

MODULE-II (10 HOURS)

Neural networks: Single layer networks, Perceptrons: Adaline, Multilayer Perceptrons, Supervised Learning, Back-propagation, LM Method, Radial Basis Function Networks, Unsupervised Learning Neural Networks, Competitive Learning Networks, Kohonen Self-Organizing Networks, Learning Vector Quantization, Hebbian Learning. Recurrent neural networks, Adaptive neuro-fuzzy information systems (ANFIS), Hybrid Learning Algorithm, Applications to control and pattern recognition.

LECTURE-1

NEURAL NETWORK INTRODUCTION:

What is a neuron? A neuron is the basic processing unit in a neural network sitting on our brain. It consists of

1. Nucleus-
2. Axon- Output node
3. Dendrites-Input node
4. Synaptic junction

The dynamics of this synaptic junction is complex. We can see the signal inputs from the action of a neuron and through synaptic junction an output is actuated which is carried over through dendrites to another neuron. Here, these are the neurotransmitters. We learned from our experience that these synaptic junctions are either reinforced or in the sense they behave in such a way that the output of synaptic junction may excite a neuron or inhibit the neuron. This reinforcement of the synaptic weight is a concept that has been taken to artificial neural model.

The objective is to create artificial machine and this artificial neural networks are motivated by certain features that are observed in human brain, like as we said earlier, parallel distributed information processing.

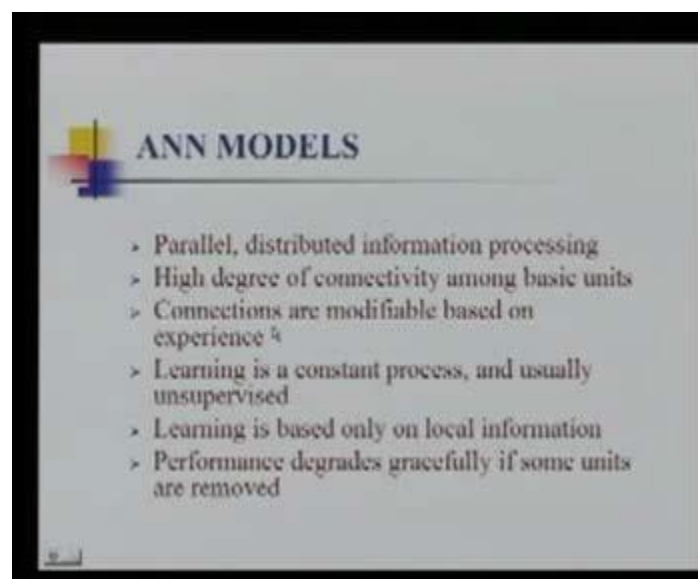


Fig. ANN model

Artificial neural networks are among the most powerful learning models. They have the versatility to approximate a wide range of complex functions representing multi-dimensional input-output maps. Neural networks also have inherent adaptability, and can perform robustly even in noisy environments.

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected simple processing elements (neurons) working in unison to solve specific problems. ANNs, like people, learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons. This is true of ANNs as well. ANNs can process information at a great speed owing to their highly massive parallelism.

A trained neural network can be thought of as an "expert" in the category of information it has been given to analyse. This expert can then be used to provide projections given new situations of interest and answer "what if" questions.

Advantages of ANN:

1. Adaptive learning: An ability to learn how to do tasks based on the data given for training or initial experience.
2. Self-Organisation: An ANN can create its own organisation or representation of the information it receives during learning time.
3. Real Time Operation: ANN computations may be carried out in parallel, and special hardware devices are being designed and manufactured which take advantage of this capability.
4. Fault Tolerance via Redundant Information Coding: Partial destruction of a network leads to the corresponding degradation of performance. However, some network capabilities may be retained even with major network damage.

Table- Difference between the brain and a digital Computer

Property	Computer	Brain
Shape	2d Sheets of inorganic matter	3d volume of organic matter
Power	Powered by DC mains	Powered by ATP
Signal	Digital	pulsed
Clock	Centralized clock	No centralized clock
Clock speed	Gigahertz	100s of Hz
Fault tolerance	Highly fault-sensitive	Very fault-tolerant
Performance	By programming	By learning

Differences human brain & ANN:

1. Computer has such fast speed of GHz, a traditional computer, however, when it comes to certain processing like pattern recognition and language understanding, the brain is very fast.
2. Intelligence and self-awareness, are absent in an artificial machine.

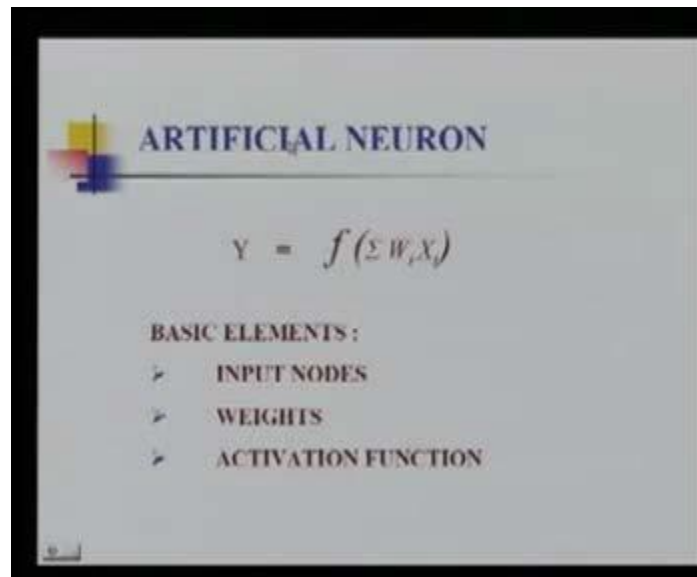


Fig. An artificial neuron

An Artificial Neuron:

Basic computational unit in an artificial neural network is neuron. Obviously, it has to be an artificial neuron.

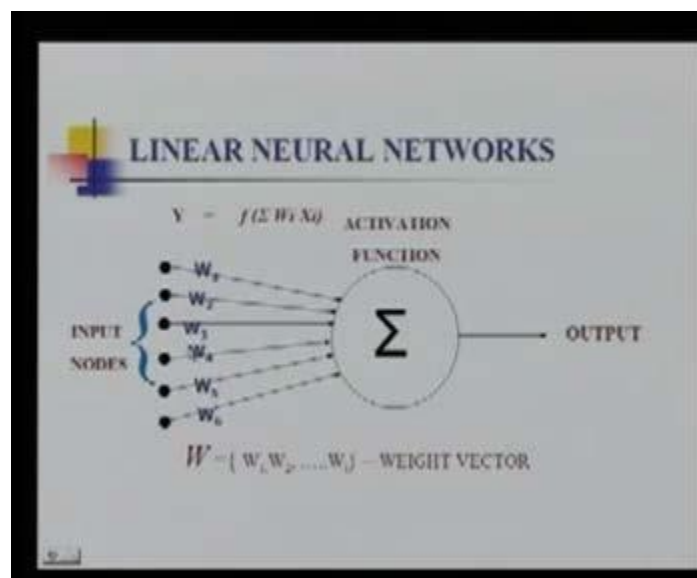


Fig. An artificial Neuron in linear NN

This artificial neuron has three basic elements:

1. Nodes,
2. Weights and
3. Activation function.

Between input nodes and output nodes, there are synaptic weights w_1, w_2, w_3, w_4, w_5 and w_6 . There can be as many weights and these weights are multiplied with the signal as they reach the output unit, where the output is simply sum of the signal multiplied with the weights and then this output goes to an activation function f .

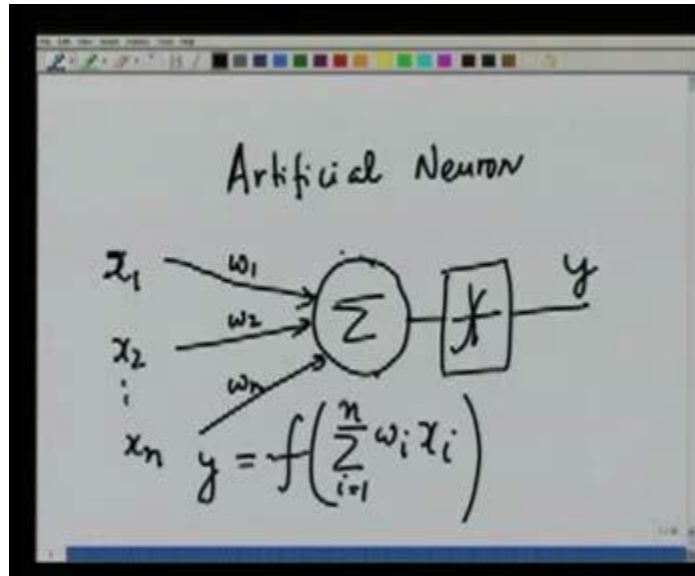


Fig. Basic processing unit- the neuron

In a simple neuron, if input signals be x_1, x_2, x_n with weights w_1, w_2 and w_n . The weighted sum will activate this total output by an activation function f . That is your output. What you are seeing is actually a nonlinear map from input vector x_i to output y . A single neuron has single output but multiple inputs. Inputs are multiple for a single neuron and the output is unique, y and this output y and the input bear a nonlinear relationship, by f . Neural networks can be built using this single neuron. We can use the single neuron and build neural networks.

Analogy to brain:

Artificial Neural Network (ANN) is a system which performs information processing. An ANN resembles or it can be considered as a generalization of mathematical model of human brain assuming that

1. Information processing occurs at many simple elements called neurons.
2. Signals are passed between neurons over connection links.
3. Each connection link has an associated weight, which in a typical neural net multiplies the signal transmitted.

ANN is built with basic units called *neurons* which greatly resemble the neurons of human brain. A neural net consists of a large number of simple processing elements called neurons. Each neuron applies an activation function to its net input to determine its output signal. Every neuron is connected to other neurons by means of directed communication links, each with an associated weight. Each neuron has an internal state called its *activation level*, which is a function of the inputs it has received. As and when the neuron receives the signal, it gets added up and when the cumulative signal reaches the activation level the neuron sends an output. Till then it keeps receiving the input. So activation level can be considered as a threshold value for us to understand.

In general, a neural network is characterized by

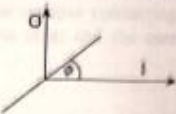
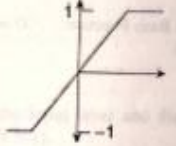
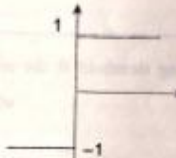
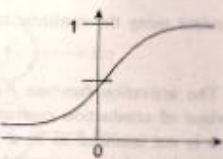
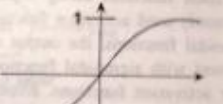
1. *Pattern of connections* between the neurons called its architecture
2. Method of *determining the weights* on the connections called its training or learning algorithm
3. Its *internal state* called its Activation function.

The arrangement of neurons into layers and the connection patterns within and between layers is called the net *architecture*. A neural net in which the signals flow from the input units to the output units in a forward direction is called *feed forward nets*.

Interconnected competitive net in which there are closed loop signal paths from a unit back to it is called a *recurrent network*. In addition to architecture, the method of setting the values of the weights called *training* is an important characteristic of neural nets. Based on the training methodology used neural nets can be distinguished into *supervised* or *unsupervised* neural nets. For a neural net with supervised training, the training is accomplished by presenting a sequence of training vectors or patterns each with an associated target output vector. The weights are then adjusted according to a learning algorithm. For neural nets with unsupervised training, a sequence of input vectors is provided, but no target vectors are specified. The net modifies the weights so that the most similar input vectors are assigned to the same output unit. The neural net will produce a representative vector for each cluster formed. Unsupervised learning is also used for other tasks, in addition to clustering.

LECTURE-2

Activation functions:

Type	Equation	Functional form
Linear	$O = gI$ $g = \tan \phi$	
Piecewise Linear	$O = \begin{cases} 1 & \text{if } mI > 1 \\ gI & \text{if } mI < 1 \\ -1 & \text{if } mI < -1 \end{cases}$	
Hard Limiter	$O = \text{sgn } [I]$	
Unipolar Sigmoidal	$O = \frac{1}{(1 + \exp(-\lambda I))}$	
Bipolar Sigmoidal	$O = \tanh [\lambda I]$	

Type	Equation	Functional form
Unipolar Multimodal	$O = \frac{1}{2} \left[1 + \frac{1}{M} \sum_{m=1}^M \tanh (g^m (I - W_{O^m}^*)) \right]$	
Radial Basis Function (RBF)	$O = \exp(I)$ $I = \left[\frac{-\sum_{i=1}^N (W_i(t) - X_i(t))^2}{2\sigma^2} \right]$	

Architecture:

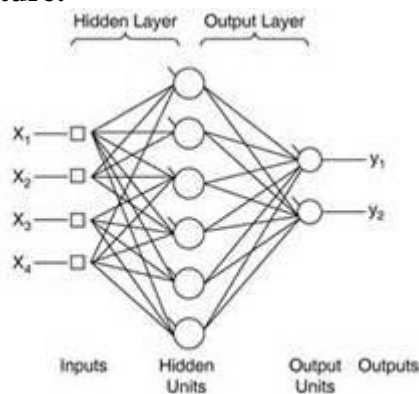


Fig. Architecture of multilayer Neural network

Artificial neural networks are represented by a set of nodes, often arranged in layers, and a set of weighted directed links connecting them. The nodes are equivalent to neurons, while the links denote synapses. The nodes are the information processing units and the links acts as communicating media.

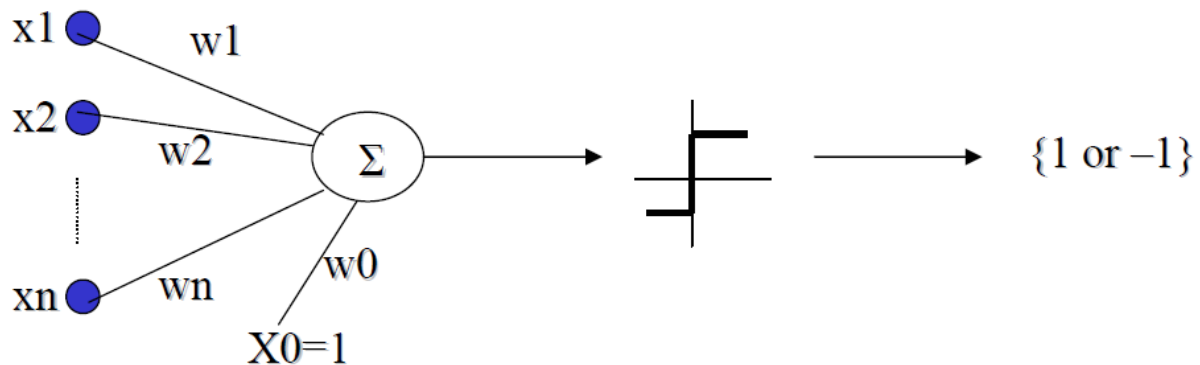
A neural network may have different layers of neurons like

1. input layer,
2. hidden layer,
3. output layer.

The input layer receives input data from the user and propagates a signal to the next layer called the hidden layer. While doing so it multiplies the weight along with the input signal. The hidden layer is a middle layer which lies between the input and the output layers. The hidden layer with non linear activation function increases the ability of the neural network to solve many problems than the case without the hidden layer. The output layer sends its calculated output to the user from which decision can be made. Neural nets can also be classified based on the above stated properties.

There are a wide variety of networks depending on the nature of information processing carried out at individual nodes, the topology of the links, and the algorithm for adaptation of link weights. Some of the popular among them include:

Perceptron: Definition: It's a step function based on a linear combination of real-valued inputs. If the combination is above a threshold it outputs a 1, otherwise it outputs a -1. This consists of a single neuron with multiple inputs and a single output. It has restricted information processing capability. The information processing is done through a transfer function which is either linear or non-linear.



$$O(x_1, x_2, \dots, x_n) = \begin{cases} 1 & \text{if } w_0 + w_1x_1 + w_2x_2 + \dots + w_nx_n > 0 \\ -1 & \text{otherwise} \end{cases}$$

Fig. A perceptron

A perceptron can learn only examples that are called “linearly separable”. These are examples that can be perfectly separated by a hyperplane.

Perceptrons can learn many boolean functions: AND, OR, NAND, NOR, but not XOR

However, every boolean function can be represented with a perceptron network that has two levels of depth or more.

The weights of a perceptron implementing the AND function is shown below.

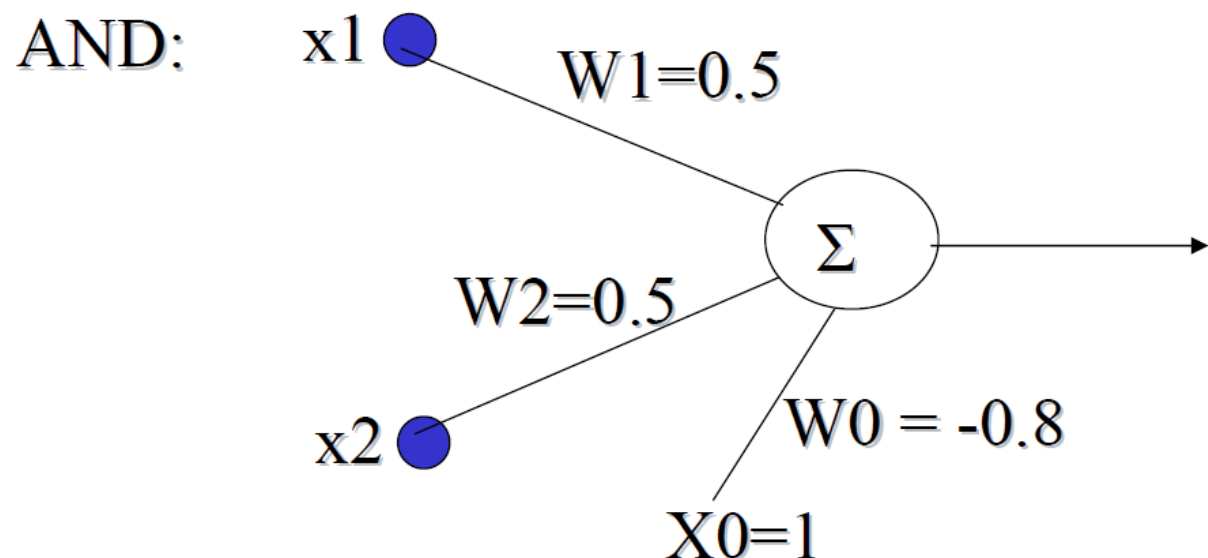


Fig. AND operation on inputs by a single perceptron

Multi-layered Perceptron (MLP): It has a layered architecture consisting of input, hidden and output layers. Each layer consists of a number of perceptrons. The output of each layer is

transmitted to the input of nodes in other layers through weighted links. Usually, this transmission is done only to nodes of the next layer, leading to what are known as feed forward networks. MLPs were proposed to extend the limited information processing capabilities of simple perceptrons, and are highly versatile in terms of their approximation ability. Training or weight adaptation is done in MLPs using supervised backpropagation learning.

Adding a hidden layer:

The perceptron, which has no hidden layers, can classify only linearly separable patterns.

The MLP, with at least 1 hidden layer can classify *any* linearly non-separable classes also.

An MLP can approximate any continuous multivariate function to any degree of accuracy, provided there are sufficiently many hidden neurons (Cybenko, 1988; Hornik et al, 1989). A more precise formulation is given below.

A serious limitation disappears suddenly by adding a single hidden layer.

It can easily be shown that the XOR problem which was not solvable by a Perceptron can be solved by a MLP with a single hidden layer containing two neurons.

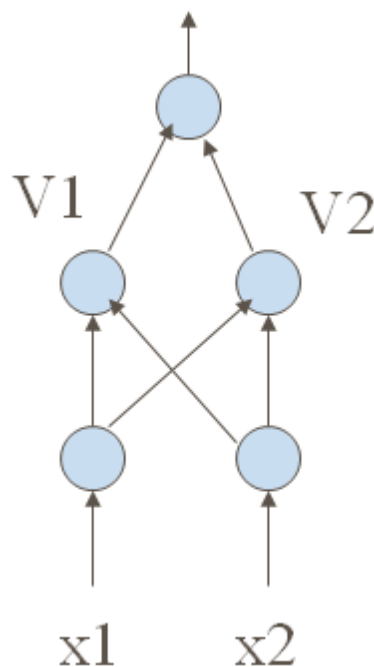


Figure 6.2.1.1: MLP for solving Xor

Recurrent Neural Networks: RNN topology involves backward links from output to the input and hidden layers. The notion of time is encoded in the RNN information processing scheme. They are thus used in applications like speech processing where inputs are time sequences data.

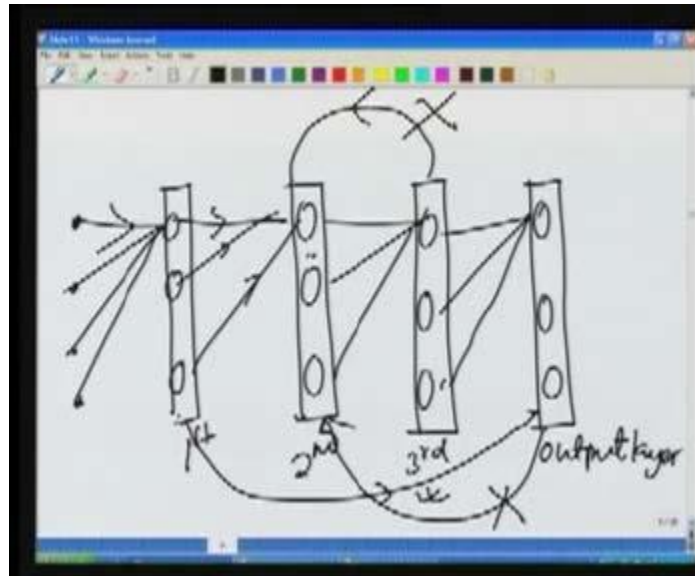


Fig. Multilayer feed back network (Recurrent Neural Network)

Self-Organizing Maps: SOMs or Kohonen networks have a grid topology, with unequal grid weights. The topology of the grid provides a low dimensional visualization of the data distribution. These are thus used in applications which typically involve organization and human browsing of a large volume of data. Learning is performed using a winner take all strategy in an unsupervised mode. It is described in detail later.

Single layer Network:

A neural net with only input layer and output layer is called single layer neural network. A neural network with input layer, one or more hidden layers and an output layer is called a multilayer neural network. A single layer network has limited capabilities when compared to the multilayer neural networks.

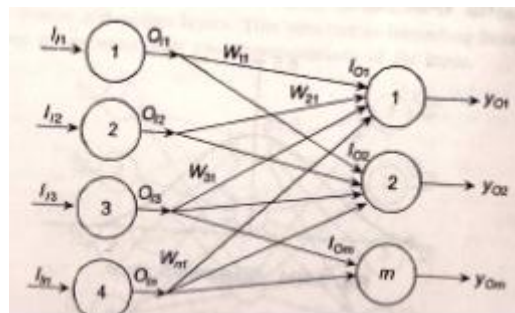


Fig. Single Layer feed forward Neural Network

LECTURE-3

Steps in developing NN:

1.1. Network formation

Neural network consists of an input layer, an output layer and a hidden layer. While a neural network is constructed, the number of neurons in each layer has to be fixed. The input layer will have neurons whose number will be equal to the number of features extracted. The number of neurons in the output layer will be equal to the number of pattern classes. The

number of neurons in the hidden layer is decided by trial and error basis. With a minimum number of neurons in the hidden layer, the neural network will be constructed and the convergence will be checked for. Then the error will be noted. The number of neurons for which the error is minimum, can be taken and will be checked for reduced error criterion.

1.2. Data preprocessing and normalization

Data selection and pre processing can be a demanding and intricate task. Neural net is as good as the input data used to train it. If important data inputs are missing, then the effect on the neural network's performance can be significant. The most appropriate raw input data must be preprocessed. Otherwise the neural network will not produce accurate results. Transformation and normalization are two widely used preprocessing methods. Transformation involves manipulating raw data inputs to create a single input to a net, while normalization is a transformation performed on a single data input to distribute the data evenly and scale it into an acceptable range for the network. Knowledge of the domain is important in choosing preprocessing methods to highlight the features in the data, which can increase the ability of the network to learn the association between inputs and outputs. Data normalization is the final preprocessing step. In normalizing data, the goal is to ensure that the statistical distribution of values should be scaled to match the range of the input neurons. The simplest method of normalization can be done using the formula

$X_{\text{normalized}} = (X - \mu) / \sigma$ where μ and σ are the mean and standard deviation of the input data.

Perceptron Learning

Learning a perceptron means finding the right values for W . The hypothesis space of a perceptron is the space of all weight vectors.

The perceptron learning algorithm can be stated as below.

1. Assign random values to the weight vector
2. Apply the *weight update rule* to every training example
3. Are all training examples correctly classified?
 - a. Yes. Quit
 - b. No. Go back to Step 2.

There are two popular weight update rules.

- i) The perceptron rule, and
- ii) Delta rule

The Perceptron Rule

For a new training example $X = (x_1, x_2, \dots, x_n)$, update each weight according to this rule:

$$w_i = w_i + \Delta w_i$$

Where $\Delta w_i = \eta (t - o) x_i$

t: target output

o: output generated by the perceptron

η : constant called the learning rate (e.g., 0.1)

Comments about the perceptron training rule:

Example means training data.

- If the example is correctly classified the term $(t - o)$ equals zero, and no update on the weight is necessary.
- If the perceptron outputs -1 and the real answer is 1 , the weight is increased.
- If the perceptron outputs a 1 and the real answer is -1 , the weight is decreased.
- Provided the examples are linearly separable and a small value for η is used, the rule is proved to classify all training examples correctly (i.e, is consistent with the training data).

The Delta Rule

What happens if the examples are not linearly separable?

To address this situation we try to approximate the real concept using the delta rule.

The key idea is to use a *gradient descent search*. We will try to minimize the following error:

$$E = \frac{1}{2} \sum_i (t_i - o_i)^2$$

where the sum goes over all training examples. Here o_i is the inner product WX and not $\text{sgn}(WX)$ as with the perceptron rule. The idea is to find a minimum in the space of weights and the error function E .

The delta rule is as follows:

For a new training example $X = (x_1, x_2, \dots, x_n)$, update each weight according to this rule:

$$w_i = w_i + \Delta w_i$$

Where $\Delta w_i = -\eta E'(W)/w_i$

η : learning rate (e.g., 0.1)

It is easy to see that

$$E'(W)/w_i = \sum_i (t_i - o_i) (-x_i)$$

So that gives us the following equation:

$$w_i = \eta \sum_i (t_i - o_i) x_i$$

There are two differences between the perceptron and the delta rule. The perceptron is based on an output from a step function, whereas the delta rule uses the linear combination of inputs directly. The perceptron is guaranteed to converge to a consistent hypothesis assuming the data is linearly separable. The delta rule converges in the limit but it does not need the condition of linearly separable data.

There are two main **difficulties with the gradient descent method**:

1. Convergence to a minimum may take a long time.

2. There is no guarantee we will find the global minimum.

These are handled by using momentum terms and random perturbations to the weight vectors.

LECTURE-4

ADALINE & MADALINE:

The Adaline networks (ADaptive LINEar Element) and Madaline (Multiple Adaline) were developed by Widrow. The structures use neurons and step/ sigmoidal activation function. ADALINE has one output neuron but MADALINE has many. The learning is different from a perceptron. It is here by Widrow-Hoff or LMS (Least Mean Square error) rule. Analogical input or output can be found by this network as minimum error function is searched before applying activation function.

ADALINE:

The structure includes an adaptive linear combiner (ALC) to obtain linear response that can be applied to other elements of bipolar commutation. If O/P of ALC is +ve response of ADALINE is +1, and if -ve result of ADALINE is -1. It is represented by: