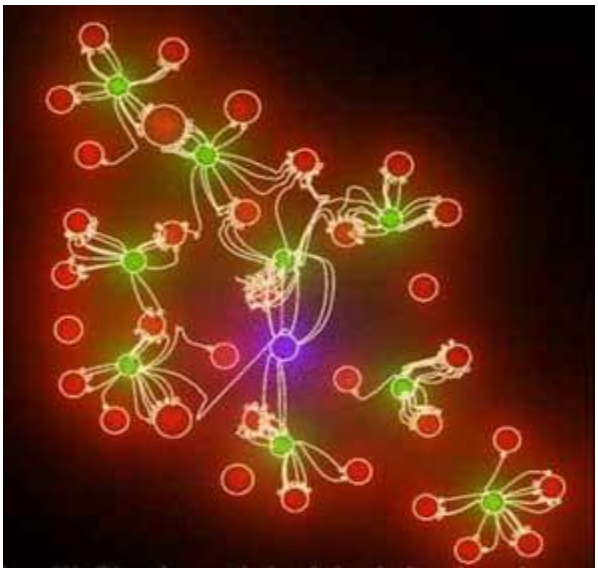
Basics of Artificial Neural Networks

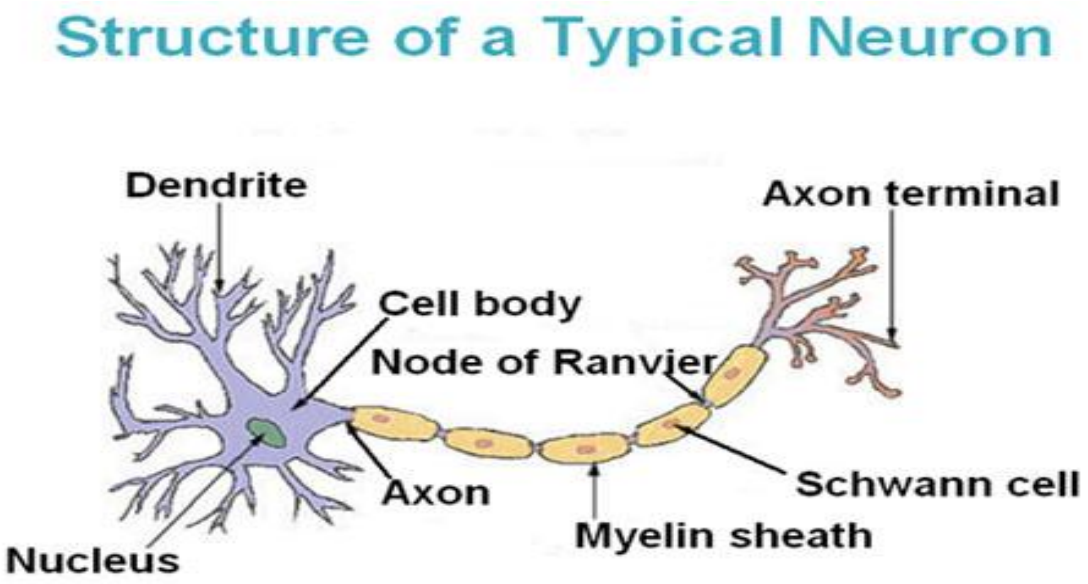


Animals are able to react adaptively to changes in their external and internal environment, and they use their nervous system to perform these behaviours.

An appropriate model/simulation of the nervous system should be able to produce similar responses and behaviours in artificial systems

Artificial Neural Networks is a function approximator where it transforms inputs into outputs to the best of its ability.

Neural Networks comprises of many neurons that co-operate to perform the desired function.

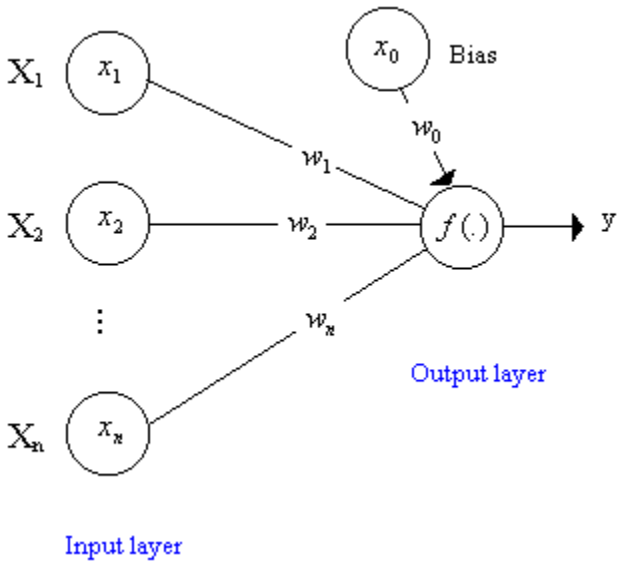


Historical Development of Neural Networks Principles:

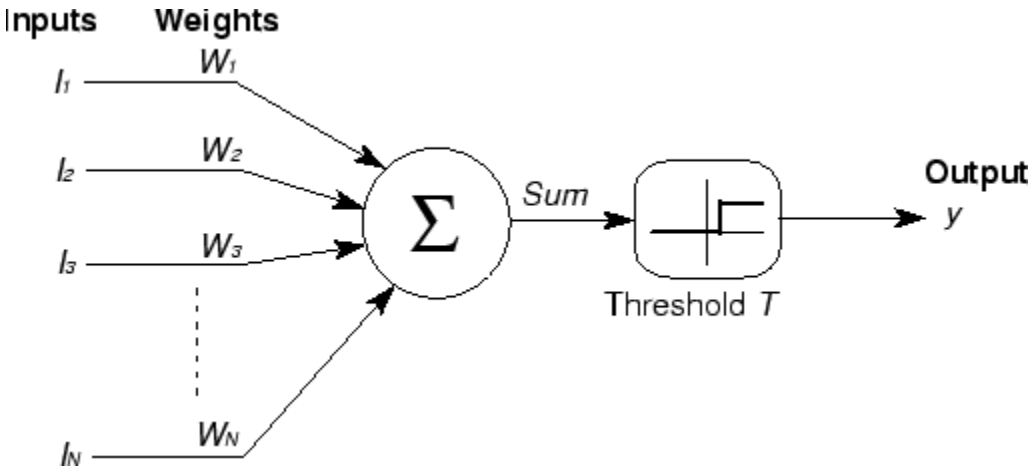
Neurons work by processing information. Artificial Neural Networks are used for

a) Classification
b) Noise reduction
c) Prediction
d) Ability to learn
e) Ability to generalize

The output of the neuron is a function of the weighted sum of the inputs plus a bias.



McCulloch-Pitts neuron model performs a weighted sum of the inputs to the element followed by a threshold logic operation.



Combinations of these computing elements were used to realize several logic computations. The main drawback of this model of computation is that the weights are fixed and hence the model could not learn from examples.

Hebb's law became a fundamental learning rule in neural networks literature which adjusts a connection weight based on pre and post synaptic values of the variables.

A learning machine was developed by Marvin Minsky in which the connection strengths could be adapted automatically.

Rosenblatt proposed the perceptron model which has weights adjustable by the perceptron learning law. The learning law was shown to converge for pattern classification problems which are linearly separable in the feature space. A single layer of perceptron could handle only linearly separable classes while multilayer perceptron could be used to perform any pattern classification task.

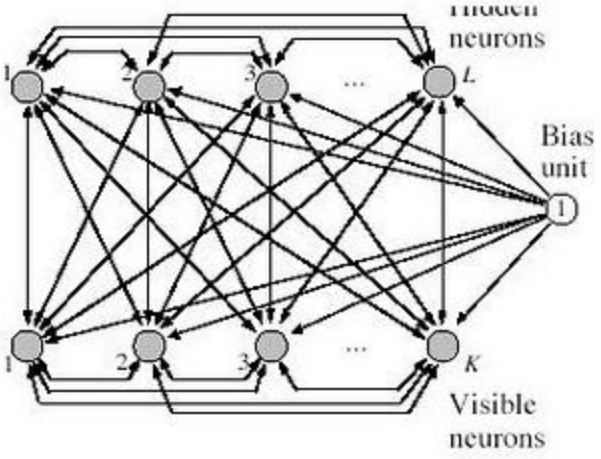
Widrow and his group proposed an Adaline model for a computing element and an LMS learning algorithm to adjust the weights of an Adaline model. The algorithm was successfully used for adaptive signal processing situations.

The resurgence of interest in artificial neural networks is due to two key developments.

- The first is the energy analysis of feedback neural networks by John Hopfield. The analysis has shown the existence of stable equilibrium states in a feedback network provided that the network has symmetric weights and that the state update is made asynchronously.

Rumelhart has shown that it is possible to adjust the weights of a multilayer feed forward neural network in a systematic way to learn the implicit mapping in a set of input – output pattern pairs. The learning law is called generalized delta rule or error back propagation learning law.

Ackley, Hinton and Sejnowski proposed the Boltzmann machine which is a feedback neural network with stochastic neuron units. A stochastic neuron has an output function which is implemented using a probabilistic update rule instead of a deterministic update rule as in the Hopfield model.



The Boltzmann machine has several additional neuron units called hidden units which are used to make a given pattern storage problem representable in a feedback network.

Other key developments are the concepts of competitive learning, self organization and simulated annealing. Self organization led to the realization of feature mapping. Simulated annealing has been very useful in implementing the learning law for the Boltzmann machine.

Artificial Neural Networks : Terminology

a) **Processing unit** – An artificial neural networking (ANN) is a highly simplified model of the structure of the biological neural network. An ANN consists of interconnected processing units. The general model of a processing unit consists of a summing part followed by an output part. The summing part receives N input values, weights each value and computes a weighted sum. The weighted sum is called the activation value. The output part produces a signal from the activation value.

b) **Interconnections** – Several processing units are interconnected according to some topology to accomplish a pattern recognition task. Therefore the inputs to a processing unit may come from the outputs of other processing units and/or from external sources. The output of each unit may be given to several units including itself. The amount of the output of one unit received by another unit depends on the strength of the connection between the units and it is reflected in the weight value associated with the connecting link. The set of N activation values of the network defines the activation state of the network at that instant. The set of N output values of the network define the output state of the network at that instant.

c) **Operations** – In operation, each unit of ANN receives inputs from other connected units and/or from an external source. A weighted sum of inputs is computed at a given instant of time. The activation value determines the actual output from the output function unit i.e. the output state of the unit. The output values and other external inputs determine the activation and output states of the other units. Activation dynamics determines the

activation values of all the units i.e. the activation state of the network as a function of time. Activation dynamics also determines the dynamics of the output state of the network.

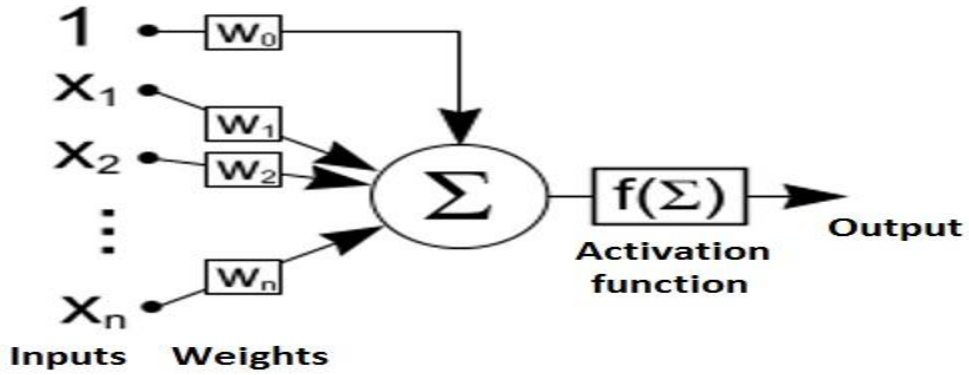
Set of all activation states defines the **activation state space** of the network. Set of all output states defines the **output state space** of the network. For a given network, defined by the units and their interconnections with appropriate units, the activation state determines the **short term memory function** of the network.

Given an external input, the activation dynamics is followed to recall a pattern stored in a network. The set of all weights on all connections in a network form a weight vector. The set of all possible weight vectors define the **weight space.** Synaptic dynamics is followed to adjust the weights in order to store the given patterns in the network. The process of adjusting the weights is referred to as **learning.** Once the learning process is completed, the final set of weight values corresponds to the **long term memory function** of the network. The procedure to incrementally update each of the weights is called a learning law or learning algorithm.

d) **Update** – The updating of the output states of all the units in activation and synaptic dynamics could be performed synchronously. In this case, the activation values of all the units are computed at the same time, assuming a given output state throughout. From the activation values, the new output state of the network is derived. In an asynchronous update, each unit is updated sequentially taking the current output state of the network into account. For each unit, the output state can be determined from the activation value either deterministically or stochastically.

Models of Neuron - 3 classical models

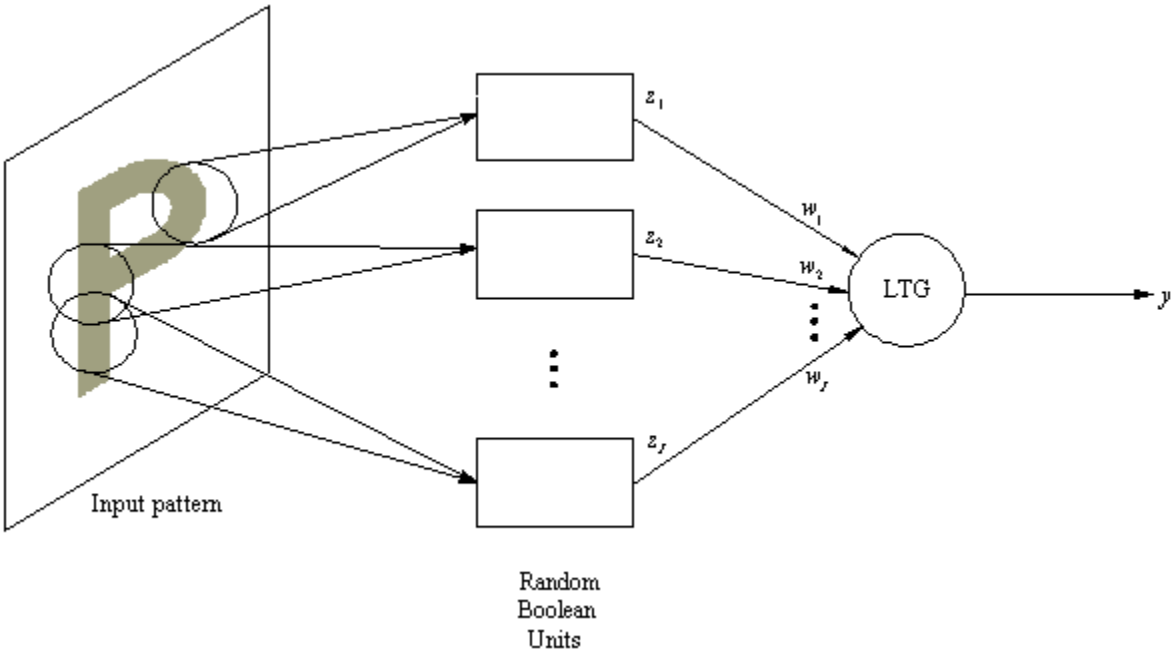
a) **McCulloch – Pitts model**: In McCulloch Pitts model, the activation(x) is given by a weighted sum of its M input values (a_i) and a bias term Θ. The output signal(s) is a non linear function f(x) of the activation value x.



			M
Activation	x	=	$\Sigma\, w_i\, a_i - \Theta$
			i=1
Output signal	s	=	f(x)

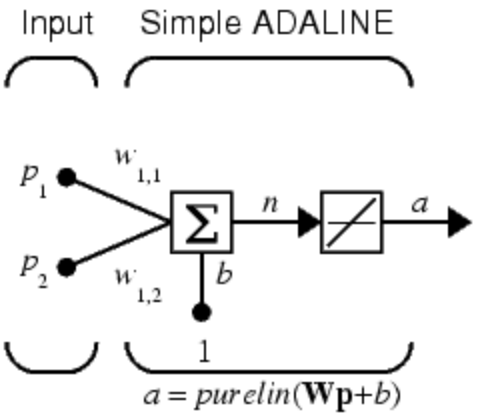
In the MP model, the weights are fixed. Hence a network using this model does not have the capability of learning.

b) **Perceptron**: The Rosenblatt's perceptron model 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 (adjustment of weight) is incorporated in the operation of the unit. The desired or target output (b) is compared with the actual binary output(s) and the error (δ) is used to adjust the weights.



			M
Activation	x	=	$\Sigma\, w_i\, a_i - \Theta$
			i=1
Output signal	s	=	f(x)
Error	δ	=	b-s
Weight change	Δw_i	=	$\eta\delta a_i$ where η is the learning rate parameter.

c) **Adaline**: Adaptive Linear Element is a computing model proposed by Widrow. The main distinction between the Rosenblatt's perceptron model and the Widrow's adaline model is that in the Adaline, the analog activation value(x) is compared with the target output (b). The output is a linear function of the activation value(x).



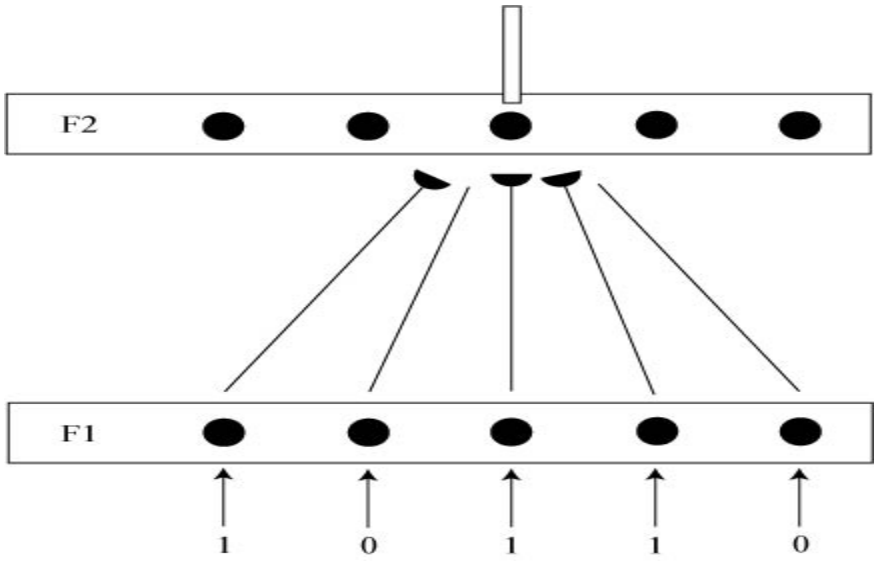
Activation	x	=	$\sum_{i=1}^{M} w_i a_i - \Theta$		
Output signal	s	=	f(x)	=	x
Error	δ	=	b-s	=	b-x
Weight change	Δw_i	=	$\eta \delta a_i$	where η is the learning parameter	

Topology -

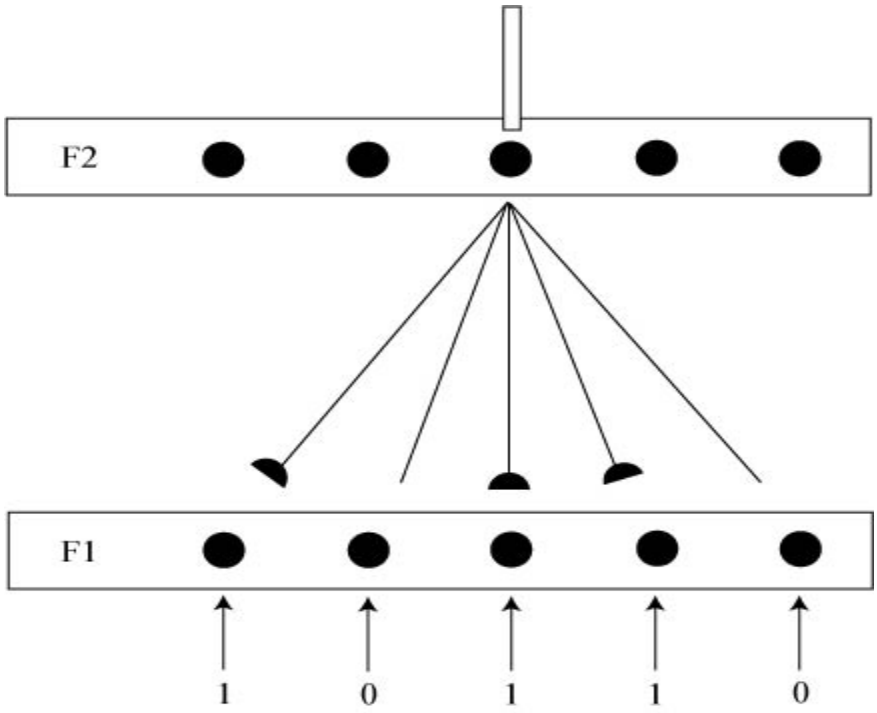
Artificial neural networks are useful only when the processing units are organized in a suitable manner to accomplish a given pattern recognition task. The arrangement of the processing units, connections and pattern input/output is referred to as topology. Artificial neural networks are normally organized into layers of processing units. The units of the layer have the same activation dynamics and output function. Connections can be made either from the units of one layer to units of another layer (interlayer connections) or among the units within the layer (intralayer connections) or both interlayer and intralayer connections. Connections across the layers can be organized either in a feed forward or feedback manner.

In a feedback network, the same processing unit may be visited more than once.

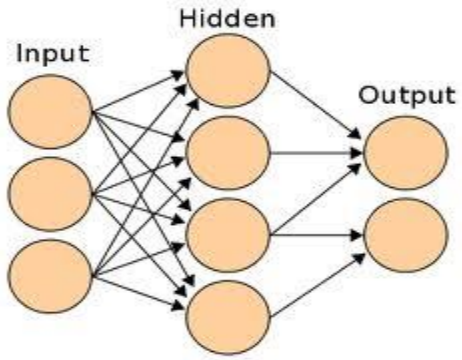
Instar – operation of an instar can be viewed as content addressing the memory.



Outstar – Operation of an outstar can be viewed as memory addressing the contents.

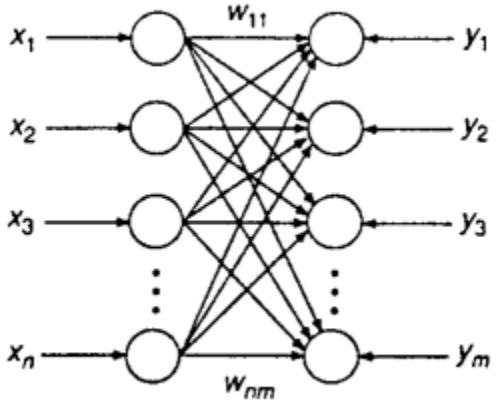


Group of instars – Hetero association network



Group of outstars – the opposite of group of instars

When the flow is bidirectional, we get a bidirectional associative memory where either of the layers can be used as input/output.



When the two layers coincide and the weights are symmetric then we obtain an auto associative memory in which each unit is connected to every other unit and to itself.

Basic learning laws –

The operation of a neural network is governed by neuronal dynamics. Neuronal dynamics consists of two parts: one corresponding to the dynamics of the activation state and the other corresponding to the dynamics of the synaptic state. The Short Term Memory (STM) is neural networks is modeled by the activation state of the network. The Long Term Memory (LTM) corresponds to the encoded pattern information in the synaptic weights due to learning.

Learning laws are merely implementation models of synaptic dynamics. A model of synaptic dynamics is described in terms of expressions for the first derivative of the weights. They are called learning equations.

Learning laws describe the weight vector for the i^{th} processing unit at time instant (t+1) in terms of weight vector at time instant (t) as follows –

$w_i(t+1) = w_i(t) + \Delta w_i(t)$ where $\Delta w_i(t)$ is the change in the weight vector.

1. **Hebb's law –**

   Change in the weight vector:

   $\Delta w_i = \eta f(w_i^T a)a$

   The j^{th} component of Δw_i is given by –

   $\Delta w_{ij} = \eta f(w_i^T a)a_j$

   $= \eta s_i a_j$

   where s_i is the output signal of the i^{th} unit. The law states that the weight increment is proportional to the product of the input data and the resulting output signal of the unit. This law represents unsupervised learning.

2. **Perceptron learning law –**

   Change in the weight vector:

   $\Delta w_i = \eta[b_i - sgn(w_i^T a)a]$ where sgn(x) is sign of x.

   The j^{th} component of Δw_i is given by –

   $\Delta w_{ij} = \eta[b_i - sgn(w_i^T a)]a_j$

   $= \eta(b_i - s_i)a_j$

   This law is applicable only for bipolar output functions. This is also called discrete perceptron learning law. This is supervised learning law as the law requires a desired output for each input.

3. **Delta learning law –**

   Change in the weight vector:

   $\Delta w_i = \eta[b_i - f(w_i^T a)]\, f(w_i^T a)a$ where f(x) is the derivative w.r.t x.

   The j^{th} component of Δw_i is given by –

   $\Delta w_{ij} = \eta[b_i - f(w_i^T a)]\, f(w_i^T a)a_j$

$= \eta[b_i - s_i] f(x_i a) a_j$

This law is valid only for a differentiable output function as it depends on the derivative of the output function. It is a supervised learning law since the change in the weight is based on the error between the desired and actual output values for a given input. Delta learning law can also be called a continuous perceptron learning law.

4. Widrow and Hoff LMS learning law –

Change in the weight vector:

$\Delta w_i = \eta[b_i - w_i^T a] a$ where f(x) is the derivative w.r.t x.

The j^{th} component of Δw_{ij} is given by

$\Delta w_{ij} = \eta[b_i - w_i^T a] a_j$

This is a supervised learning law and is a special case of delta learning law where the output function is assumed linear. Here, the change in the weight is made proportional to the negative gradient of the error between the desired output and the continuous activation value which is also the continuous output signal due to linearity of the output function. Hence this is also called the Least Mean Squared error learning law.

5. Correlation learning law –

Change in the weight vector:

$\Delta w_i = \eta b_i a$

The j^{th} component of Δw_{ij} is given by

$\Delta w_{ij} = \eta b_i a_j$

This is a special case of the Hebbian learning with the output signal (s_i) being replaced by the desired signal (b_i). Hebbian learning is unsupervised learning whereas the correlation learning is supervised learning since it uses the desired output value to adjust the weights.

6. Instar learning law –

This is relevant for a collection of neurons organized in a layer. All the inputs are connected to each of the units in the output layer in a feed forward manner. For a given input vector a , the output from each unit i is computed using the weighted sum $w_i^T a$.

The unit k that gives maximum output is identified i.e

$$W_k^T a = \max_i (w_i^T) a$$

Weight vector leading to k^{th} unit is given by –

$$\Delta w_k = \eta(a - w_k)$$

Therefore,

$$\Delta w_{kj} = \eta(a_j - w_{kj})$$

The final weight vector tends to represent a group of input vectors within a small neighborhood. This is a case of unsupervised learning.

7. Outstar learning law –

The outstar learning law is related to a group of units arranged in a layer. In this law, the weights are adjusted sp as to capture the desired output pattern characteristics. The adjustment of weights is given by –

$$\Delta w_{jk} = \eta(b_j - w_{jk})$$

where the k^{th} unit is the only active unit in the input layer. The vector b is the desired response from the layer of M units. The outstar learning law is a supervised learning law and it is used with a network of instars to capture the characteristics of the input and output patterns for data compression.

Functional Units of ANN for Pattern Recognition Tasks

Pattern Recognition Problem:

Functional units form building blocks for developing neural architectures to solve complex pattern recognition problems. In any pattern recognition task, we have a set of input patterns and the corresponding output patterns. Depending on the nature of the output patterns and the nature of the task environment, the problem could be identified as one of association or classification or mapping. The given set of input - output pattern pairs form only a few samples of an unknown system. From these samples, the pattern recognition model should capture the characteristics of the system.

Pattern Association Problem –

Given a set of input – output pattern pairs$(a1,b1),(a2,b2),(a3,b3)\ldots,(a_l,b_l)\ldots,(a_L,b_L)$ where

$a_l=(a_{l1},a_{l2,\ldots,}a_{lM})$ and $b_l=(b_{l1},b_{l2\ldots},b_{lN})$ are M and N dimensional vectors respectively, design a neural network to associate each input pattern with corresponding output pattern.

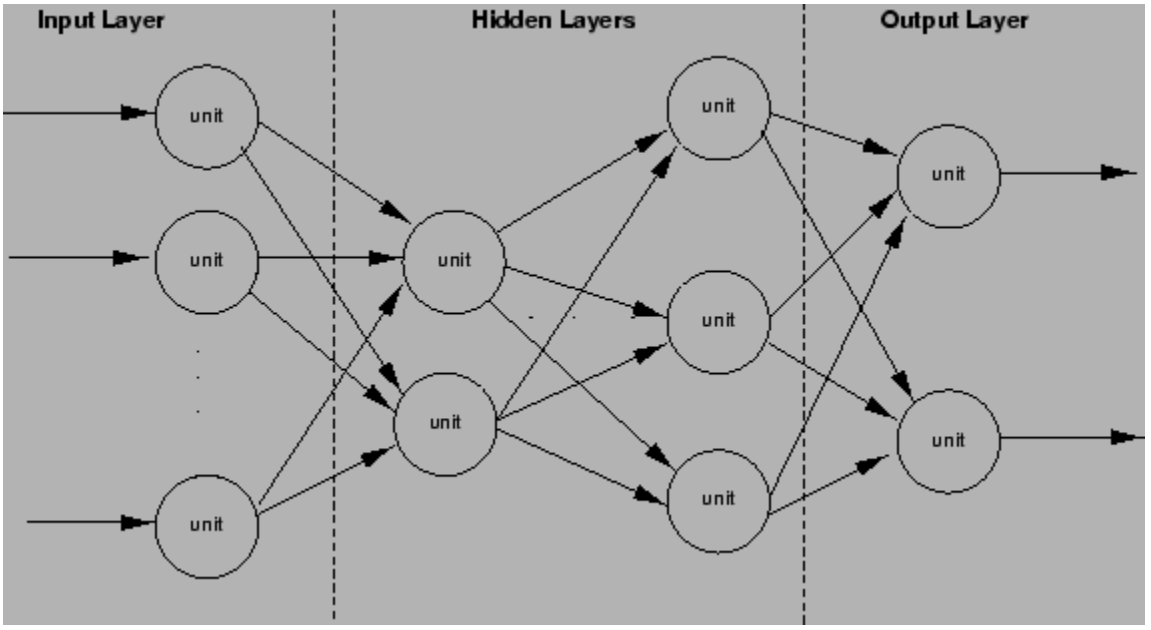
If a_l and b_l are distinct, then the problem is called hetero association. If $b_l=a_l$ then the problem is called auto association. The problem of storing the association of the input – output pattern pairs involves determining the weights of a network to accomplish the task (training part). Once stored, the problem of recall involves determining the output pattern for a given input pattern by applying the operations of the network on the input pattern.

The recalled output pattern depends on the nature of the input and the design of the network. If the input pattern is same as the one used in training, then the recalled output pattern is the same as the associated pattern in the training. If the input pattern is a noisy version of the trained input pattern, then the pattern may not be identical to any of the patterns used in training the network.

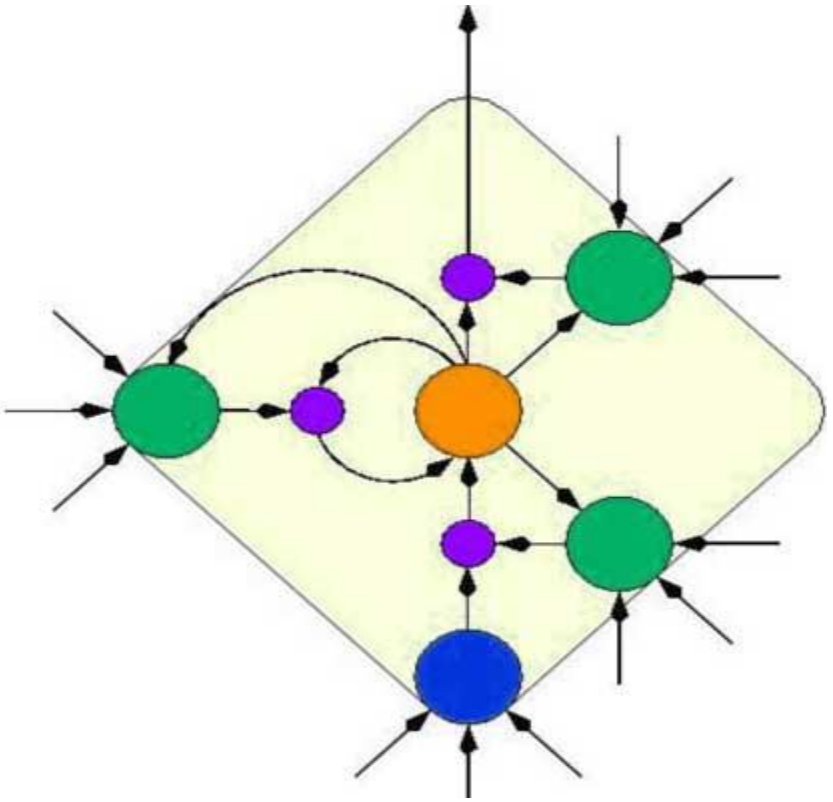
Basic functional units:

There are three types of artificial neural networks. The simplest networks of each of these types form the basic functional unit. They are functional because they can perform by themselves some simple pattern recognition tasks. They are basic because they form building blocks for developing neural network architectures for complex pattern recognition tasks. They are:

1. **Feed forward** – Simplest feed forward network is a two layer network with M input units and N output units. Each input unit is connected to each of the output units and each connection is associated with a weight or strength of the connection. Input units are linear and perform fan out task. Output units are linear or non –linear depending on the task that the network should perform. Feed forward networks are used for pattern association or pattern classification or pattern mapping.



2. **Feedback** – Simplest feedback network consists of a set of N processing units each connected to all other units. The connection strengths or weights are assumed to be symmetric. Depending on the task, units of the networks could be linear or non –linear. Feedback networks are used for auto association or pattern storage.



3. **Combination of feed forward and feed back** – The simplest combination network is called a competitive learning network. It consists of an input layer of units feeding to the units in the output layer in a feed forward manner and a feedback connection among the units in the output layer including self – feed back. The connection strengths or weights of the feed forward path are adjustable by training the network for a given pattern recognition task. The feedback connection strengths or weights in the output layer are fixed to specific values depending on the problem. The input units are all linear and perform fan – out task. The output units are either linear or non linear depending on the task the network should perform. The competitive learning network is used for pattern grouping/clustering.

Pattern Recognition Tasks by Functional Units –

The input pattern space R^M is an M-dimensional space and the input patterns are points in this space. The output pattern space R^N is an N – dimensional space and the output patterns are points in this space.

A. Pattern recognition tasks by feed forward neural networks –

1. **Pattern Association Problem**: The objective of designing a neural network is to capture the association between input – output pattern pairs in the given set of training data so that when any of the inputs is given, corresponding output is retrieved.
2. **Pattern Classification Problem**: If a group of input patterns correspond to the same output pattern, then there will be far fewer output patterns compared to number of input patterns. If some of the output patterns in the pattern association problem are identical, then the number of distinct output patterns can be viewed as class labels and the input patterns corresponding to each class can be viewed as samples of that class. Whenever a pattern belonging to a class is given as input, the network identifies the class label. During training, only a few samples of patterns for each class are given. In testing, the input patterns are usually different from the patterns used in the training set for the class. Network displays an accretive behavior. The performance of a network for the pattern classification problem depends on the characteristics of the samples associated with each class.
3. **Pattern mapping**: Given a set of input – output pattern pairs as in the pattern association problem, if the objective is to capture the implied mapping, then problem becomes a pattern mapping problem. Both the input and the output patterns are only samples from the mapping system. Once the system behavior is captured by the network, the network would produce a possible output pattern for a new input pattern not used in the training set. The possible output pattern would be approximately interpolated version of the output patterns corresponding to the input training patterns close to the given test input pattern. Network displays an interpolative behavior.

B. Pattern recognition tasks by feedback neural networks –

1. **Auto association Problem**: If each of the output patterns b_l in a pattern association problem is identical to the corresponding input patterns a_l then the output pattern space is identical to input pattern space. Then the problem becomes auto association problem. When a noisy input pattern is given, network retrieves the same noisy pattern.
2. **Pattern storage Problem**: In the auto association problem, if a given input pattern is stored in a network for later recall by an approximate input pattern, then the problem becomes a pattern storage problem. Due to its accretive behavior, pattern storage network is very useful.
3. **Pattern environment storage problem**: If a set of patterns together with their probabilities of occurrence are specified, then the resulting specification is called pattern environment. The design of a network to store a given pattern environment aims at recall of the stored patterns with the lowest probability of error. This is called pattern environment storage problem.

C. Pattern recognition tasks by competitive learning neural networks –

1. **Temporary pattern storage**: If a given input pattern is stored in a network in such a way that the pattern remains only until a new input pattern is given, then the problem becomes that of a short term memory or temporary storage problem.
2. **Pattern clustering problem**: Given a set of patterns, if they are grouped according to similarity of the patterns, then the resulting problem is called pattern clustering. There are two types of problems –
    a) Accretive behavior
    b) Interpolative behavior

    Pattern clustering leads to the problem of vector quantization.

3. **Feature mapping problem**: If similarities of features of input patterns have to be retained in the output, then the problem becomes a feature mapping problem. In this, a given set of input patterns are mapped onto output patterns in such a way that proximity of the output patterns reflects the similarity of the features of the corresponding input patterns.