Comment
Combine Files
Organize Pages
Redact
Protect
Optimize PDF
Fill & Sign
Send for Review
More Tools
Convert and edit PDFs with Acrobat Pro DC
Start Free Trial
Type here to search
9:45 AM 9/28/2019


Hebb Net.pdf - Adobe Acrobat Reader DC
Second input [x1 x2 b] = [1 -1 1] and target = -1
[i.e., y = -1]:
New updated weights are [w1 w2 b] = [1 1 1]
Weight changes here are :
$\Delta w_1 = x_1 y = 1 \times -1 = -1$
$\Delta w_2 = x_2 y = -1 \times -1 = 1$
$\Delta b = y = -1$
The new weights here are
w1(new) = w1(old) + delta w1 =1 -1 = 0
w2 (new) = w2(old) + delta w2, = 1 + 1 = 2
b(new) = b(old) + delta b = 1- 1 = 0
unacademy


Hebb Net.pdf - Adobe Acrobat Reader DC
Similarly, by presenting the third and fourth input patterns, the new weights can be calculated.
Inputs | Weight changes | Weights
x1 x2 b | y | Δw1 Δw2 Δb | w1 w2 b
(0 0 0)
1 1 1 1 1 1 1 1 1
1 -1 1 -1 -1 1 -1 0 2 0
-1 1 1 -1 1 -1 -1 1 1 -1
-1 -1 1 -1 1 1 -1 2 2 -2
Final Architecture with new weights :
W1 = 2, w2 = 2, b = -2
unacademy

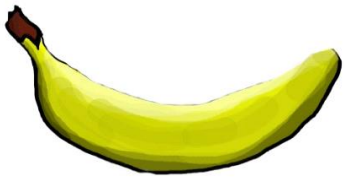
Y = x1 and x2

Supervised Learning Network

Introduction

- Supervised learning as the name indicates the presence of a supervisor as a teacher.
- Basically supervised learning is a learning in which we teach or train the machine using data which is well labeled that means some data is already tagged with the correct answer.
- After that, the machine is provided with a new set of examples(data) so that supervised learning algorithm analyses the training data(set of training examples) and produces a correct outcome from labeled data.
- For instance, suppose you are given an basket filled with different kinds of fruits. Now the first step is to train the machine with all different fruits one by one like this:

- If shape of object is rounded and depression at top having color Red then it will be labelled as –Apple.
- If shape of object is long curving cylinder having color Green-Yellow then it will be labelled as –Banana.
- Now suppose after training the data, you have given a new separate fruit say Banana from basket and asked to identify it.



- Since the machine has already learned the things from previous data and this time have to use it wisely.
- It will first classify the fruit with its shape and color and would confirm the fruit name as BANANA and put it in Banana category.
- Thus the machine learns the things from training data(basket containing fruits) and then apply the knowledge to test data(new fruit).

Supervised learning classified into two categories of algorithms:

- Classification: A classification problem is when the output variable is a category, such as "Red" or "blue" or "disease" and "no disease".
- Regression: A regression problem is when the output variable is a real value, such as "dollars" or "weight".

Unsupervised learning

- Unsupervised learning is the training of machine using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance.
- Here the task of machine is to group unsorted information according to similarities, patterns and differences without any prior training of data.
- Unlike supervised learning, no teacher is provided that means no training will be given to the machine.
- Therefore machine is restricted to find the hidden structure in unlabeled data by our-self.
- For instance, suppose it is given an image having both dogs and cats which have not seen ever.

- Thus the machine has no idea about the features of dogs and cat so we can't categorize it in dogs and cats.
- But it can categorize them according to their similarities, patterns, and differences i.e., we can easily categorize the above picture into two parts.
- First first may contain all pics having dogs in it and second part may contain all pics having cats in it.
- Here you didn't learn anything before, means no training data or examples.

Unsupervised learning classified into two categories of algorithms:

- Clustering: A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.
- Association: An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

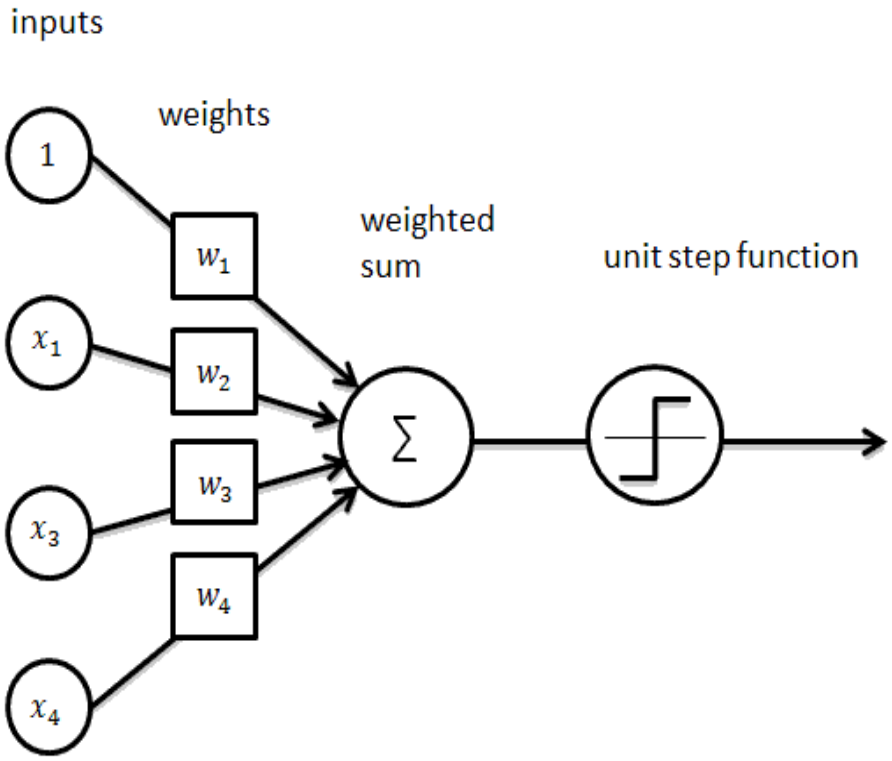
Difference b/w Supervised and Unsupervised Learning :

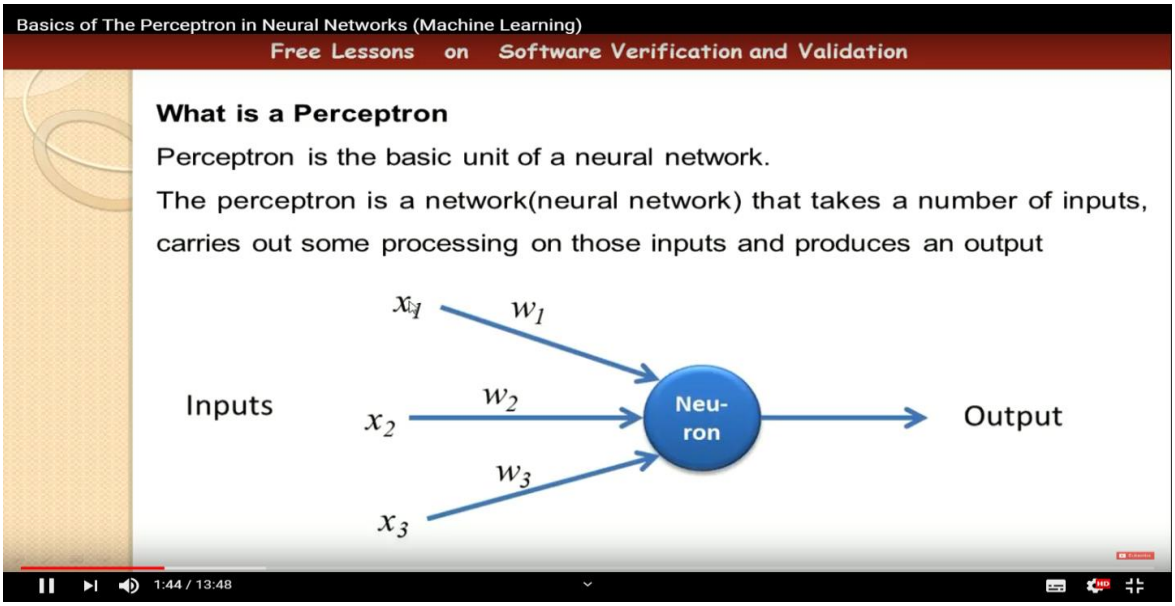
	SUPERVISED LEARNING	UNSUPERVISED LEARNING
Input Data	Uses Known and Labeled Data as input	Uses Unknown Data as input
Computational Complexity	Very Complex	Less Computational Complexity
Real Time	Uses off-line analysis	Uses Real Time Analysis of Data
Number of Classes	Number of Classes are known	Number of Classes are not known
Accuracy of Results	Accurate and Reliable Results	Moderate Accurate and Reliable Results

Perceptron Network

- Perception not only creates our experience of the world around us; it allows us to act within our environment.
- Perception is very important in understanding human behavior because every person perceives the world and approaches life problems differently
- Perceptron is a single layer neural network and a multi-layer perceptron is called Neural Networks.
- Perceptron is a linear classifier (binary). Also, it is used in supervised learning. It helps to classify the given input data.
- In machine learning, the perceptron is an algorithm for supervised learning of binary classifiers.
- A binary classifier is a function which can decide whether or not an input, represented by a vector of numbers, belongs to some specific class.

- It is a type of linear classifier, i.e. a classification algorithm that makes its predictions based on a linear predictor function combining a set of weights with the feature vector.
- Binary or binomial classification is the task of classifying the elements of a given set into two groups on the basis of a classification rule.





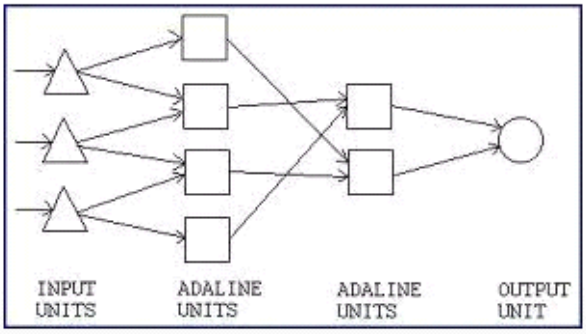
Adaptive Linear Neuron(ADALINE)

- Adaline and madaline comes under the supervised learning networks
- ADALINE Known as Adaptive Linear Neuron
- Adaline is a network with a single linear unit
- The Adaline network is trained using the delta rule

Architecture

As already stated Adaline is a single-unit neuron, which receives input from several units and also from one unit, called bias.

An Adeline model consists of trainable weights. The inputs are of two values (+1 or -1) and the weights have signs (positive or negative).



Adaline Madaline neural network

Initially random weights are assigned.

The net input calculated is applied to a quantizer transfer function (possibly activation function) that restores the output to +1 or -1.

The Adaline model compares the actual output with the target output and with the bias and the adjusts all the weights.

Training Algorithm

The Adaline network training algorithm is as follows:

Step0: weights and bias are to be set to some random values but not zero. Set the learning rate parameter α.

Step1: perform steps 2-6 when stopping condition is false.

Step2: perform steps 3-5 for each bipolar training pair s:t

Step3: set activations foe input units i=1 to n.

Step4: calculate the net input to the output unit.

Step5: update the weight and bias for i=1 to n

Step6: if the highest weight change that occurred during training is smaller than a specified tolerance then stop the training process, else continue. This is the test for the stopping condition of a network.

- ADALINE (Adaptive Linear Neuron or later Adaptive Linear Element) is an early single-layer artificial neural network and the name of the physical device that implemented this network.

- The network uses memistors.
- It is based on the McCulloch–Pitts neuron.

- The difference between Adaline and the standard (McCulloch–Pitts) perceptron is that in the learning phase, the weights are adjusted according to the weighted sum of the inputs (the net).

- In the standard perceptron, the net is passed to the activation (transfer) function and the function's output is used for adjusting the weights.

Adaline is a single layer neural network with multiple nodes where each node accepts multiple inputs and generates one output. Given the following variables as:

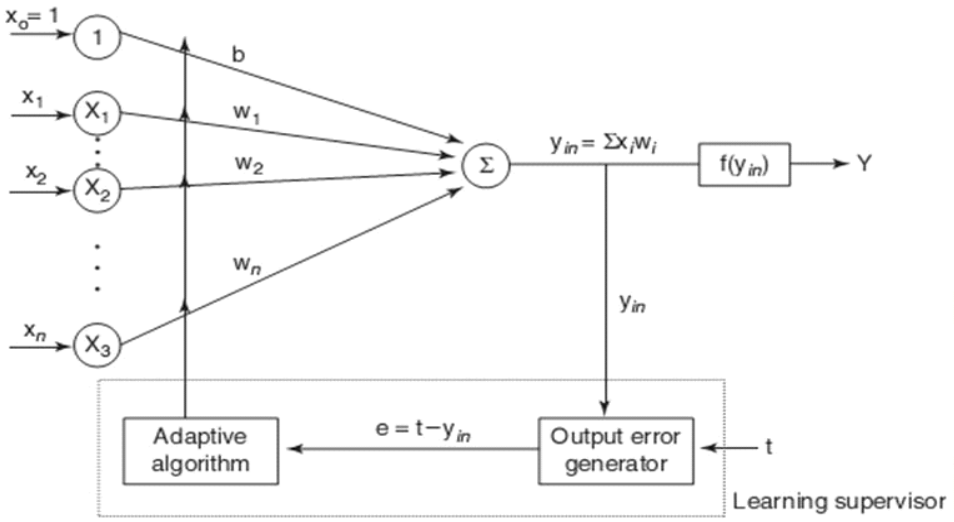
- x is the input vector
- w is the weight vector
- n is the number of inputs
- θ some constant
- y is the output of the model

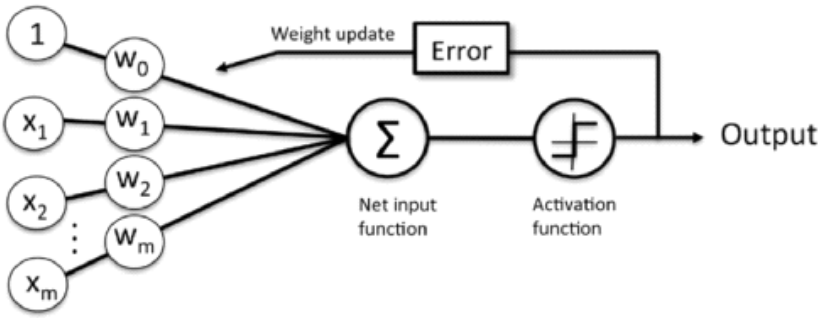
then we find that the output is $y = \sum_{j=1}^{n} x_j w_j + \theta$. If we further assume that

- $x_0 = 1$
- $w_0 = \theta$

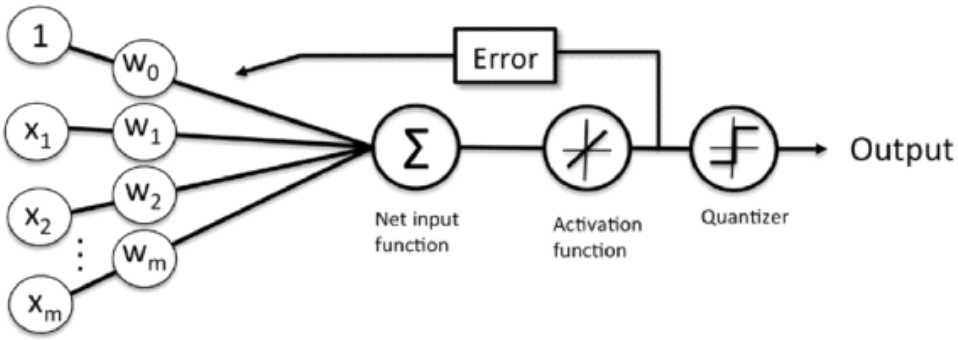
then the output further reduces to: $y = \sum_{j=0}^{n} x_j w_j$

ADALINE MODEL





Perceptron



Adaptive linear neuron

Madaline(multiple adaptive linear neuron)

- Stands for multiple adaptive linear neuron
- It consists of many adalines in parallel with a single output unit whose value is based on certain selection rules.
- It uses the majority vote rule
- On using this rule, the output unit would have an answer either true or false.
- On the other hand, if AND rule is used, the output is true if and only if both the inputs are true and so on.
- The training process of madaline is similar to that of adaline

Architecture

- It consists of "n" units of input layer and "m" units of adaline layer and "1"
- Unit of the Madaline layer
- Each neuron in the adaline and madaline layers has a bias of excitation "1"

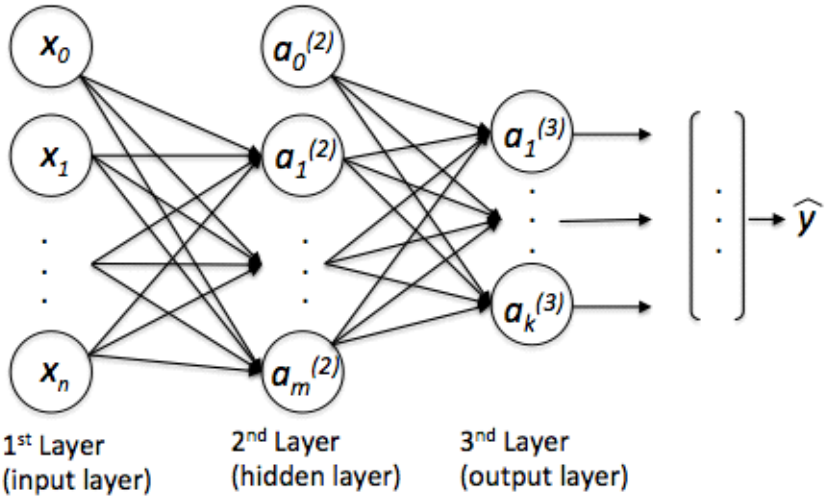
- The Adaline layer is present between the input layer and the madaline layer; the adaline layer is considered as the hidden layer.

Uses

- The use of hidden layer gives the net computational capability which is not found in the single-layer nets, but this complicates the training process to some extent.

Training Algorithm:

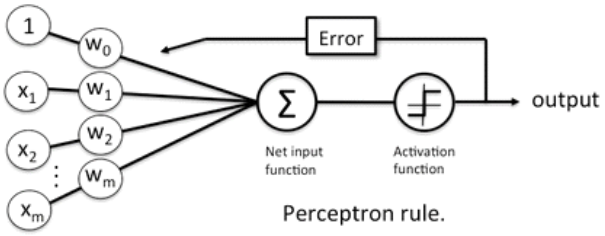
In this training algorithm, only the weights between the hidden layers are adjusted, and the weights for the output units are fixed. The weights v1, v2……vm and the bias b0 that enter into output unit Y are determined so that the response of unit Y is 1.



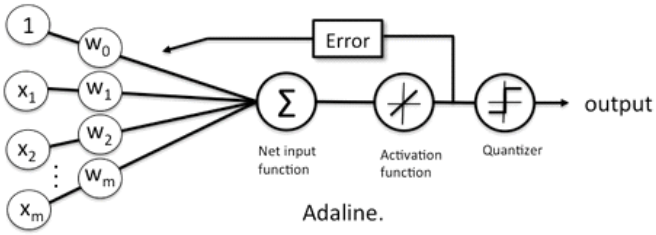
The differences between the Perceptron and Adaline

- the Perceptron uses the class labels to learn model coefficients
- Adaline uses continuous predicted values (from the net input) to learn the model coefficients, which is more "powerful" since it tells us by "how much" we were right or wrong

So, in the perceptron, as illustrated below, we simply use the predicted class labels to update the weights, and in Adaline, we use a continuous response:



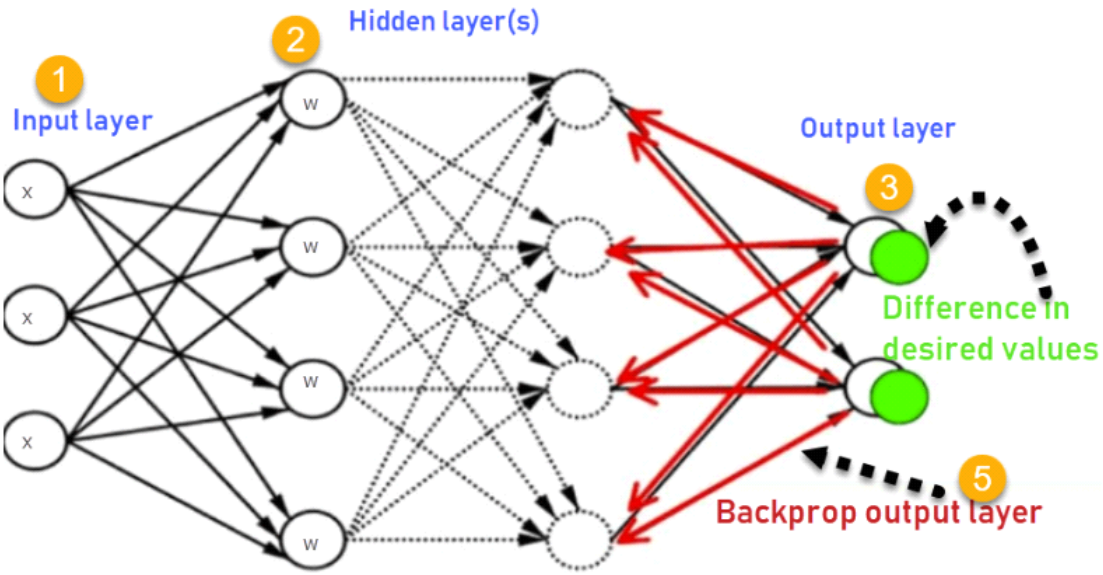
Perceptron rule.



Adaline.

Back-propagation Network

- Back-propagation is just a way of propagating the total loss back into the neural network to know how much of the loss every node is responsible for, and subsequently updating the weights in such a way that minimizes the loss by giving the nodes with higher error rates lower weights and vice versa.
- Back-propagation is the essence of neural net training. It is the method of fine-tuning the weights of a neural net based on the error rate obtained in the previous epoch (i.e., iteration).
- Proper tuning of the weights allows you to reduce error rates and to make the model reliable by increasing its generalization.
- Backpropagation is a short form for "backward propagation of errors." It is a standard method of training artificial neural networks.
- This method helps to calculate the gradient of a loss function with respects to all the weights in the network.



How Backpropagation Works

- nputs X, arrive through the preconnected path
- Input is modeled using real weights W. The weights are usually randomly selected.
- Calculate the output for every neuron from the input layer, to the hidden layers, to the output layer.
- Calculate the error in the outputs

$$Error_B = \text{Actual Output} - \text{Desired Output}$$

- Travel back from the output layer to the hidden layer to adjust the weights such that the error is decreased.
- Keep repeating the process until the desired output is achieved

Advantages of Backpropagation are:

- Backpropagation is fast, simple and easy to program
- It has no parameters to tune apart from the numbers of input
- It is a flexible method as it does not require prior knowledge about the network
- It is a standard method that generally works well
- It does not need any special mention of the features of the function to be learned.

Types of Backpropagation Networks

Two Types of Backpropagation Networks are:

- Static Back-propagation

- Recurrent Backpropagation

Static back-propagation:

- It is one kind of backpropagation network which produces a mapping of a static input for static output. It is useful to solve static classification issues like optical character recognition.

Recurrent Backpropagation:

- Recurrent backpropagation is fed forward until a fixed value is achieved. After that, the error is computed and propagated backward.
- The main difference between both of these methods is: that the mapping is rapid in static back-propagation while it is nonstatic in recurrent backpropagation.

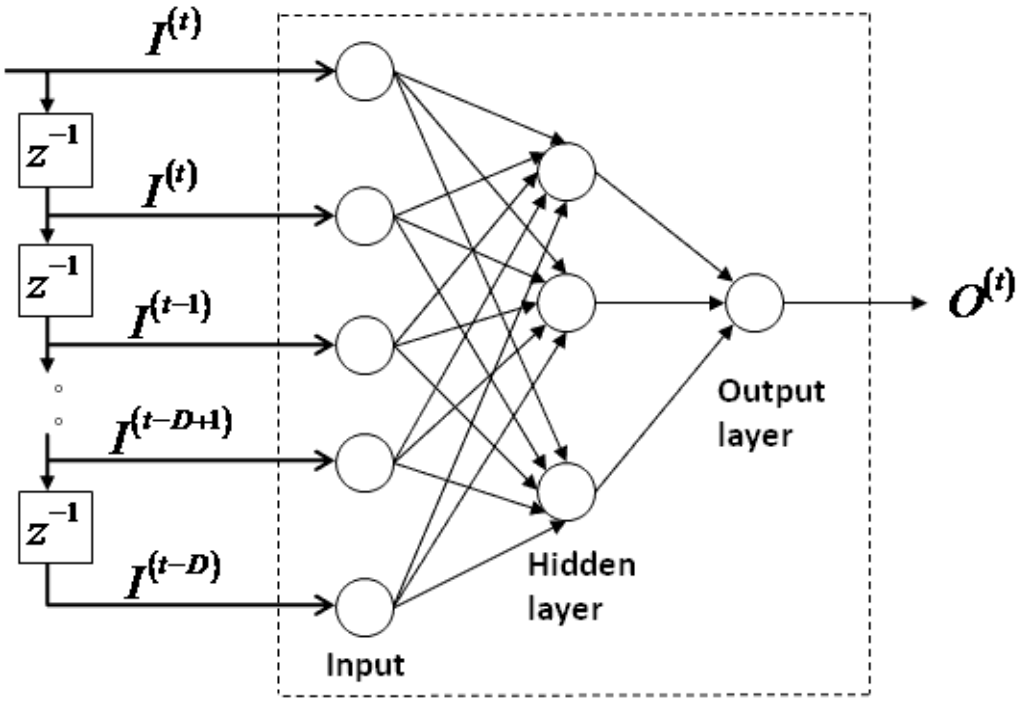
Time delay neural network (TDNN)

- Time delay neural network (TDNN) is a multilayer artificial neural network architecture whose purpose is to
  - classify patterns with shift-invariance, and
  - model context at each layer of the network.
- Shift-invariant classification means that the classifier does not require explicit segmentation prior to classification.
- For the classification of a temporal pattern (such as speech), the TDNN thus avoids having to determine the beginning and end points of sounds before classifying them.
- For contextual modelling in a TDNN, each neural unit at each layer receives input not only from activations/features at the layer below, but from a pattern of unit output and its context.
- For time signals each unit receives as input the activation patterns over time from units below.
- Applied to two-dimensional classification (images, time-frequency patterns), the TDNN can be trained with shift-invariance in the coordinate space and avoids precise segmentation in the coordinate space.
- The Time Delay Neural Network, like other neural networks, operates with multiple interconnected layers of perceptrons, and is implemented as a feedforward neural network.

All neurons (at each layer) of a TDNN receive inputs from the outputs of neurons at the layer below but with two differences:

- Unlike regular Multi-Layer_HYPERLINK "https://en.wikipedia.org/wiki/Multilayer_perceptron"perceptrons, all units in a TDNN, at each layer, obtain inputs from a contextual *window* of outputs from the layer below. For time varying signals (e.g. speech), each unit has connections to the output from units below but also to the time-delayed (past) outputs from these same units.

- Shift-invariance is achieved by explicitly removing position dependence during backpropagation training. This is done by making time-shifted copies of a network across the dimension of invariance (here: time). The error gradient is then computed by backpropagation through all these networks from an overall target vector, but before performing the weight update, the error gradients associated with shifted copies are averaged and thus shared and constraint to be equal.



Tree Neural Network(Decision Tree)

Definition

- Decision Tree algorithm belongs to the family of supervised learning algorithms. Unlike other supervised learning algorithms, decision tree algorithm can be used for solving regression and classification problems too.
- The general motive of using Decision Tree is to create a training model which can use to predict class or value of target variables by learning decision rules inferred from prior data(training data).

Decision Tree Algorithm

- Place the best attribute of the dataset at the root of the tree.
- Split the training set into subsets. Subsets should be made in such a way that each subset contains data with the same value for an attribute.

- Repeat step 1 and step 2 on each subset until you find leaf nodes in all the branches of the tree.

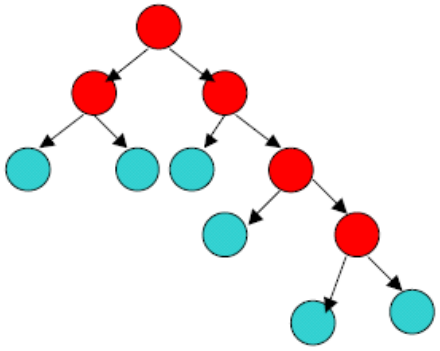
Assumptions while creating Decision Tree

The below are the some of the assumptions we make while using Decision tree:

- At the beginning, the whole training set is considered as the root.
- Feature values are preferred to be categorical. If the values are continuous then they are discretized prior to building the model.
- Records are distributed recursively on the basis of attribute values.
- Order to placing attributes as root or internal node of the tree is done by using some statistical approach.

What is a decision tree?

- Decision tree (DT) is a tree structure.
- In a DT, there are two kinds of nodes
  - Internal nodes: Used to make local decisions based on the local information they possess.
  - Terminal nodes: Used to make the final decision.



Local decision making in a DT

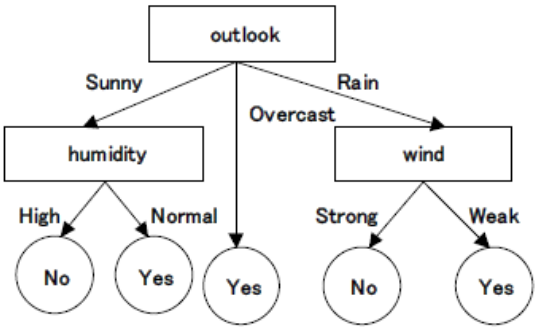
- In each internal node, a decision function $f(x)$ is used to make local decisions.
- For binary decision trees, an input pattern is classified to the left child node if $f(\boldsymbol{x})<0$; and to the right child node otherwise.
- The decision function in a standard DT is $f(\boldsymbol{x})= x_i-a_i$.
- That is, only one of the features is used for making the decisions.

Final decision making in a DT

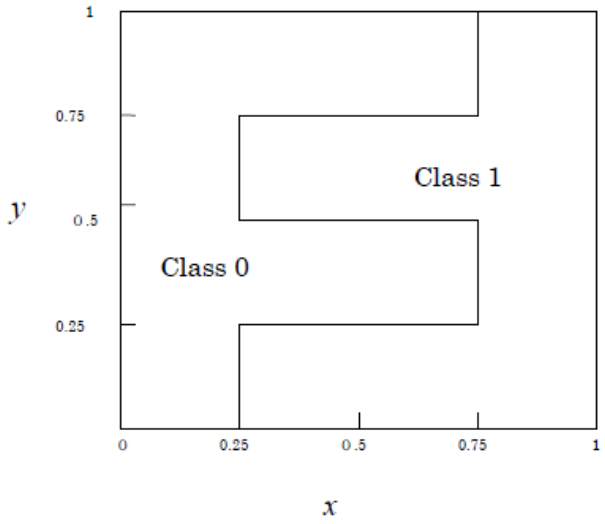
- In each terminal node, the distribution of data assigned to the node by the tree is used for making the final decisions.
- If the majority of data belong to the ith class, the terminal node is assigned with the class label i.
- All test data assigned to this node by the tree are assigned with the label i.

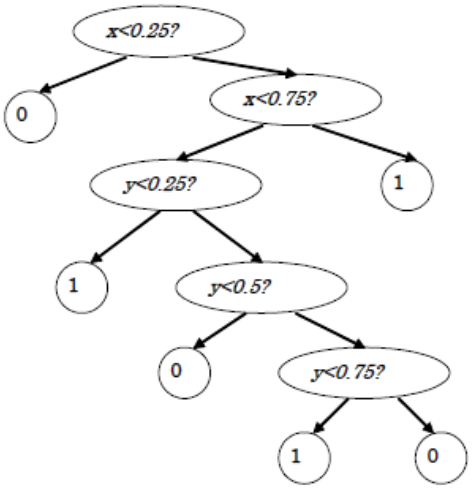
Example 1: Shall I play tennis today ?

- Play tennis if (outlook is sunny & humidity is normal).
- Play tennis if (outlook is overcast).
- Play tennis if (outlook is rain & wind is weak).
- Otherwise not play.



Example 2: A binary decision tree





Basic steps for inducing a DT

Step 1: If all data assigned to the current node belong to the same class, the current node is terminal. Define the class label and return;

Step 2: Otherwise, find the best decision function $f(x)$ based on some criterion (Information Gain Ratio is used in C4.5);

Step 3: Split the data assigned to the current node to N-groups using $f(x)$;

Step 4: For each group, if it is not empty, make a new node, set the current node as this new node, and call the same sub-routine recursively.

- The 'network' consists of a set of perceptrons functionally organized in a binary tree ('neural tree').
- The learning algorithm is inspired from a growth algorithm, the tiling algorithm, recently introduced for feedforward neural networks.
- As in the former case, this is a constructive algorithm, for which convergence is guaranteed.
- In the neural tree one distinguishes the structural organization from the functional organization: each neuron of a neural tree receives inputs from, and only from, the input layer; its output does not feed into any other neuron, but is used to propagate down a decision tree.
- The main advantage of this approach is due to the *local* processing in restricted portions of input space, during both learning and classification.
- Moreover, only a small subset of neurons have to be updated during the classification stage.
- Finally, this approach is easily and efficiently extended to classification in a multiclass problem.
- Promising numerical results have been obtained on different two- and multiclass problems.

Associative Memory Networks

Introduction

- These kinds of neural networks work on the **basis of pattern association**, which means they can store different patterns and at the time of giving an output they can produce one of the stored patterns by matching them with the given input pattern.
- These types of memories are also called **Content-Addressable Memory** (CAM). Associative memory makes a parallel search with the stored patterns as data files.

Pattern Association.

Associative memory neural nets are single-layer nets in which the. weights are determined in such a way that the net can store a set of. **pattern associations**. - Each **association** is an input-output vector pair, s: t.

Following are the two types of associative memories we can observe −

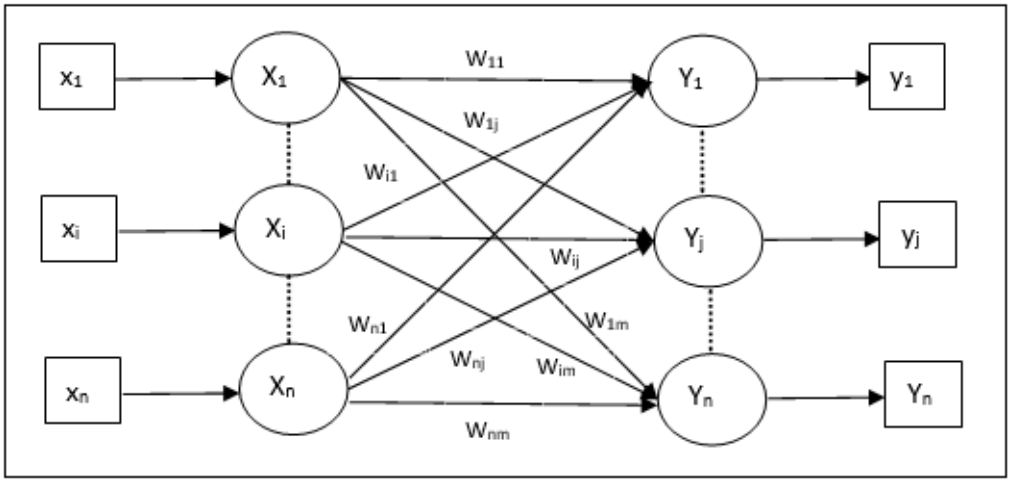
- Auto Associative Memory
- Hetero Associative memory

Auto Associative Memory

- One of the primary functions of the brain is associative memory.
- We associate the faces with names, letters with sounds, or we can recognize the people even if they have sunglasses or if they are somehow elder now.
- This is a single layer neural network in which the input training vector and the output target vectors are the same. The weights are determined so that the network stores a set of patterns.

Architecture

As shown in the following figure, the architecture of Auto Associative memory network has **'n'** number of input training vectors and similar **'n'** number of output target vectors.



In an associative memory, we store a set of patterns μ*k*, *k*=1...*K*, so that the network responds by producing whichever of the stored patterns most closely resembles the one presented to the network

Hetero Associative memory

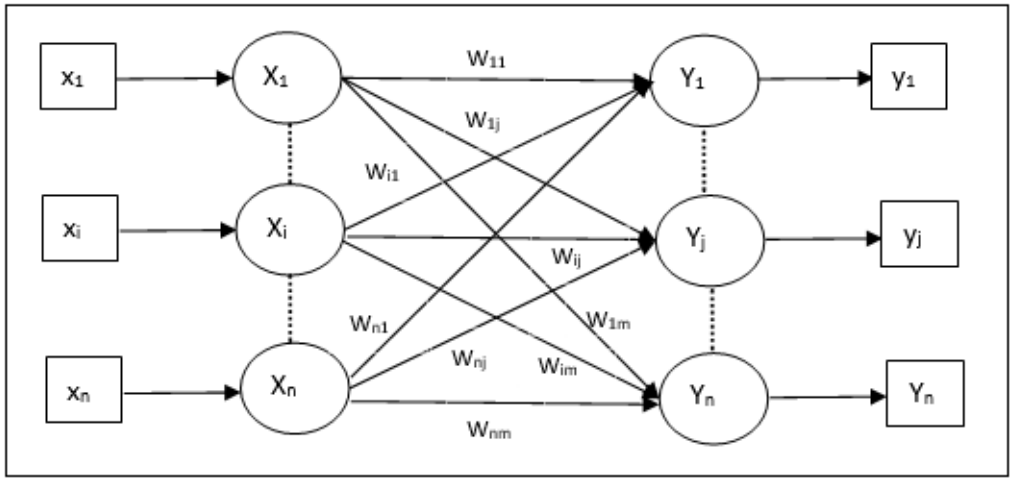
Similar to Auto Associative Memory network, this is also a single layer neural network.

However, in this network the input training vector and the output target vectors are not the same.

The weights are determined so that the network stores a set of patterns. Hetero associative network is static in nature, hence, there would be no non-linear and delay operations.

Architecture

As shown in the following figure, the architecture of Hetero Associative Memory network has **'n'** number of input training vectors and **'m'** number of output target vectors.

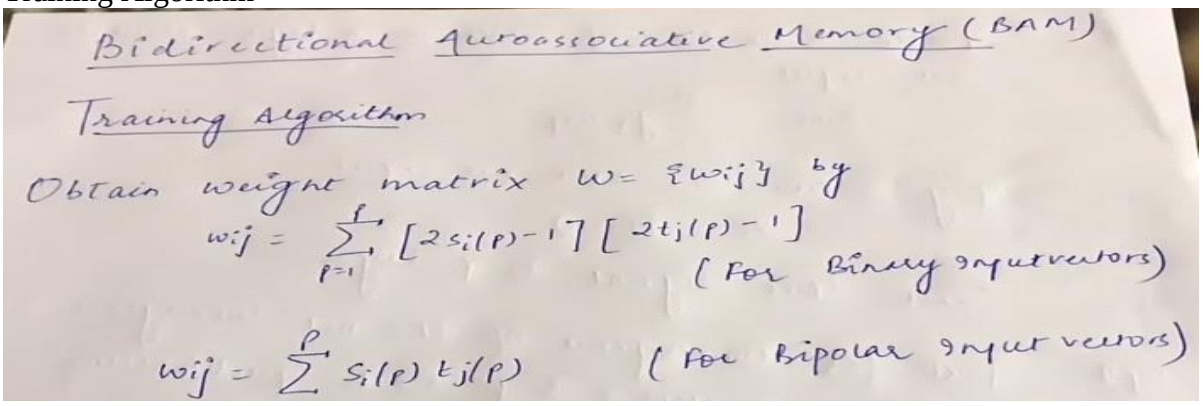


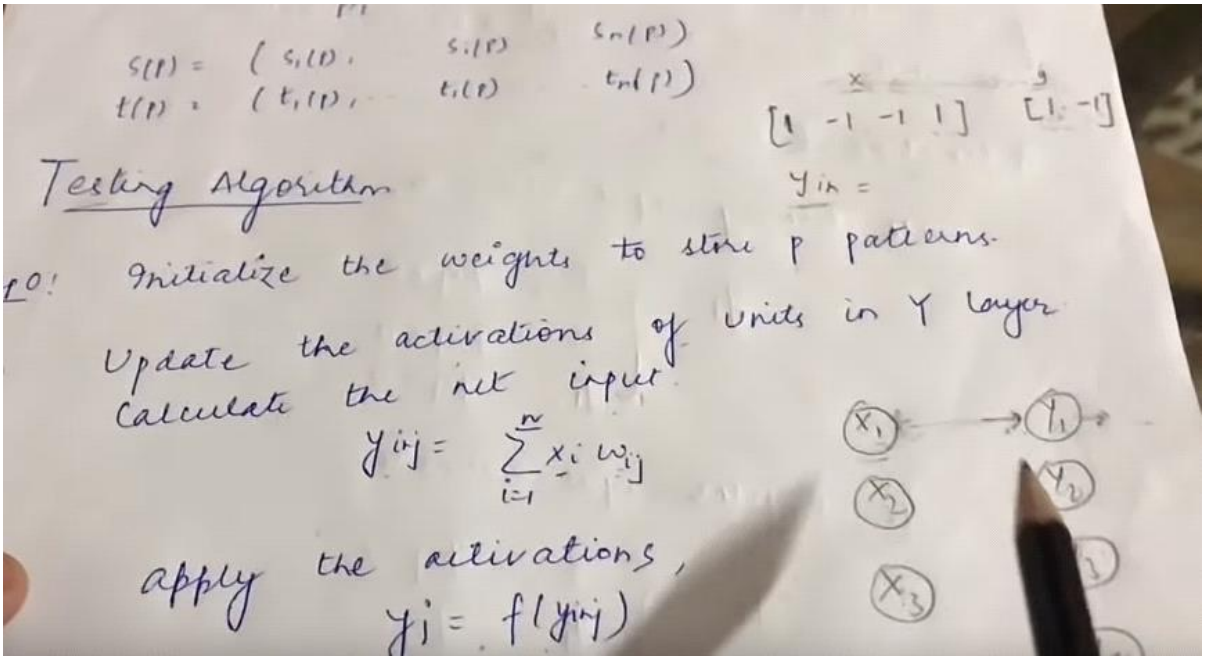
BIDIRECTIONAL ASSOCIATIVE MEMORY (BAM)

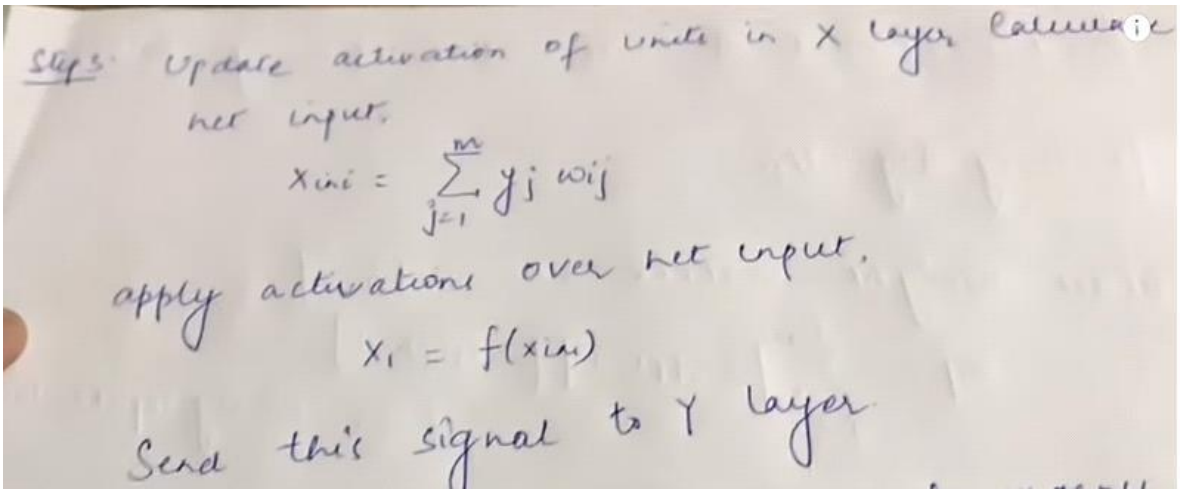
- Several versions of the heteroassociative recurrent neural network, or bidirectional associative memory (BAM), developed by Kosko (1988).

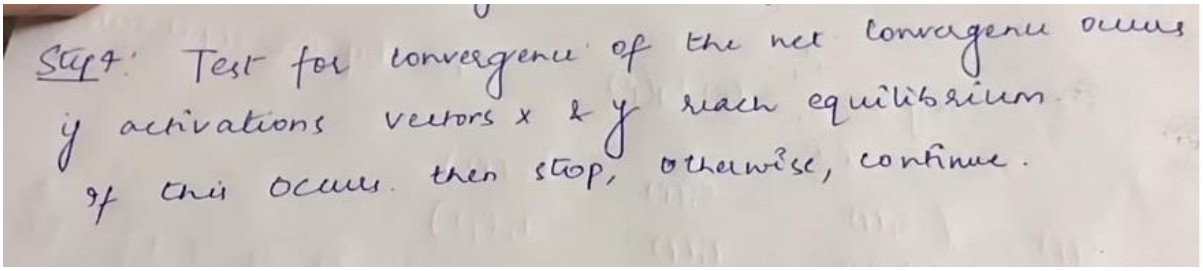
- Bidirectional Associative Memory (BAM) is a type of recurrent neural network.
- BAM has 2 layers: input and output and information can go in both directions - from input to output and back from output to input.
- **BAM** is hetero-**associative**, meaning given a pattern it can return another pattern which is potentially of a different size.
- It is similar to the Hopfield network in that they are both forms of **associative memory**.

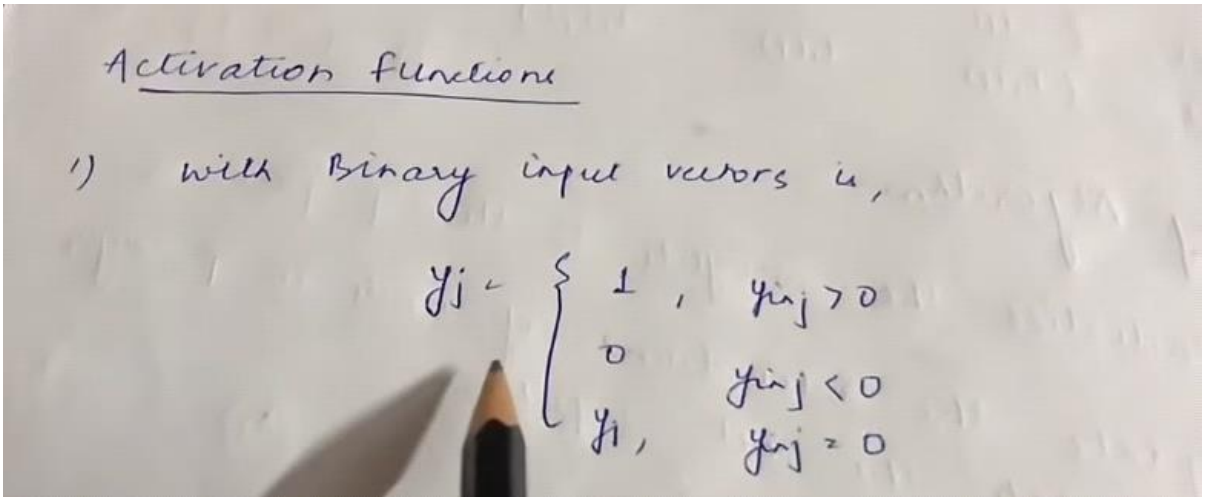
Algorithm
Training Algorithm

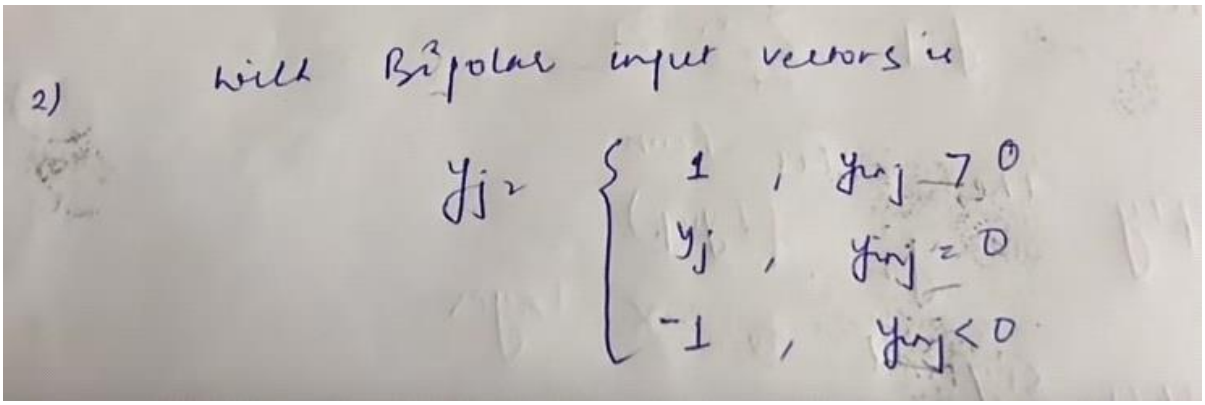


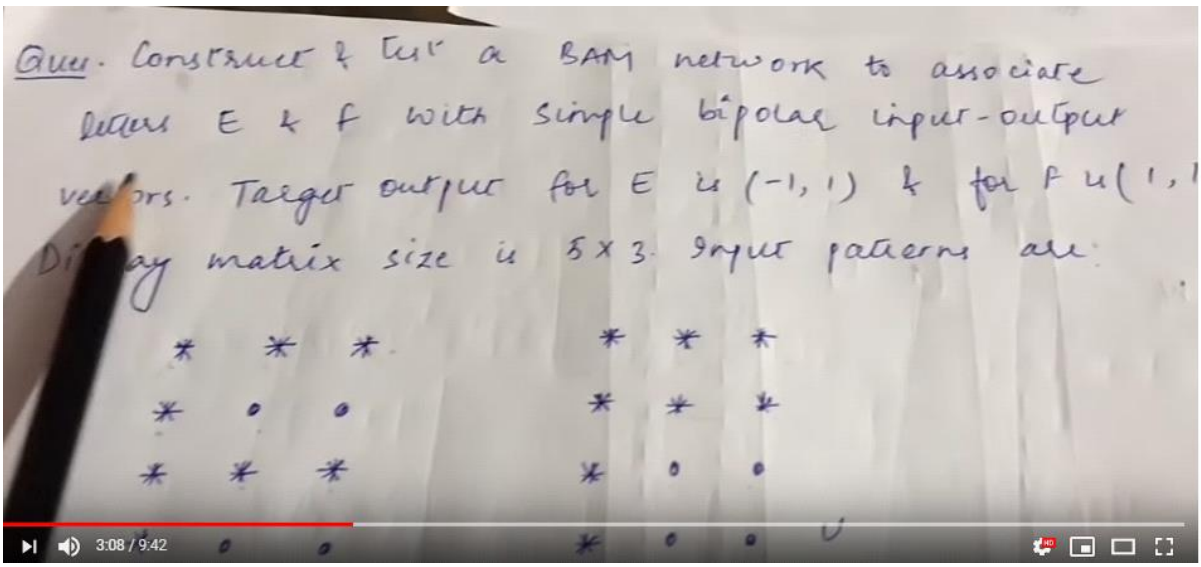


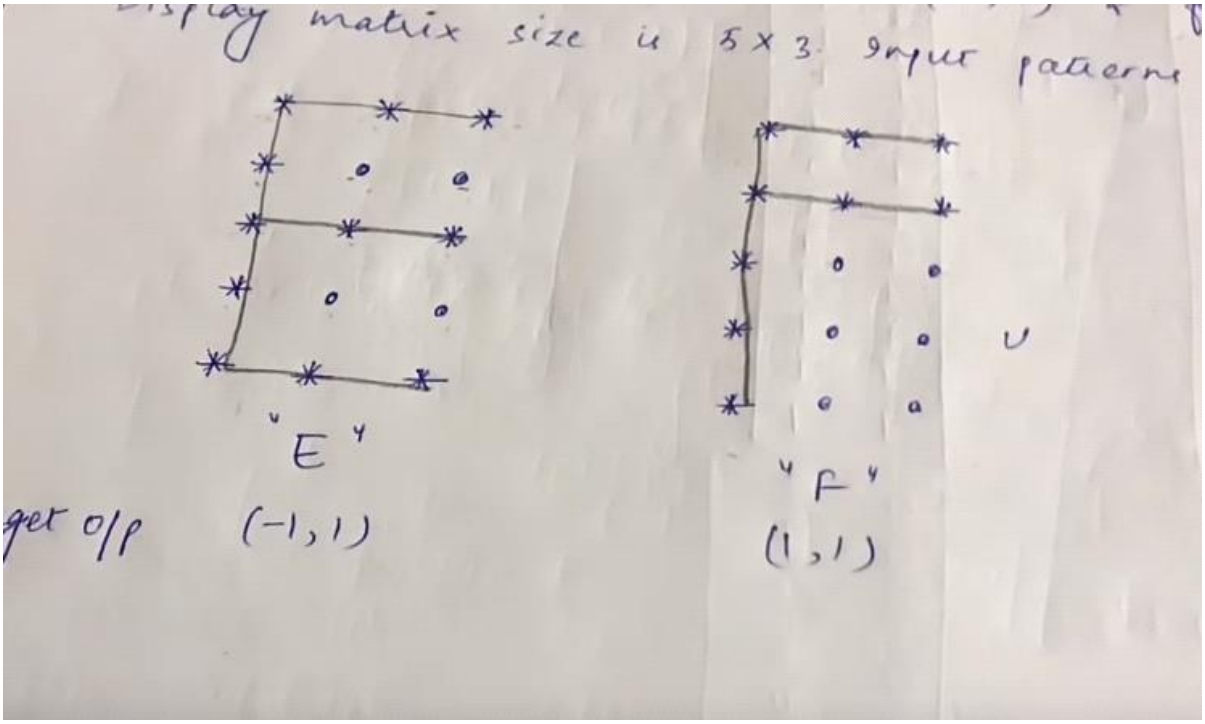

Step 3: Update activation of units in X layer Calculate
net input.
$x_{ini} = \sum_{j=1}^{m} y_j w_{ij}$
apply activations over net input.
$x_i = f(x_{ini})$
Send this signal to Y layer

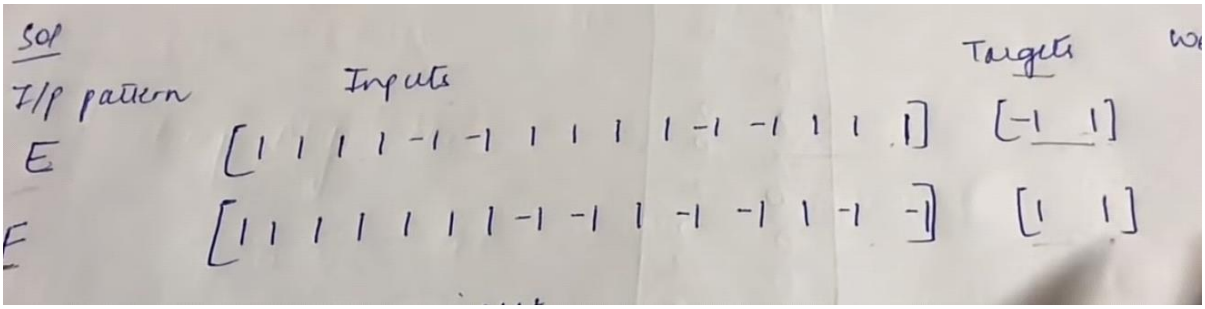

Step 4: Test for convergence of the net convergence occurs
if activations vectors x & y reach equilibrium.
If this occurs then stop, otherwise, continue.

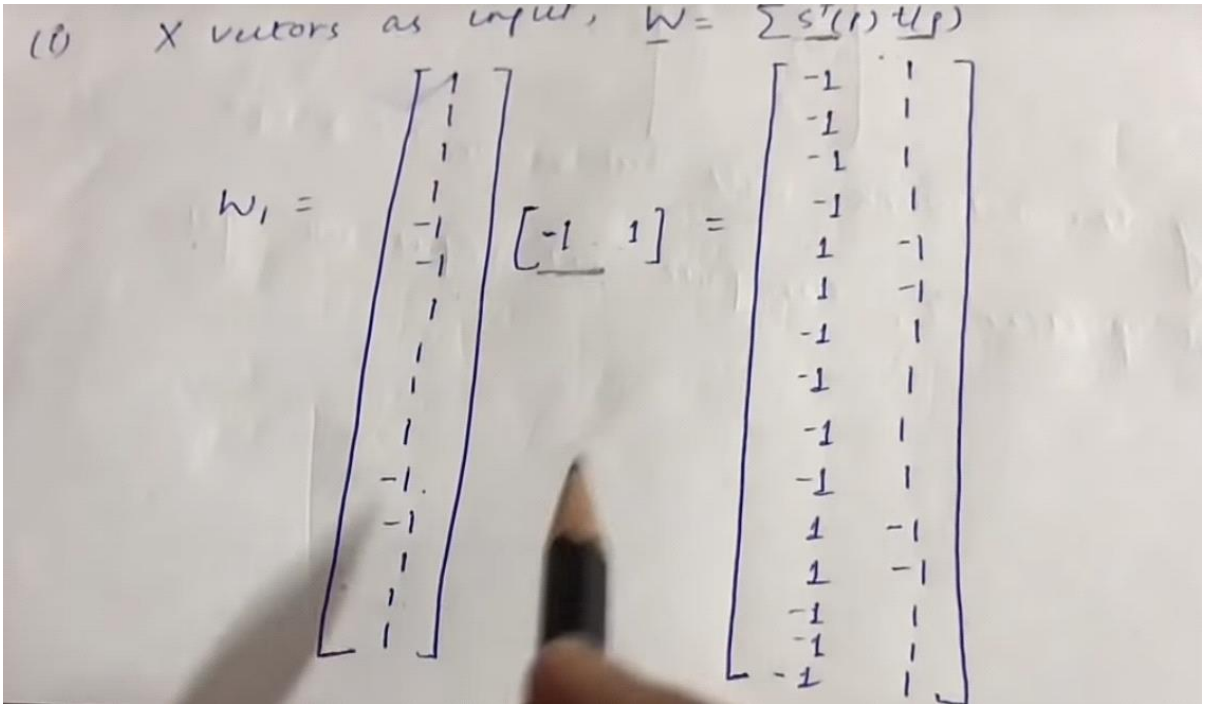

Activation functions
1) with Binary input vectors is,
$y_j = \begin{cases} 1, & y_{inj} > 0 \\ 0 & y_{inj} < 0 \\ y_j, & y_{inj} = 0 \end{cases}$

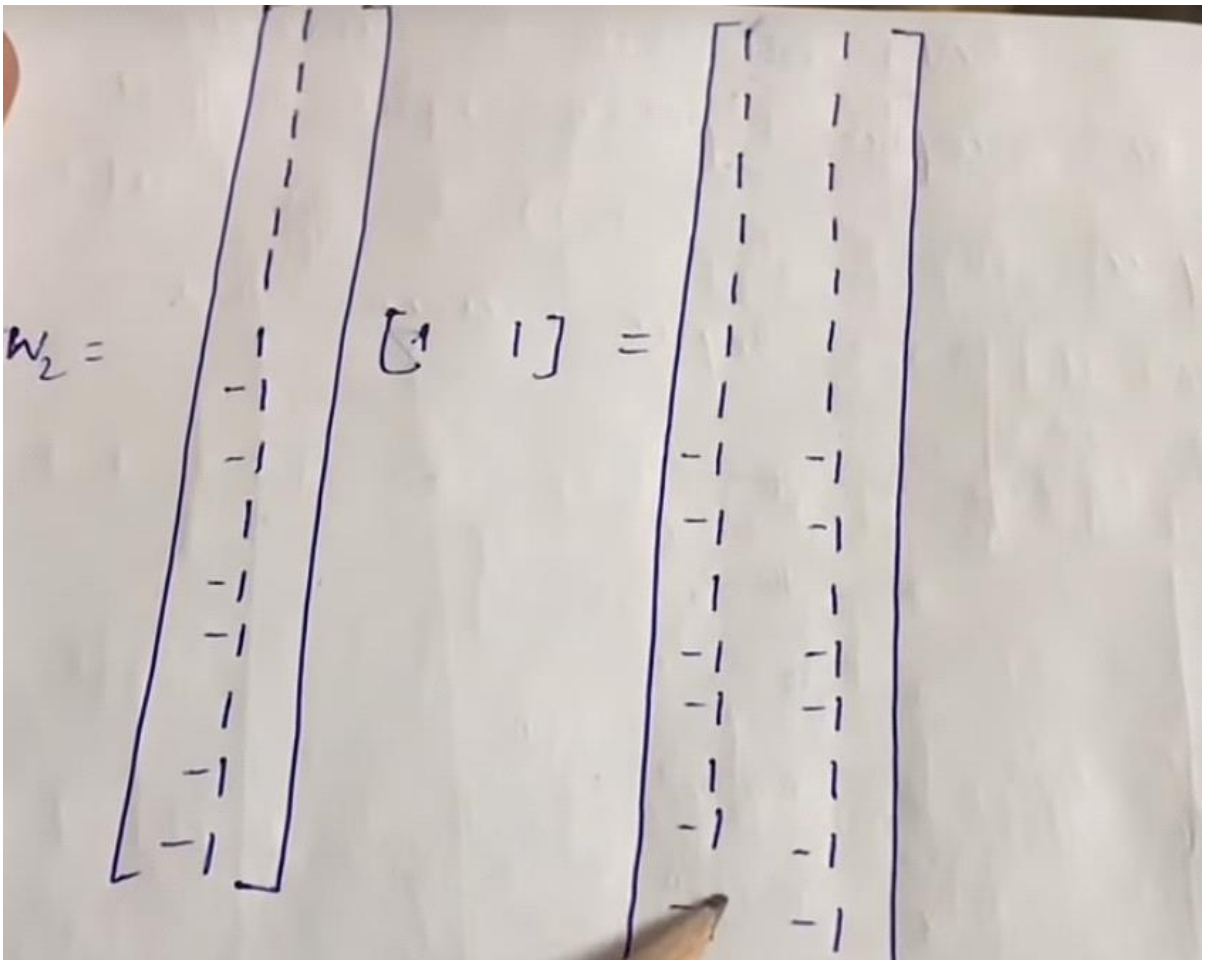

2) With Bipolar input vectors is
$y_j = \begin{cases} 1, & y_{inj} > 0 \\ y_j, & y_{inj} = 0 \\ -1, & y_{inj} < 0 \end{cases}$

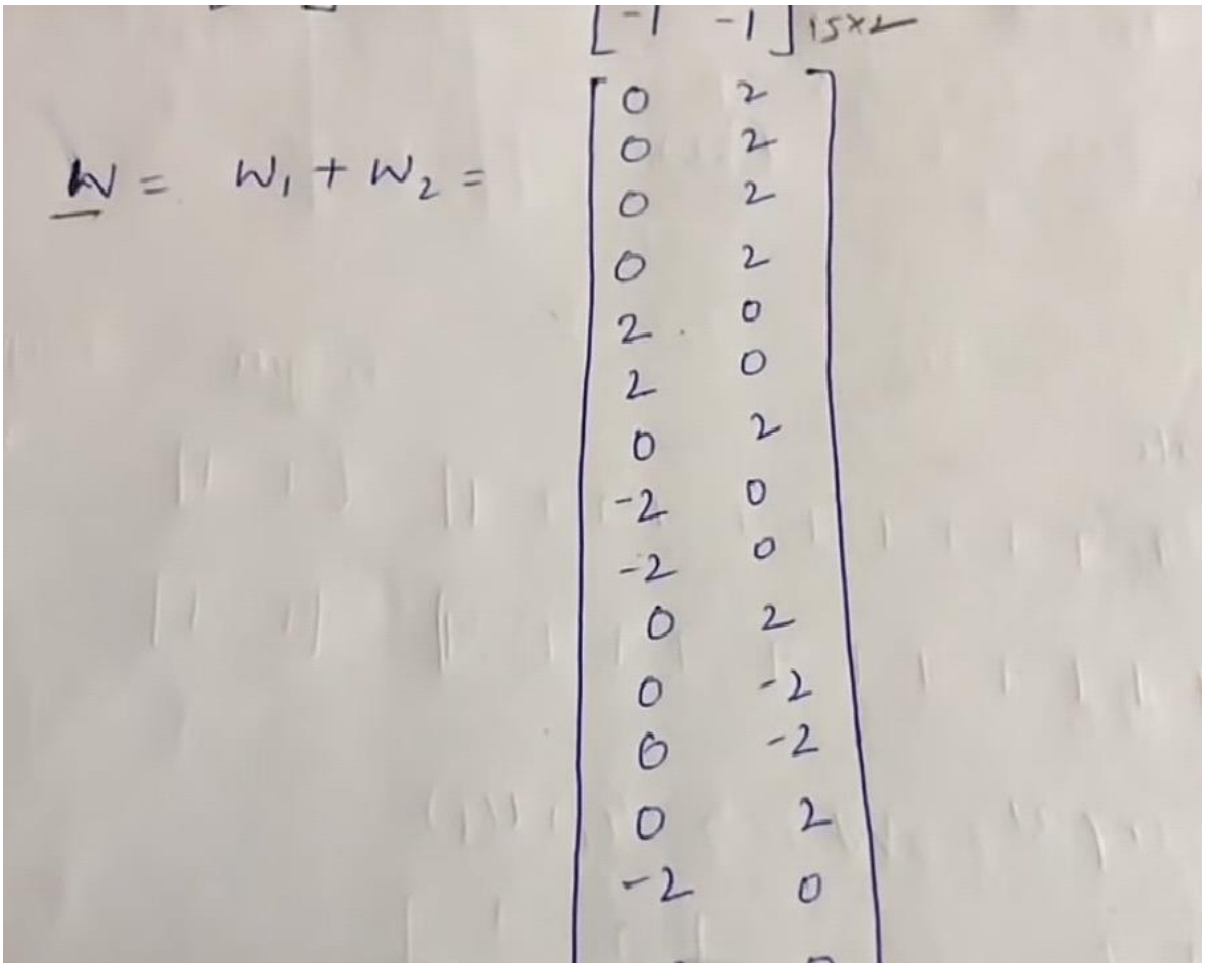

Ques. Construct & test a BAM network to associate letters E & F with simple bipolar input-output vectors. Target output for E is (-1, 1) & for F is (1,
Display matrix size is 5 x 3. Input patterns are
3:08 / 9:42

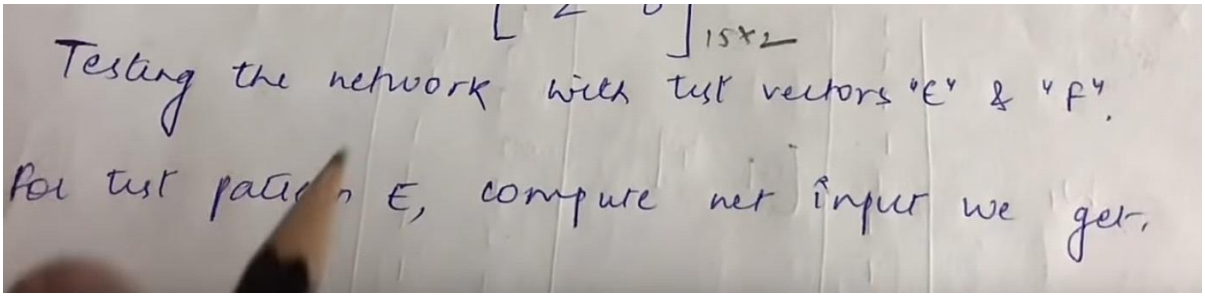

matrix size is 5 x 3 input patterns
"E"
"F"
get o/p (-1, 1)
(1, 1)

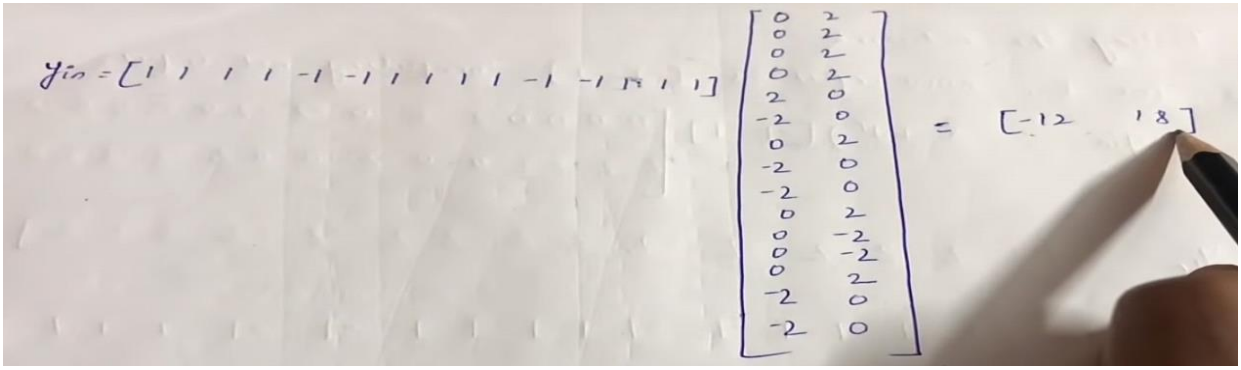

Sol
I/p pattern
Inputs
Targets
E
F
[1 1 1 1 -1 -1 1 1 1 1 -1 -1 1 1 1] [-1 1]
[1 1 1 1 1 1 1 -1 -1 1 -1 -1 1 -1 -1] [1 1]

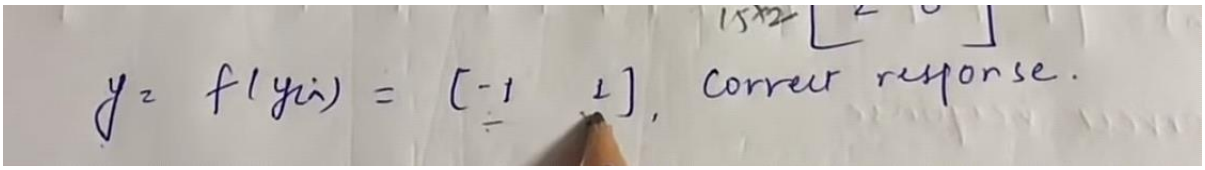

(0 X vectors as
W = Σ s'(i) t(i)
W1 =
[-1 1] =

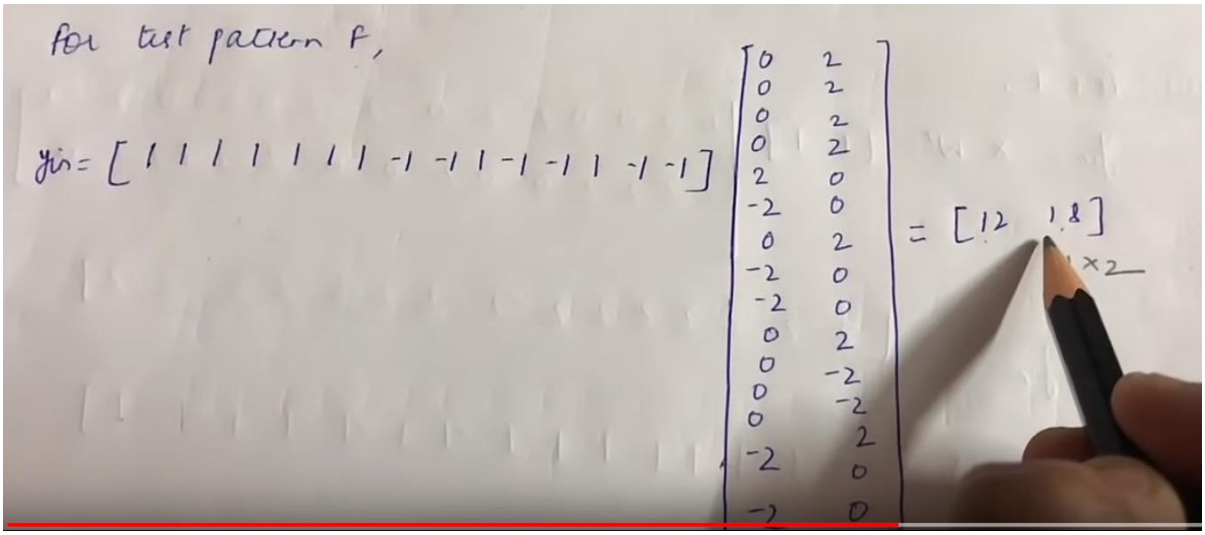

W2 =
[1 1] =

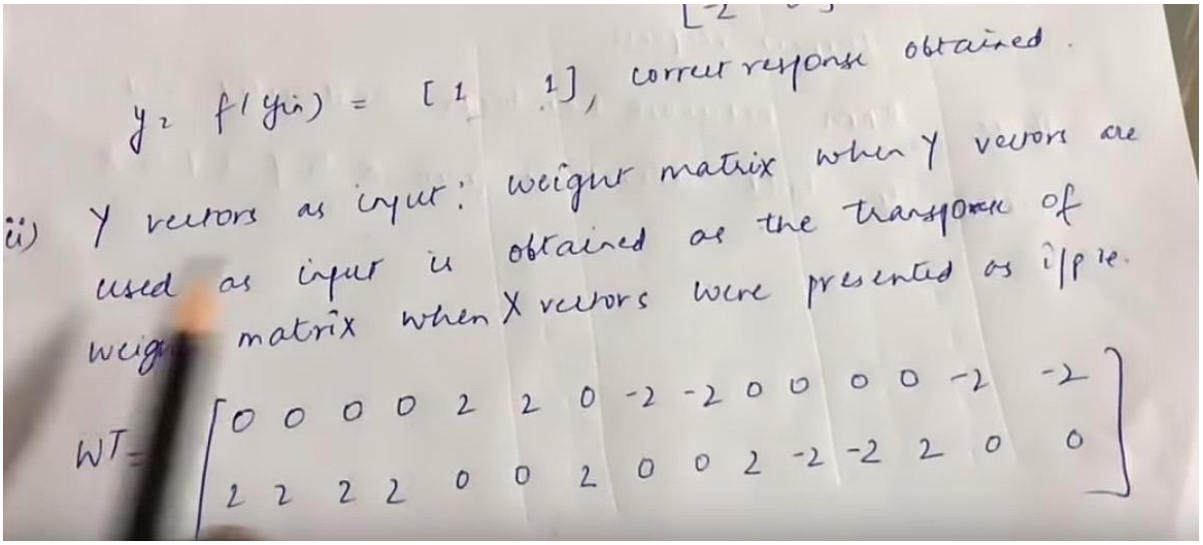

[-1 -1] 15×2
W = W1 + W2 =
0 2
0 2
0 2
0 2
2 0
2 0
2 0
0 2
-2 0
-2 0
0 2
0 -2
0 -2
0 2
-2 0

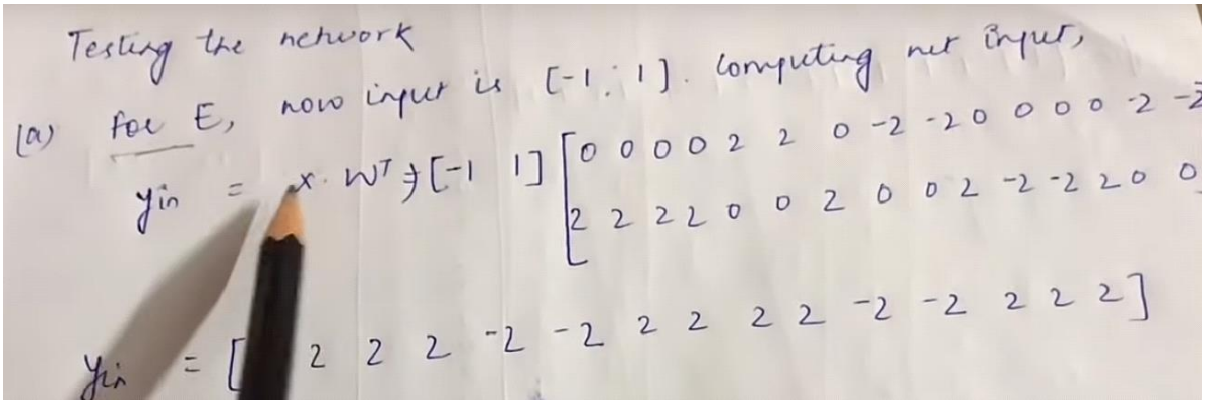

15×2
Testing the network with test vectors "E" & "F".
For test pattern E, compute net input we get,

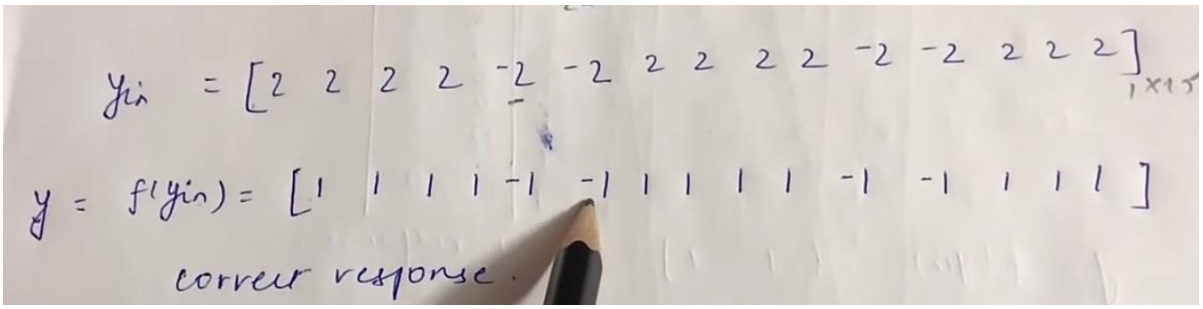

yin = [1 1 1 1 -1 -1 1 1 1 1 -1 -1 1 1]
0 2
0 2
0 2
0 2
2 0
-2 0
0 2
-2 0
-2 0
0 2
0 -2
0 -2
0 2
-2 0
-2 0
= [-12 18]

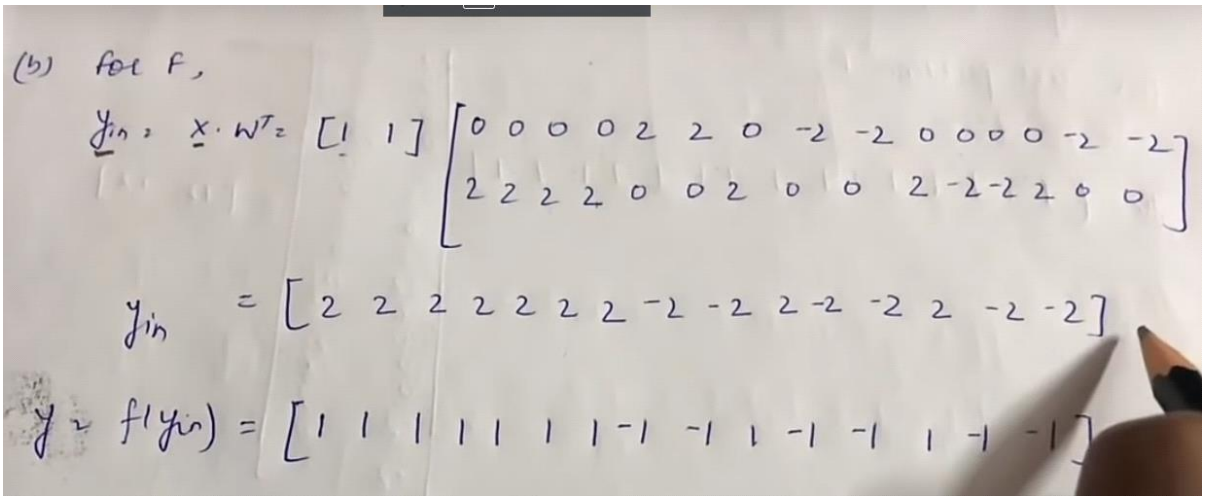

y = f(yin) = [-1 1], Correct response.

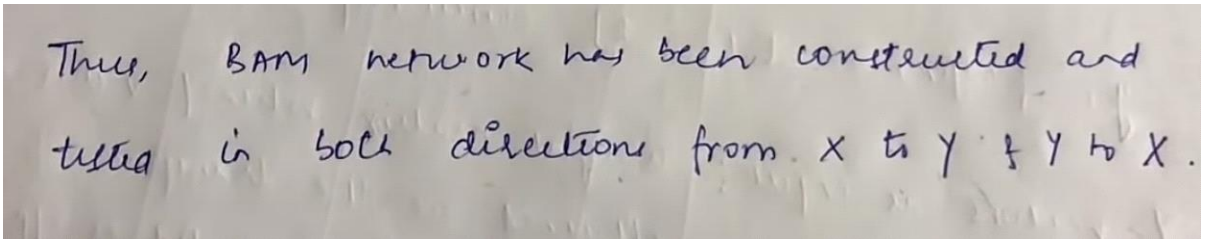

for test pattern f,
yin = [1 1 1 1 1 1 -1 -1 1 -1 -1 1 -1 -1]
= [12 1.8]


y = f(yin) = [1 1], correct response obtained.
ii) Y vectors as input: weight matrix when Y vectors are used as input is obtained as the transpose of weight matrix when X vectors were presented as i/p ie.
WT = [0 0 0 0 2 2 0 -2 -2 0 0 0 0 -2 -2
2 2 2 2 0 0 2 0 0 2 -2 -2 2 0 0]


Testing the network
(a) For E, now input is [-1 1]. Computing net input,
$y_{in} = x \cdot W^T = [-1\ 1] \begin{bmatrix} 0\ 0\ 0\ 0\ 2\ 2\ 0\ -2\ -2\ 0\ 0\ 0\ -2\ -2 \\ 2\ 2\ 2\ 2\ 0\ 0\ 2\ 0\ 0\ 2\ -2\ -2\ 2\ 0\ 0 \end{bmatrix}$
$y_{in} = [2\ 2\ 2\ -2\ -2\ 2\ 2\ 2\ 2\ -2\ -2\ 2\ 2\ 2]$


$y_{in} = [2\ 2\ 2\ 2\ -2\ -2\ 2\ 2\ 2\ 2\ -2\ -2\ 2\ 2\ 2]_{1\times 15}$
$y = f(y_{in}) = [1\ 1\ 1\ 1\ -1\ -1\ 1\ 1\ 1\ 1\ -1\ -1\ 1\ 1\ 1]$
correct response.


(b) For F,
$y_{in} = x \cdot W^T = [1\ 1] \begin{bmatrix} 0\ 0\ 0\ 0\ 2\ 2\ 0\ -2\ -2\ 0\ 0\ 0\ 0\ -2\ -2 \\ 2\ 2\ 2\ 2\ 0\ 0\ 2\ 0\ 0\ 2\ -2\ -2\ 2\ 0\ 0 \end{bmatrix}$
$y_{in} = [2\ 2\ 2\ 2\ 2\ 2\ 2\ -2\ -2\ 2\ -2\ -2\ 2\ -2\ -2]$
$y = f(y_{in}) = [1\ 1\ 1\ 1\ 1\ 1\ 1\ -1\ -1\ 1\ -1\ -1\ 1\ -1\ -1]$


Thus, BAM network has been constructed and tested in both directions from X to Y & Y to X.

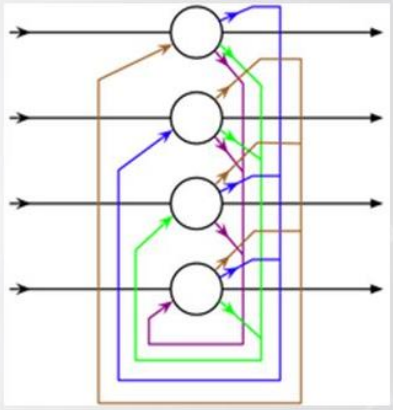
Hopfield network

- A Hopfield network is a form of recurrent artificial neural network popularized by John Hopfield in 1982, but described earlier by Little in 1974.
- Hopfield nets serve as content-addressable ("associative") memory systems with binary threshold nodes.
- Hopfield networks also provide a model for understanding human memory.
- Hopfield neural networks represent a new neural computational paradigm by implementing an autoassociative memory.
- They are recurrent or fully interconnected neural networks.
- There are two versions of Hopfield neural networks: in the binary version all neurons are connected to each other but there is no connection from a neuron to itself, and in the continuous case all connections including self-connections are allowed.

Hopfield network (HN) model

- A Hopfield network is a form of **recurrent** artificial neural network invented by John Hopfield in 1982
- The network is entirely **interconnected**
  - All neurons **are both input and output**
- **with binary threshold units**

$$F(x) = \begin{cases} 1, & \text{if } x > 0 \\ -1, & \text{otherwise} \end{cases}$$



These are single layered recurrent networks