VIGNAN'S LARA
INSTITUTE OF TECHNOLOGY & SCIENCE
Approved by AICTE New Delhi & Affiliated to JNTUK Kakinada
Accredited by **NAAC 'A+'** and **NBA** | **ISO 9001 : 2015**
Vadlamudi - 522 213, Guntur District

UNIT 2: Artificial Neural Networks and Paradigms

- Introduction to Neuron Model
- Neural Network Architecture
- Learning Rules
- Perceptrons-Single Layer Perceptrons
- Multilayer Perceptrons
- Back propagation Networks
- Kohnen's self organizing networks
- Hopfield network
- Applications of NN

VIGNAN'S LARA
INSTITUTE OF TECHNOLOGY & SCIENCE
Approved by AICTE New Delhi & Affiliated to JNTUK Kakinada
Accredited by NAAC 'A+' and NBA | ISO 9001 : 2015
Vadlamudi - 522 213, Guntur District

IV B.Tech II Sem (R20)

Assignment-2 Question Bank
Neural Networks and Soft Computing(NNSC)

(*Computer Science Engineering*)

Question No.	Question	Marks allotted	Course Outcome	Bloom's Taxonomy	Page No.
1	Define Perceptron? Explain in brief about Perceptron Model?	5	1	U	Unit-2 3
2	What are the various types of neuron activation functions? Explain with a neat sketch.	5	1	U	Unit-2 8-10
3	Discuss various learning rules of Artificial Neural Networks	5	1	Ap	Unit-2 25-26
4	Explain the following architectures of the Neural Networks i. Single Layer Feed Forward Neural Networks ii. Multi Layer Feed Forward Neural Networks iii. Feed Back Neural Networks	5	1	E	Unit-2 11-14
5	a) How to achieve the performance of Back Propagation Learning? Discuss in detail. b) What are the limitations of Back Propagation algorithm?	5	1	Ap	Unit-2 14-18

Question No.	Question	Marks allotted	Course Outcome	Bloom's Taxonomy	Page No.
6	Discuss the Kohnen's self organizing networks	5	1	Ap	Unit-2 22-23
7	Give the architectural description and applications of Hopfield Neural network	5	1	Ap	Unit-2 19-21

R - Remember, **U** - Understand, **Ap** – Apply, **A** - Analyse, **E**- Evaluate, **C**- Create

CO1: Understand the concepts of Artificial intelligence and soft computing techniques

CO2: Analyze the concepts of Neural Networks and select the Learning Networks in modeling real world systems.

CO3: Implement the concepts of Fuzzy reasoning and concepts of Genetic algorithm and its applications to soft computing.

CO4: Classify Biologically inspired algorithm such as neural networks, genetic algorithms, ant colony optimization, and bee colony optimization.

CO5: Design hybrid system incorporating neural network, genetic algorithms, fuzzy systems

Introduction

Neural Networks, which are simplified models of the biological neuron system, is a massively parallel distributed processing system made up of highly interconnected neural computing elements that have the ability to learn and thereby acquire knowledge and make it available for use. Various learning mechanisms exist to enable the NN acquire knowledge.

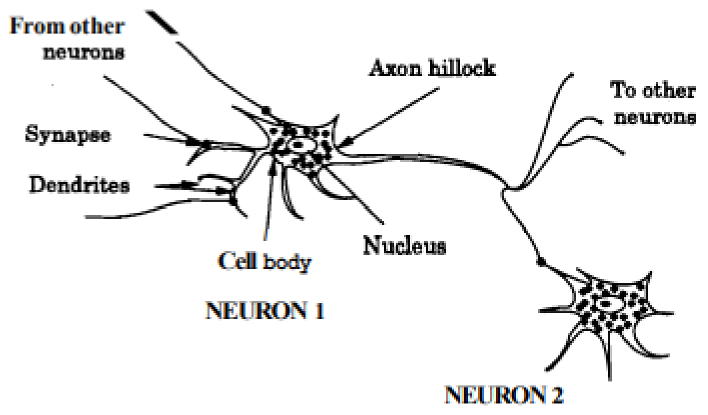
NN architectures have been classified into various types based on their learning mechanisms and other features. Some classes of NN refer to this learning process as training and the ability to solve a problem using the knowledge acquired as inference.

NNs are simplified imitations of the central nervous system, and obviously therefore, have been motivated by the kind of computing performed by the human brain. The structural constituents of a human brain termed neurons are the entities, which perform computations such as cognition, logical inference, pattern recognition and so on. Hence the technology, which has been built on a simplified imitation of computing by neurons of a brain, has been termed Artificial Neural Systems (ANS) technology or Artificial Neural Networks (ANN) or simply Neural Networks.

In the literature, this technology is also referred to as Connectionist Networks, Neuro-Computers, Parallel Distributed Processors etc. Also neurons are referred to as neurodes, Processing Elements (PEs), and nodes. In this book, we shall use the terms Neural Networks or Artificial Neural Networks and neurons.

Biological Neural Networks

The features of the biological neural network are attributed to its structure and function. The fundamental unit of the network is called a neuron or a nerve cell. Figure 1.1 shows a schematic of the structure of a neuron. It consists of a cell body or soma where the cell nucleus is located. Treelike nerve fibres called dendrites are associated with the cell body. These dendrites receive signals from other neurons. Extending from the cell body is a single long fibre called the axon, which eventually branches into strands and substrands connecting to many other neurons at the synaptic junctions, or synapses. The receiving ends of these junctions on other cells can be found both on the dendrites and on the cell bodies themselves. The axon of a typical neuron leads to a few thousand synapses associated with other neurons.



The transmission of a signal from one cell to another at a synapse is a complex chemical process in which specific transmitter substances are released from the sending side of the junction. The effect is to raise or lower the electrical potential inside the body of the receiving cell. If this potential reaches a threshold, an electrical activity in the form of short pulses is generated. When this happens, the cell is said to have fired. These electrical signals of fixed strength and duration are sent down the axon. Generally the electrical activity is confined to the interior of a neuron, whereas the chemical mechanism operates at the synapses. The dendrites serve as receptors for signals from other neurons, whereas the purpose of an axon is transmission of the generated neural activity to other nerve cells (inter-neuron) or to muscle fibres (motor neuron). A third type of neuron, which receives information from muscles or sensory organs, such as the eye or ear, is called a receptor neuron. The size of the cell body of a typical neuron is approximately in the range 10-80 micrometers (pm) and the dendrites and axons have diameters of the order of a few pm. The gap at the synaptic junction is about 200 nanometers (nm) wide. The total length of a neuron varies from 0.01 mm for internal neurons in the human brain up to 1 m for neurons in the limbs. In the state of inactivity the interior of the neuron, the protoplasm, is negatively charged against the surrounding neural liquid containing positive Sodium (Na+) ions.. The resulting resting potential of about - 70 mV is supported by the action of the cell membrane, which is impenetrable for the positive Sodium ions. This causes a deficiency of positive ions in the protoplasm. Signals arriving from the synaptic connections may result in a temporary depolarization of the resting potential. When the potential is increased to a level above - 60 mV, the membrane suddenly loses its impermeability against Na+ ions, which enter into the protoplasm and reduce the potential difference. This sudden change in the membrane potential causes the neuron to discharge. Then the neuron is said to have fired. The membrane then gradually recovers its original properties and regenerates the resting potential over a period of several milliseconds. During this recovery period, the neuron remains incapable of further excitation. The discharge, which initially occurs in the cell body, propagates as a signal along the axon to the synapses. The intensity of the signal is encoded in the frequency of the sequence of pulses of activity, which can range fiom about 1 to 100 per second. The speed of propagation of the discharge signal in the cells of the human brain is about 0.5-2 mls. The discharge signal travelling along the axon stops at the synapses, because there exists no conducting link to the next neuron. Transmission of the signal across the synaptic gap is mostly effected by chemical activity. When the signal arrives at the presynaptic nerve terminal, special substances called neurotransmitters are produced in tiny amounts. The neurotransmitter molecules travel across the synaptic junction reaching the postsynaptic neuron within about 0.5 ms. These substances modify the conductance of the postsynaptic membrane for certain ions, causing a polarization or depolarization of the postsynaptic potential. If the induced polarization potential is positive, the synapse is termed excitatory, because the influence of the synapse tends to activate the postsynaptic neuron. If the polarization potential is negative, the synapse is called inhibitory, since it counteracts excitation of the neuron. All the synaptic endings of an axon are either of an excitatory or an inhibitory nature.

Models of Neuron

In this section we will consider three classical models for an artificial neuron or processing unit.

McCulloch-Pitts Model

In McCulloch-Pitts (MP) model (Figure 1.2) the activation (x) is given by a weighted sum of its M input values (a_i) and a bias term (8). The output signal (s) is typically a nonlinear function flx) of the activation value x. The following equations describe the operation of an MP model:

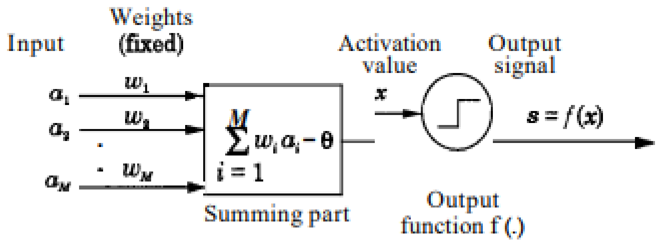


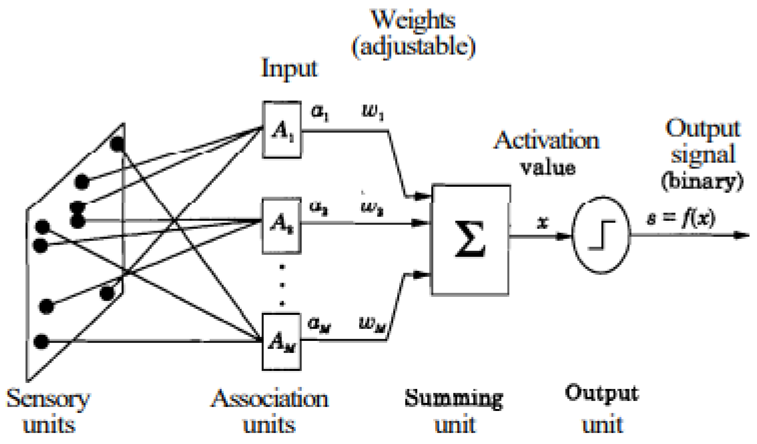
Figure 1.2 McCulloch-Pitts model of a neuron.

Activation: $x = \sum_{i=1}^{M} w_i a_i - \theta$

Output signal: $s = f(x)$

Perceptron Model

The Rosenblatt's perceptron model (Figure 1.5) for an artificial neuron consists of outputs from sensory units to a fixed set of association units, the outputs of which are fed to an MP neuron



The association units perform predetermined manipulations on their inputs. The main deviation from the MP model is that learning (i.e., adjustment of weights) is incorporated in the operation of the imit. The desired or target output (b) is compared with the actual binary output (s), and the error (delta) is used to adjust the weights. The following equations describe the operation of the perceptron model of a neuron:

Activation: $$x = \sum_{i=1}^{M} w_i a_i - \theta$$

Output signal: $s = f(x)$

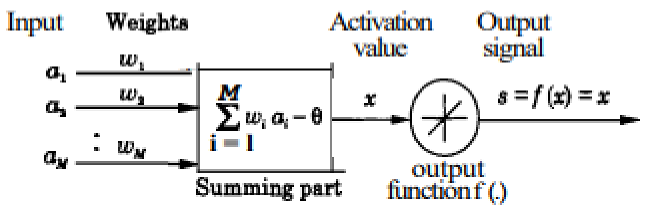
Error: 6 = b – s

Weight change: Aw, = 76a_i

where η is the learning rate parameter.

Adaline

ADAptive LINear Element (ADALINE) is a computing model proposed by Widrow and is shown in Figure



Widrow's Adaline model is that, in the Adaline the analog activation value (x) is compared with the target output (b). In other words, the output is a linear fundion of the activation value (x). The equations that describe the operation of an Adaline are as follows:

Activation:	$x = \sum_{i=1}^{M} w_i a_i - \theta$
Output signal:	$s = f(x) = x$
Error:	$6 = b - s = b - x$
Weight change:	$\Delta w_i = \eta \delta a_i$

Learning methods of ANN

The neural networks are trained by three basic learning paradigms supervisory, unsupervisory, and reinforced learning.

Supervisory Learning

Supervisory learning is nothing but the learning that takes place with the help of the supervisor. In this learning, the inputs, as well as the targeted outputs, will be specified. This learning can be explained with the help of Figure .

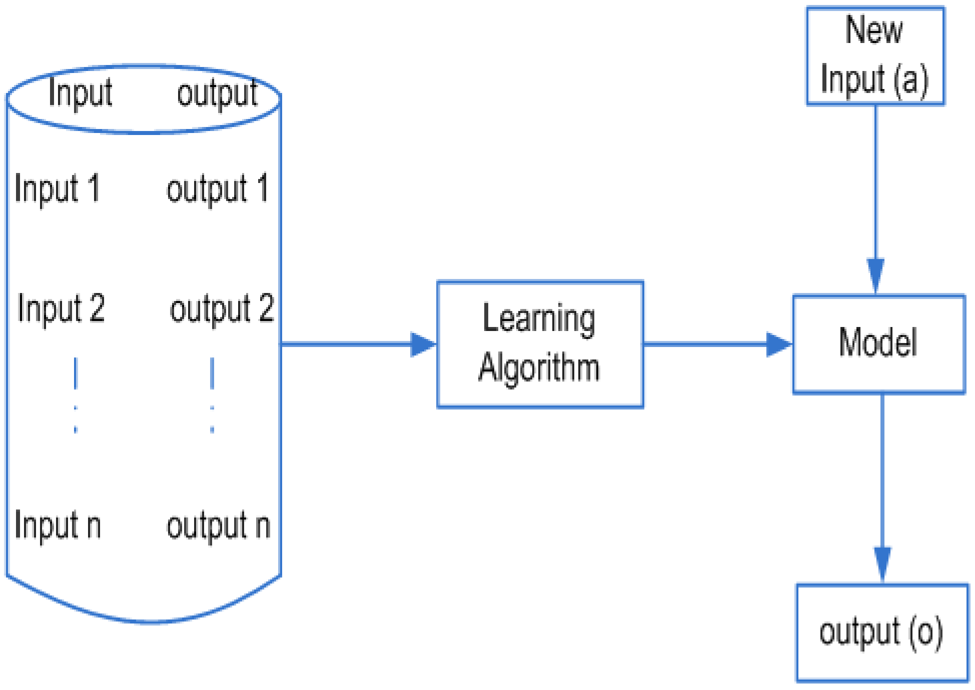


Figure Block diagram of Supervisory Learning method

As it is supervisory learning input as well as the output will be specified. One pair of input, as well as output, can be called a training pattern or instance. If the data is given to the learning algorithm it will the network and come up with the model which is nothing but the trained network. Once the network is trained for a new input ‘a’ it can able to produce the output ‘o’.

Unsupervisory Learning:

In Unsupervisory Learning no teacher is present to supervise the learning process and it contains only inputs. This learning process can be explained using Figure .

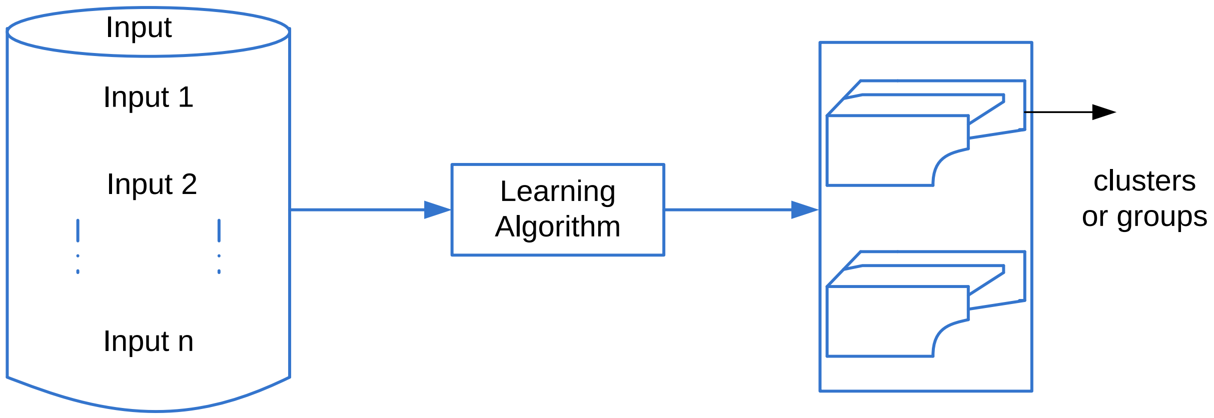


Figure Block diagram of un-supervisory learning method

Whenever input data is given to the learning algorithm it forms clusters or groups based on the similarity. The learn algorithm forms the clusters only after training for example if a set of images is given consisting of humans, trees, and animals the trained model can classify them as clusters of individual humans, trees, and animals.

Reinforced learning:

Let us consider an agent is working in a real-time environment. Whenever a problem comes agent can take the action. This action impacts the environment and already the environment is at a state S_t and it is changed to S_{t+1}. For every action taken by the agent, there is there will be a reward. The reward may be positive or negative. Like this in the reinforced learning, the training process takes place. The learning process can be clearly explained in Figure.

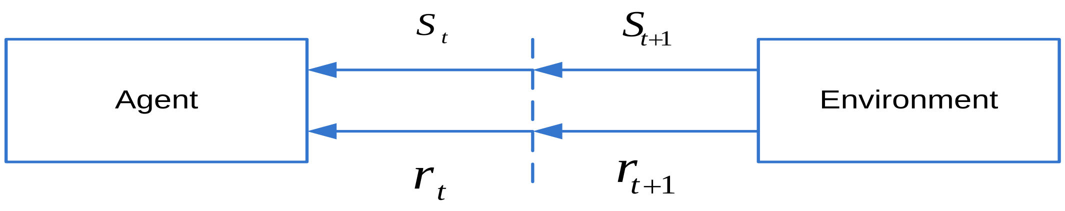


Figure Block diagram of reinforced learning

Activation Functions:

The neuron output is calculated by processing the weighted sum (or) NET value through the activation function nothing but activation functions are mainly used to find the output of a neuron. The following are the different activation functions.

Binary activation function:

In this, the neuron will not produce output if the activation value is less than or equal to zero and it produces output if it is greater than zero. The Binary step function is as shown in Figure .

Mathematically it can be represented as

$$f(x) = 0, x \leq 0$$
$$f(x) = 1, x > 0$$

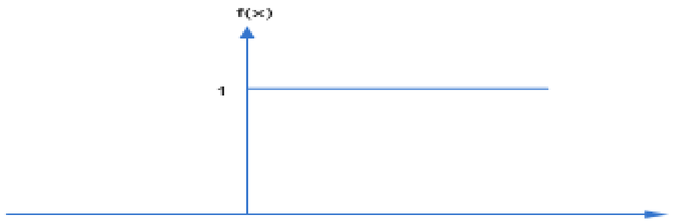


Figure. Binary Activation function

Step Activation function:

The output is zero if the activation value is less than t and it is equal to 1 if the activation value is more than or equal to t. The step activation function is as shown in Figure.

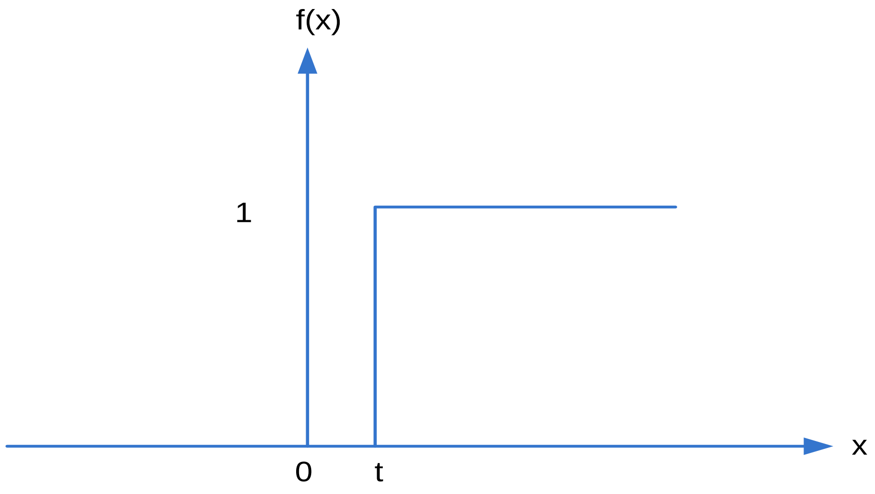


Figure Step activation function

It can be mathematically written as

$$f(x) = 0, x < t$$
$$f(x) = 1, x \geq t$$

Sign Activation Function:

The sign activation function will produce output 1 if the activation value is greater than zero and produces -1 if the activation value is -1. The sign activation function is as depicted in below Figure.

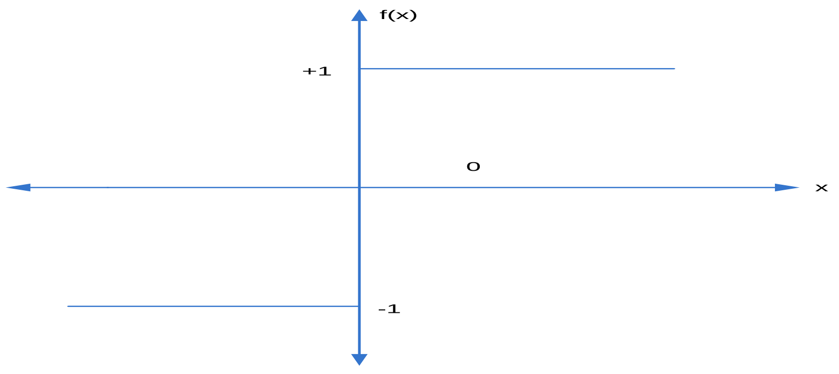


Figure Sign Activation function

Mathematically it can be represented as

$$f(x) = 1, x > 0$$
$$f(x) = -1, x < 0$$

Sigmoidal activation function

The output is obtained by processing the activation function value through the sigmoidal function and it is as depicted in Figure .

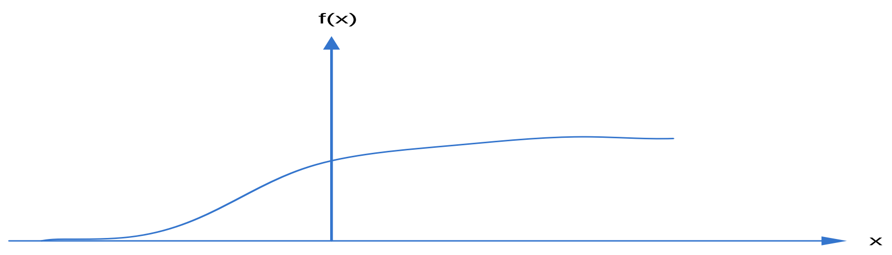


Figure Sigmoidal activation function

Mathematically it can be represented as

$$f(x) = \frac{1}{1+e^{-x}}$$

Architectures of ANN

There is mainly three different types of architectures of artificial neural networks such as a single layer, multilayer, and feedback neural networks.

Single-layer Neural Networks

This form of neural network will have input and the output layers. The input layer neurons receive the input and neurons in the output layer produce the output. The weighted links interconnect the layers and transfers the information between the layers. Despite the two layers, the computations are performed only at output layer. The single-layer NN is as depicted in Figure 4.9. These neural networks are having some limitations such as only applicable to linear separable problems.

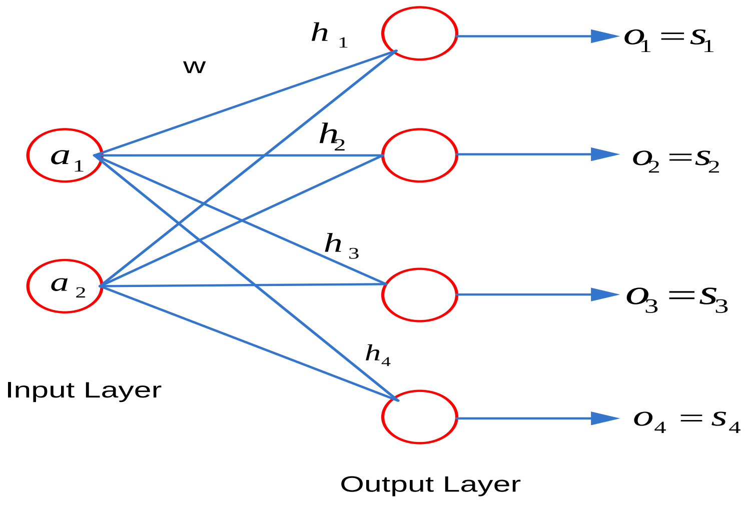


Figure Single layer neural networks

Multi Layer Neural Network

It consists of input, hidden, and output layers . The input layer only supplies the input and no computations are performed at this layer. Weighted connections interconnect the input and hidden layers. Once the input is applied hidden layer, the weighted sum or activation value will be calculated. All these activation values are passed through the activation functions to evaluate the output of all hidden layer neurons. The outputs of the hidden layer neurons are given as input to the output layer.

The hidden and output layers are also interconnected by weighted connections. Again weighted sum or activation values will be calculated at each neuron in the output layer and these values will be processed through the activation value to determine the output at each neuron of the output layer. Even though these networks can be applied to both linearly separable and nonlinear separable problems but they are suffering from storage-related problems. It is as depicted in Figure.

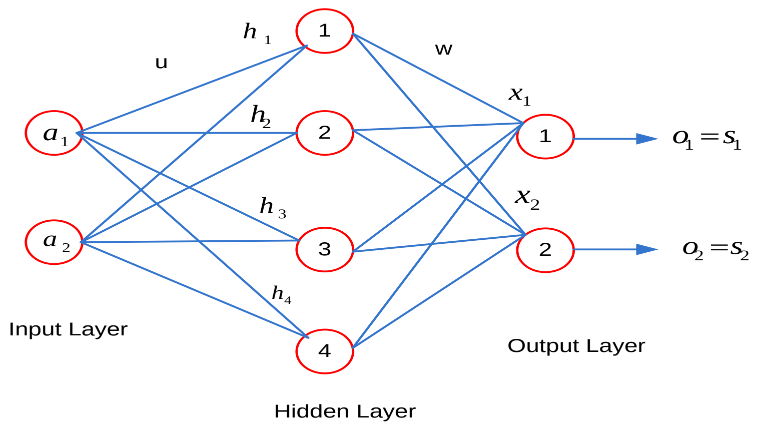


Figure Architecture of Multilayer Neural Networks

Feed Back or Recurrent Neural Networks

The artificial neural networks whose output is fed back to the input are called feedback or recurrent networks. In these types of networks, the present output not only depends upon the input but also on the previous output. Examples of such types of problems are stock market price, weather forecasting, etc.

A scientist called John Hopfield had made important contributions to feedback networks. So these feedback networks are popular with his name and also called the Hopfield network. Two different Hopfield networks are there one is discrete and the other is a continuous Hopfield neural network. If the output is determined using discrete activation function then it is called a discrete Hopfield network and the continuous activation function like sigmoid is used to determine the output then it is called a continuous Hopfield network. The Architecture of the Hopfield network is as shown in Figure 4.11.

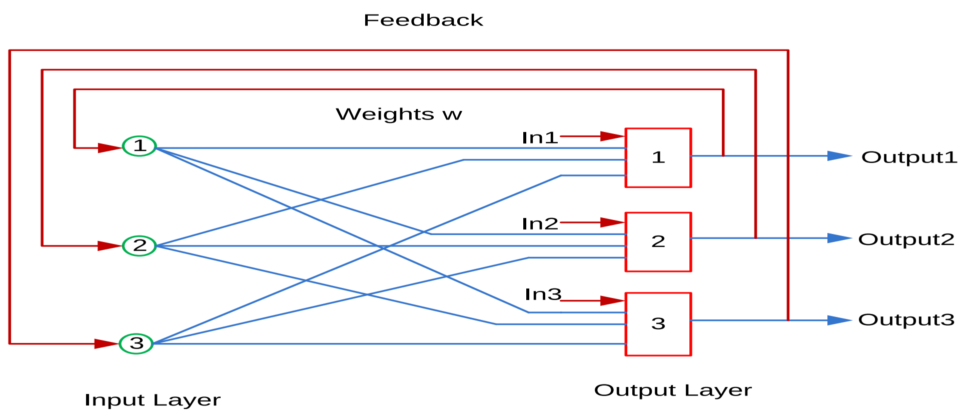


Figure. Feed Back Neural Network

Training Algorithms

Training algorithms provide detail and step-by-step procedure for training the neural network.

Back Propagation Algorithm

Back Propagation algorithm(BP) provides a systematic method to train the Multi-Layer Neural Networks (MLNN). It is also called as back propagation rule which makes use of the gradient descent method. For the training purpose the BP algorithm uses extended gradient-descent-based method. It is a computationally efficient method for optimizing the weights in a feed-forward network. Training is nothing but modifying the weights to minimize the squared error such that the actual output is nearly equal to target output.

According to the BP algorithm, inputs will be applied to hidden layer and activation values are calculated at each and every unit of hidden layer. The output of each hidden layer unit will be calculated by processing the respective activation value through activation function. The outputs of hidden layer neurons is given as inputs to the output layer and again weighted sum or activation values will be calculated at each unit in the output layer which are processed through activation function to find the output. The outputs are compared with the target values and the squared error will be calculated. To minimize the error first the weights connects output and hidden layers will be modified and next the weights connects hidden and output layers will be modified. So while modifying the error the movement is backward so the algorithm is called as BP algorithm.

The architecture of BPNN and the proposed BPNN is as shown in Figure.4.12. The general architecture has input layer with n neurons, hidden layer with p neurons and output layer consists of m number of neurons. The weights between input layer(V_{ij}) and the weights between hidden and output layer (W_{jk}) which are initialized randomly.

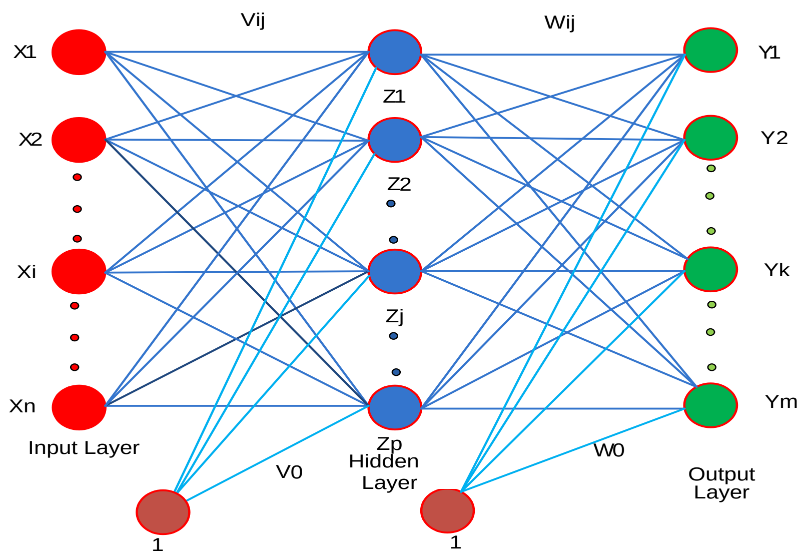


Figure. Architectures of Back propagation Neural Network

The input layer neurons are applied with input vector and the weighted sum and the output at all the neurons in the hidden layer can be calculated as given in Equations

$$Z_{-inj} = Voj + \sum_{i=1}^{n} (X_i \times V_{ij})$$

$$Z_j = f(Z_{-inj}) = \frac{1}{1+e^{Z_{-inj}}}$$

The outputs of hidden layer neurons will be the inputs to the neurons in the output layer. The weighted sum and the output at all the neurons in the output layer can be calculated as given in Equations 4.7 and 4.8.

$$Y_{-ink} = W_{ok} + \sum_{j=1}^{p} (Z_j * W_{jk})$$

$$Y_k = f(Y_{-ink}) = \frac{1}{1+e^{Y_{-ink}}}$$

The output at each neuron is compared with the target output and squared error is calculated. The weights will be varied until the error is minimized below a tolerance value. The weights between hidden layer and the output layer can be calculated as given in equations.

$$W_{jk'}(new)=W_{jk'}(old)+\Delta W_{jk'},$$

$$W_{ok'}(new)=W_{ok'}(old)+\Delta W_{ok}$$

Where $\Delta W_{jk}=\eta*\frac{\partial E}{\partial W_{jk}}=\eta*\delta_k*Z_j$

$$\Delta W_{ok}=\eta*\frac{\partial E}{\partial W_{ok}}=\eta*\delta_k$$

The weights connects the hidden and input layer, bias neuron will be updated as given in Equation.

$$V_{ij'}(new)=V_{ij'}(old)+\Delta V_{ij}$$

$$V_{oj'}(new)=V_{oj'}(old)+\Delta V_{oj}$$

Where

$$\Delta V_{ij}=\eta*\delta_j*X_i$$

$$\Delta V_{oj}=alpha*\delta_i \qquad (4.11)$$

4.6.1.1 Methodology

Step1: Initialize all weights to random values

Step2: Apply the deviation in speed as an input to the BPNN.

Step3: Calculate the outputs of the hidden and output layers in the forward pass. For the proposed Neural network the output is change in voltage

Step4: Calculate the squared error which is to be minimized for better training.

Step5: Update the weights between output and hidden, hidden, and input layers in the backward process.

Step6: Repeat the procedure till the squared error is blow 0.001.

4.6.1.2Drawbacks of BPNN

The drawbacks of the BPNN areas given as below.

i. It converges slowly if the learning rate is chosen small and if it chosen large the system may become unstable.

ii. These networks are suffering with local minima problem

iii. The network gives the worse generalization performance if the network is over trained.

iv. It is a gradient based training algorithm which takes longer time for training. Sometimes based on the complexity of the problem it takes days and weeks also.

step-7:-Calculate the o/p of o/p layer unit using

$$f(x) = \sum_{i=1}^{n} w_i \phi_i (\|x_i - c_i\|)$$

Step-8 :- Test stopping Condition.

stopping Condition may be minimization of error.

25/1/17 Recurrent (or) feedback neural network :-

In a feedforward n/w, signal is moving in forward direction from i/p to the output. The following are the different demerits of multi-layer feed forward networks.

1. Feed forward networks are unconditionally stable means may (or) may not stable.
2. Once the network is strain, the state of the network is fixed.
3. All feed forward networks does not have memory.
4. All the above demerits can be overcome by overcurrent (or) feedback network.
5. In feedback network, from the output signal is fed back to the input. so the network acts as a memory cell.

These feedback networks mainly used to the problem where output not only depends upon Current o/p but also on the previous inputs.

Ex:- Stock market price.
- weather forecasting

Feedback networks can be trained by back propogation algorithm. JohnHop field has made important Contributions to both the theory and applications of recurrent system. As a result Some architectures are called with his name which are called

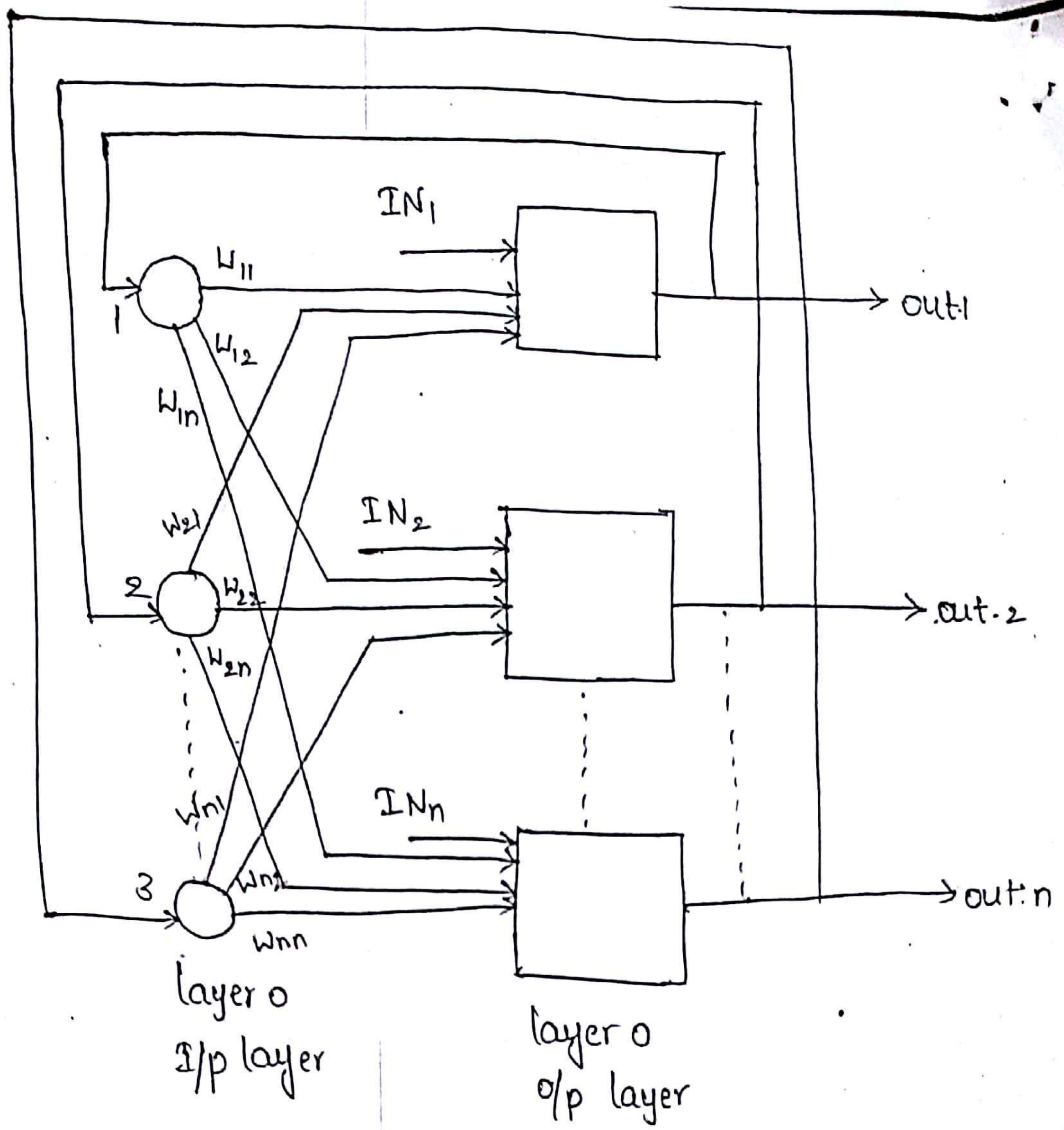


fig:- feedback network (or) recurrent n/w (or) hop field.

In the input layer no Commutations are performed, inputs are fed to the o/p layer. At each neuron in the o/p layer activation value (or) weighted sum will be calculated as follows.

$NET_1 = IN_1 + \underline{W_{11} * OUT_1 + W_{21} * OUT_2 + \cdots + W_{N1} , OUT_n}$

$NET_2 = IN_2 + \underline{W_{12} * OUT_1 + W_{22} * OUT_2 + \cdots + W_{N2} , OUT_n}$

⋮

$NET_j = IN_j + \underline{W_{1j} * OUT_1 + W_{2j} * OUT_2 + \cdots + W_{Nj} \cdot OUT_n}$

⋮

$NET_n = IN_n + \underbrace{W_{1n} * OUT_1 + W_{2n} * OUT_2 + \cdots + W_{Nn} * OUT_n}_{n}$

at all these activation

function, we will get all outputs of output Neurons. Based on activation function used, there is two types of recurrent network's.

Hopfield discrete recurrent network :-

If the activation function is discrete such as binary activation function the network is said to be hop field discrete recurrent n/w.

If NET value > 0 $\quad OUT_j = 1$

NET value ≤ 0 $\quad OUT_j = 0$

If the activation value is Continuous such as sigmoidal activatio function, the network will be Called as Hop field Continuous recurrent network.

$$\boxed{OUT_j = \frac{1}{1+e^{-NET_j}}}$$

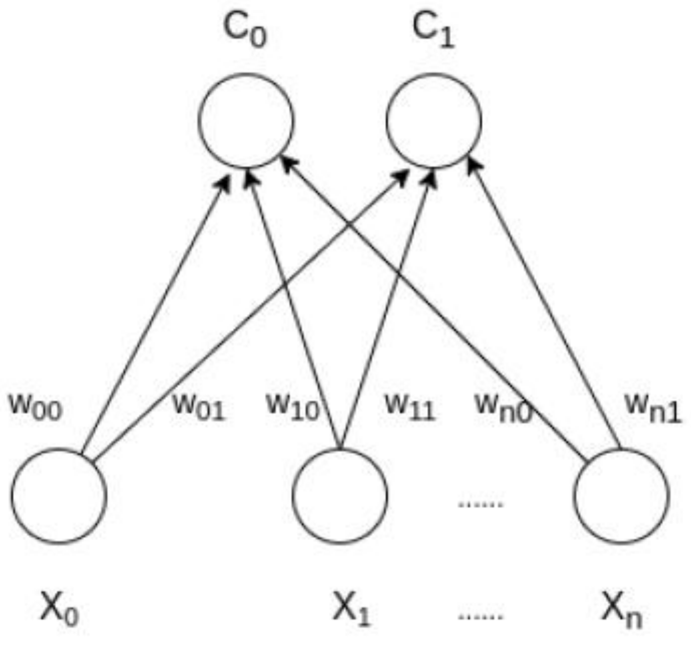
Defining the Stability of the network :-

For a stable network, successive iterations produce smaller and smaller o/p changes until eventually the o/p become Constant. Fo many networks the process never ends such networks are sai to be unstable systems.

Self Organizing Maps – Kohonen Maps

Self Organizing Map (or Kohonen Map or SOM) is a type of Artificial Neural Network which is also inspired by biological models of neural systems from the 1970s. It follows an unsupervised learning approach and trained its network through a competitive learning algorithm. SOM is used for clustering and mapping (or dimensionality reduction) techniques to map multidimensional data onto lower-dimensional which allows people to reduce complex problems for easy interpretation. SOM has two layers, one is the Input layer and the other one is the Output layer.

The architecture of the Self Organizing Map with two clusters and n input features of any sample is given below:



How do SOM works?

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. First, it initializes the weights of size (n, C) where C is the number of clusters. Then iterating over the input data, for each training example, it updates the winning vector (weight vector with the shortest distance (e.g Euclidean distance) from training example). Weight updation rule is given by :

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

where alpha is a learning rate at time t, j denotes the winning vector, i denotes the ith feature of training example and k denotes the kth training example from the input data. After training the SOM network, trained weights are used for clustering new examples. A new example falls in the cluster of winning vectors.

Algorithm

Training:

Step 1: Initialize the weights w_{ij} random value may be assumed. Initialize the learning rate α.

Step 2: Calculate squared Euclidean distance.

D(j) = Σ (wij – xi)^2   where i=1 to n and j=1 to m

Step 3: Find index J, when D(j) is minimum that will be considered as winning index.

Step 4: For each j within a specific neighborhood of j and for all i, calculate the new weight.

wij(new)=wij(old) + α[xi – wij(old)]

Step 5: Update the learning rule by using :

α(t+1) = 0.5 * t

Step 6: Test the Stopping Condition.

Learning rules:- 31/12/16

Learning is nothing but changing or updating the wei in such a way that error is minimized (or) the actual o/p must be equal to expected o/p (or) target o/p.

Change in weight (or) updation of weight is directly proportional to product of error and corresponding o/p.

$$\Delta W \propto \text{error} \times a$$

$$\Delta W = \eta \times \delta \times a$$

$$= \text{learning rate factor} \times (\text{expected o/p} - \text{Actual o/p}) \times \text{Input}$$

$$\boxed{\Delta W = \eta * (b - \overset{f(x)}{s}) * a}$$

Different scientist had given different rules for updation of the weights which are called as learning rules.

1. Hebb's Rule:-

In this rule weight updation (or) change in weight can be done using the following formula.

$$\Delta W_i = \eta * s * a$$

$$= \eta * f(x) * a$$

$$= \eta * f(W_i^T a) * a$$

$$\boxed{\Delta W_{ij} = \eta * f(W_i^T a) * a_j}$$

tial weights can be assumed as zero.

Perceptron learning rule :-

In this rule weight updation is given by formula, ~~$\Delta W_i =$~~

$$\Delta W_i = \eta * \text{error} * \text{Input}$$

$$= \eta * (\text{expected value} - \text{actual value}) * \text{Input}$$

$$= \eta * (b - \overset{\curvearrowright f(x)}{s}) * a$$

$$\boxed{\Delta W_i = \eta * (b - \text{sgn}(w^T a)) * a}$$

→ It is a supervisory learning rule.

→ Initial weights can be randomly choosen.

→ Here sign activation function can be used.

3. Delta learning rule :-

This is the most widely used rule.

In this rule weight adjustment is given by

$$\Delta W_i = \eta * (b - f(w^T a)) * a * f'(w^T a)$$

$[\because s = f(x)$:

→ It is a supervisory learning rule.

→ Initial weights can be randomly choosen.

→ Here sigmoidal activation function can be used.

4. Widrow and Hoff learning rule :-

In this rule weight updation is given by formula..

$$\Delta W_i = \eta \times (b - x) * a$$

$$= \eta * (b - (w^T a)) * a$$

→ It is a supervisory learning ru

networks: Back propaga Radial

Comparision Table :-

S.no	learning rule	weight change	learning process	Initializati.
1.	Hebb's	$\Delta W_i = \eta * \overset{\nearrow f(x)}{S} * a$ $\Delta W_i = \eta * f(W^T a) * a$	unsupervisory	zero
2.	Perceptron	$\Delta W_i = \eta * (b - \overset{\nearrow sgn}{\underset{sgn}{\cancel{f}}}(W^T a)) * a$	Supervisory	randomly
3.	Delta	$\Delta W_i = \eta * (b - f(W^T a)) * a * f'(W^T a)$	Supervisory	randomly
4.	Widrow Hoff	$\Delta W_i = \eta * (b - (W^T a)) * a$	Supervisory	randomly
5.	Correlation	$\Delta W_i = \eta * b * a$	Supervisory	zero
6.	Instar	$\Delta W_{ij} = \eta * (a - W_{ij})$	Unsupervisory	randomly
7.	Outstar	$\Delta W_{ij} = \eta * (b - W_{ij})$	supervisory	zero.

3/11/17 **Architecture of Artificial Neural Network :-**

There are mainly three acrchitectures of Artificial neural network as given below.

1. Single layer feed forward network.
2. Multi-layer feed forward network.
3. Recurrent network.