C • H • A • P • T • E • R

2

Artificial Neural Network: An Introduction

Learning Objectives

- The fundamentals of artificial neural network.
- The evolution of neural networks.
- Comparison between biological neuron and artificial neuron.
- Basic models of artificial neural networks.
- The different types of connections of neural networks, learning and activation functions are included.
- Various terminologies and notations used throughout the text.
- The basic fundamental neuron model–McCulloch–Pitts neuron and Hebb network.
- The concept of linear separability to form decision boundary regions.

2.1 Fundamental Concept

Neural networks are those information processing systems, which are constructed and implemented to model the human brain. The main objective of the neural network research is to develop a computational device for modeling the brain to perform various computational tasks at a faster rate than the traditional systems. Artificial neural networks perform various tasks such as pattern-matching and classification, optimization function, approximation, vector quantization, and data clustering. These tasks are very difficult for traditional computers, which are faster in algorithmic computational tasks and precise arithmetic operations. Therefore, for implementation of artificial neural networks, high-speed digital computers are used, which makes the simulation of neural processes feasible.

2.1.1 Artificial Neural Network

As already stated in Chapter 1, an artificial neural network (ANN) is an efficient information processing system which resembles in characteristics with a biological neural network. ANNs possess large number of highly interconnected processing elements called *nodes* or *units* or *neurons*, which usually operate in parallel and are configured in regular architectures. Each neuron is connected with the other by a connection link. Each connection link is associated with weights which contain information about the input signal. This information is used by the neuron net to solve a particular problem. ANNs' collective behavior is characterized by their ability to learn, recall and generalize training patterns or data similar to that of a human brain. They have the capability to model networks of original neurons as found in the brain. Thus, the ANN processing elements are called *neurons* or *artificial neurons*.

It should be noted that each neuron has an internal state of its own. This internal state is called the *activation* or *activity level* of neuron, which is the function of the inputs the neuron receives. The activation signal of a neuron is transmitted to other neurons. Remember, a neuron can send only one signal at a time, which can be transmitted to several other neurons.

To depict the basic operation of a neural net, consider a set of neurons, say X_1 and X_2, transmitting signals to another neuron, Y. Here X_1 and X_2 are input neurons, which transmit signals, and Y is the output neuron, which receives signals. Input neurons X_1 and X_2 are connected to the output neuron Y, over a weighted interconnection links (W_1 and W_2) as shown in Figure 2-1.

For the above simple neuron net architecture, the net input has to be calculated in the following way:

$$y_{in} = -x_1 w_1 + x_2 w_2$$

where x_1 and x_2 are the activations of the input neurons X_1 and X_2, i.e., the output of input signals. The output y of the output neuron Y can be obtained by applying activations over the net input, i.e., the function of the net input:

$$y = f(y_{in})$$

Output = Function (net input calculated)

The function to be applied over the net input is called *activation function*. There are various activation functions, which will be discussed in the forthcoming sections. The above calculation of the net input is similar to the calculation of output of a pure linear straight line equation ($y = mx$)

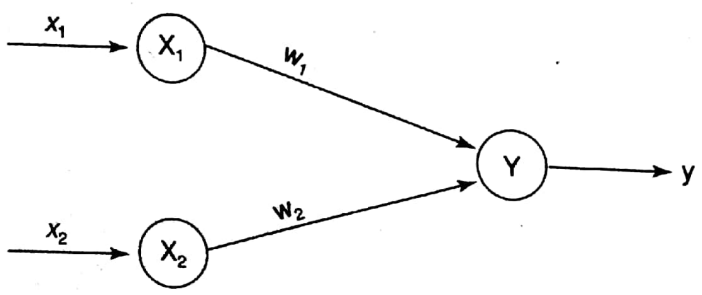

Figure 2-1 Architecture of a simple artificial neuron net.

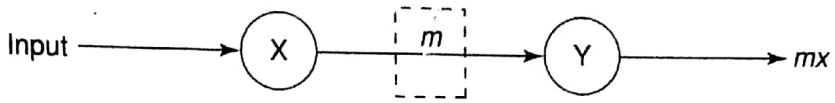

Figure 2-2 Neural net of pure linear equation.

Here, to obtain the output y, the slope m is directly multiplied with the input signal. This is a linear equation. Thus, when slope and input are linearly varied, the output is also linearly varied, as shown in Figure 2-3.

This shows that the weight involved in the ANN is equivalent to the slope of the linear straight line.

2.1.2 Biological Neural Network

It is well-known that the human brain consists of a huge number of neurons, approximately 10^{11}, with numerous interconnections. A schematic diagram of a biological neuron is shown in Figure 2-4.

The biological neuron depicted in Figure 2-4 consists of three main parts:

1. *Soma* or *cell body* – where the cell nucleus is located.
2. *Dendrites* – where the nerve is connected to the cell body.
3. *Axon* – which carries the impulses of the neuron.

Dendrites are tree-like networks made of nerve fiber connected to the cell body. An axon is a single, long connection extending from the cell body and carrying signals from the neuron. The end of the axon splits into fine strands. It is found that each strand terminates into a small bulb-like organ called *synapse*. It is through synapse that the neuron introduces its signals to other nearby neurons. The receiving ends of these synapses on the nearby neurons can be found both on the dendrites and on the cell body. There are approximately 10^4 synapses per neuron in the human brain.

Electric impulses are passed between the synapse and the dendrites. This type of signal transmission involves a chemical process in which specific transmitter substances are released

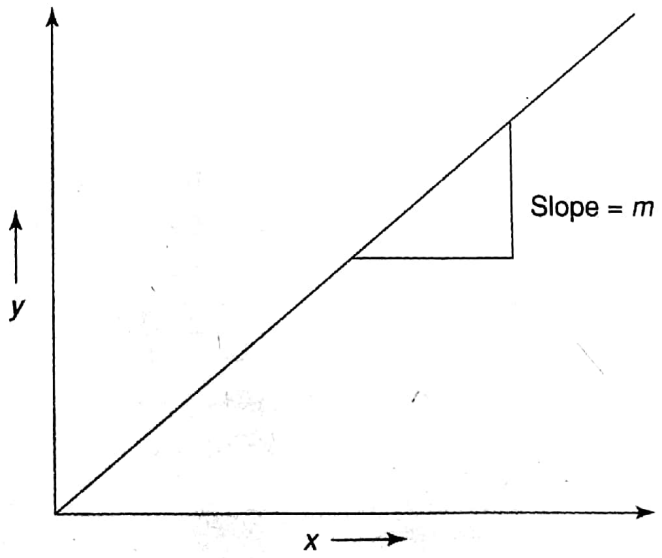

Figure 2-3 Graph for $y = mx$.

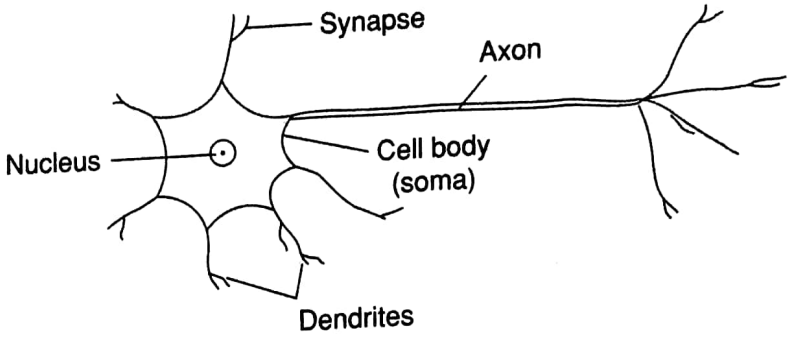

Figure 2-4 Schematic diagram of a biological neuron.

from the sending side of the junction. This results in increase or decrease in the electric potential inside the body of the receiving cell. If the electric potential reaches a threshold then the receiving cell fires and a *pulse* or *action potential* of fixed strength and duration is sent out through the axon to the synaptic junctions of the other cells. After firing, a cell has to wait for a period of time called the *refractory period* before it can fire again. The synapses are said to be *inhibitory* if they let passing impulses hinder the firing of the receiving cell or *excitatory* if they let passing impulses cause the firing of the receiving cell.

Figure 2-5 shows a mathematical representation of the above-discussed chemical processing taking place in an artificial neuron.

In this model, the net input is elucidated as

$$y_{\text{in}} = x_1 w_1 + x_2 w_2 + \cdots + x_n w_n$$
$$= \sum_{i=1}^{n} x_i w_i$$

where i represents the i^{th} processing element. The activation function is applied over it to calculate the output. The weight represents the strength of synapse connecting the input and the output neurons. A positive weight corresponds to an excitatory synapse, and a negative weight corresponds to an inhibitory synapse.

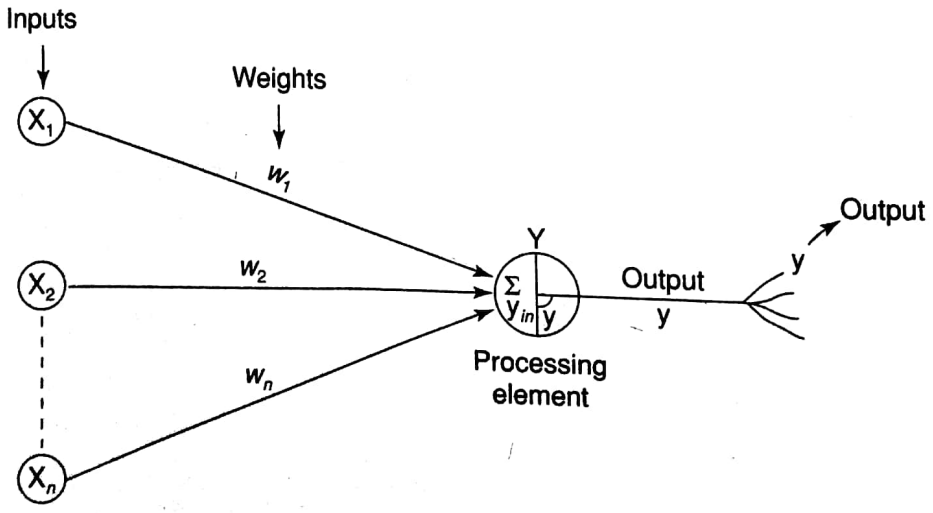

Figure 2-5 Mathematical model of artificial neuron.

Table 2-1 Terminology relationships between biological and artificial neurons

Biological neuron	*Artificial neuron*
Cell	Neuron
Dendrites	Weights or interconnections
Soma	Net input
Axon	Output

The terms associated with the biological neuron and their counterparts in artificial neuron are presented in Table 2-1.

2.1.3 Brain vs. Computer – Comparison Between Biological Neuron and Artificial Neuron (Brain vs. Computer)

A comparison could be made between biological and artificial neurons on the basis of the following criteria:

1. Speed: The cycle time of execution in the ANN is of few nanoseconds whereas in the case of biological neuron it is of a few milliseconds. Hence, the artificial neuron modeled using a computer is more faster.
2. Processing: Basically, the biological neuron can perform massive parallel operations simultaneously. The artificial neuron can also perform several parallel operations simultaneously But, in general, the artificial neuron network process is faster than that of the brain.
3. Size and complexity: The total number of neurons in the brain is about 10^{11} and the total number of interconnections is about 10^{15}. Hence, it can be noted that the complexity of the brain is comparatively higher, i.e. the computational work takes places not only in the brain cell body, but also in axon, synapse, etc. On the other hand, the size and complexity of an ANN is based on the chosen application and the network designer. The size and complexity of a biological neuron is more than that of an artificial neuron.
4. Storage capacity (memory): The biological neuron stores the information in its interconnections or in synapse strength but in an artificial neuron it is stored in its contiguous memory locations. In an artificial neuron, the continuous loading of new information may sometimes overload the memory locations. As a result, some of the addresses containing older memory locations may be destroyed. But in case of the brain, new information can be added in the interconnections by adjusting the strength without destroying the older information. A disadvantage related to brain is that sometimes its memory may fail to recollect the stored information whereas in an artificial neuron, once the information is stored in its memory locations, it can be retrieved. Owing to these facts, the adaptability is more toward an artificial neuron.

5. Tolerance: The biological neuron possesses fault tolerant capability whereas the artificial neuron has no fault tolerance. The distributed nature of the biological neurons enables to store and retrieve information even when the interconnections in them get disconnected. Thus biological neurons are fault tolerant. But in case of artificial neurons, the information gets corrupted if the network interconnections are disconnected. Biological neurons can accept redundancies, which is not possible in artificial neurons. Even when some cells die, the human nervous system appears to be performing with the same efficiency.

6. Control mechanism: In an artificial neuron modeled using a computer, there is a control unit present in Central Processing Unit, which can transfer and control precise scalar values from unit to unit, but there is no such control unit for monitoring in the brain. The strength of a neuron in the brain depends on the active chemicals present and whether neuron connections are strong or weak as a result of structure layer rather than individual synapses. However, the ANN possesses simpler interconnections and is free from chemical actions similar to those taking place in brain (biological neuron). Thus, the control mechanism of an artificial neuron is very simple compared to that of a biological neuron.

So, we have gone through a comparison between ANNs and biological neural networks. In short, we can say an ANN possesses the following characteristics:

1. It is a neurally implemented mathematical model.
2. There exist a large number of highly interconnected processing elements called *neurons* in an ANN.
3. The interconnections with their weighted linkages hold the informative knowledge.
4. The input signals arrive at the processing elements through connections and connecting weights.
5. The processing elements of the ANN have the ability to learn, recall and generalize from the given data by suitable assignment or adjustment of weights.
6. The computational power can be demonstrated only by the collective behavior of neurons, and it should be noted that no single neuron carries specific information.

The above-mentioned characteristics make the ANNs as connectionist models, parallel distributed processing models, self-organizing systems, neuro-computing systems and neuromorphic systems.

2.2 Evolution of Neural Networks

The evolution of neural networks has been facilitated by the rapid development of architectures and algorithms that are currently being used. The history of the development of neutral networks along with the names of their designers is outlined Table 2-2.

Table 2-2 Evolution of neural networks

Year	*Neural network*	*Designer*	*Description*
1943	McCulloch and Pitts neuron	McCulloch and Pitts	The arrangement of neurons in this case is a combination of logic functions. Unique feature of this neuron is the concept of threshold.
1949	Hebb network	Hebb	It is based upon the fact that if two neurons are found to be active simultaneously then the strength of the connection between them should be increased.
1958, 1959, 1962, 1988	Perceptron	Frank Rosenblatt, Block, Minsky and Papert	Here the weights on the connection path can be adjusted.
1960	Adaline	Widrow and Hoff	Here the weights are adjusted to reduce the difference between the net input to the output unit and the desired output. The result here is very negligible. Mean squared error is obtained.
1972	Kohonen self-organizing feature map	Kohonen	The concept behind this network is that the inputs are clustered together to obtain a fired output neuron. The clustering is performed by winner-take all policy.
1982, 1984, 1985, 1986, 1987	Hopfield network	John Hopfield and Tank	This neural network is based on fixed weights. These nets can also act as associative memory nets.
1986	Back-propagation network	Rumelhart, Hinton and Williams	This network is multi-layer with error being propagated backwards from the output units to the hidden units.
1986	Counter-propagation network	Grossberg	This network is similar to the Kohonen network; here the learning occurs for all units in a particular layer, and there exists no competition among these units.
1987–1990	Adaptive Resonance Theory (ART)	Carpenter and Grossberg	The ART network is designed for both binary inputs and analog valued inputs. Here the input patterns can be presented in any order.
1988	Radial basis function network	Broomhead and Lowe	This resembles a back propagation network but the activation function used is a Gaussian function.
1988	Neo cognitron	Fukushima	This network is essential for character recognition. The deficiency occurred in cognitron network (1975) was corrected by this network.

In the later years, the discovery of the neural net resulted in the implementation of optical neural nets, Boltzmann machine, spatiotemporal nets, pulsed neural networks and support vector machines.

2.3 Basic Models of Artificial Neural Network

The models of ANN are specified by the three basic entities namely:

1. the model's synaptic interconnections;
2. the training or learning rules adopted for updating and adjusting the connection weights;
3. their activation functions.

2.3.1 Connections

The neurons should be visualized for their arrangements in layers. An ANN consists of a set of highly interconnected processing elements (neurons) such that each processing element output is found to be connected through weights to the other processing elements or to itself; delay lead and lag-free connections are allowed. Hence, the arrangements of these processing elements and the geometry of their interconnections are essential for an ANN. The point where the connection originates and terminates should be noted, and the function of each processing element in an ANN should be specified.

Besides the simple neuron shown in Figure 2-1, there exist several other types of neural network connections. The arrangement of neurons to form layers and the connection pattern formed within and between layers is called the *network architecture*. There exist five basic types of neuron connection architectures. They are:

1. single-layer feed-forward network;
2. multilayer feed-forward network;
3. single node with its own feedback;
4. single-layer recurrent network;
5. multilayer recurrent network.

Figures 2-6–2-10 depict the five types of neural network architectures.

Basically, neural nets are classified into single-layer or multilayer neural nets. A layer is formed by taking a processing element and combining it with other processing elements. Practically, a layer implies a stage, going stage by stage, i.e., the input stage and the output stage are linked with each other. These linked interconnections lead to the formation of various network architectures. When a layer of the processing nodes is formed, the inputs can be connected to these nodes with various weights, resulting in a series of outputs, one per node. Thus, a single-layer *feed-forward network* is formed.

A multilayer feed-forward network (Figure 2-7) is formed by the interconnection of several layers. The input layer is that which receives the input and this layer has no function except

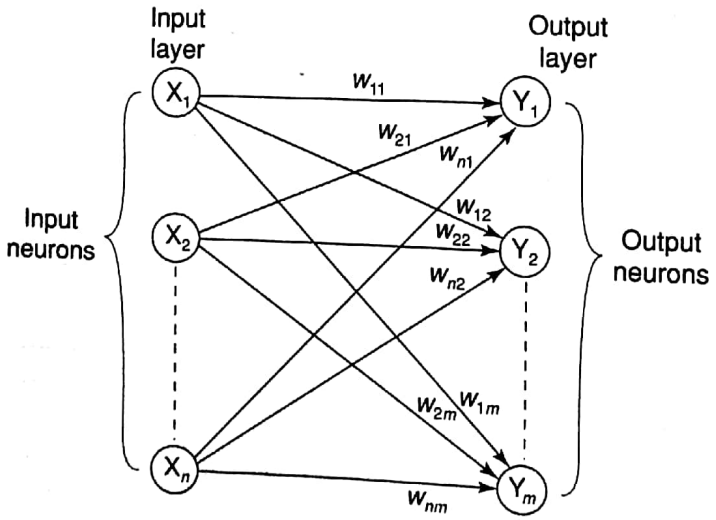

Figure 2-6 Single-layer feed-forward network.

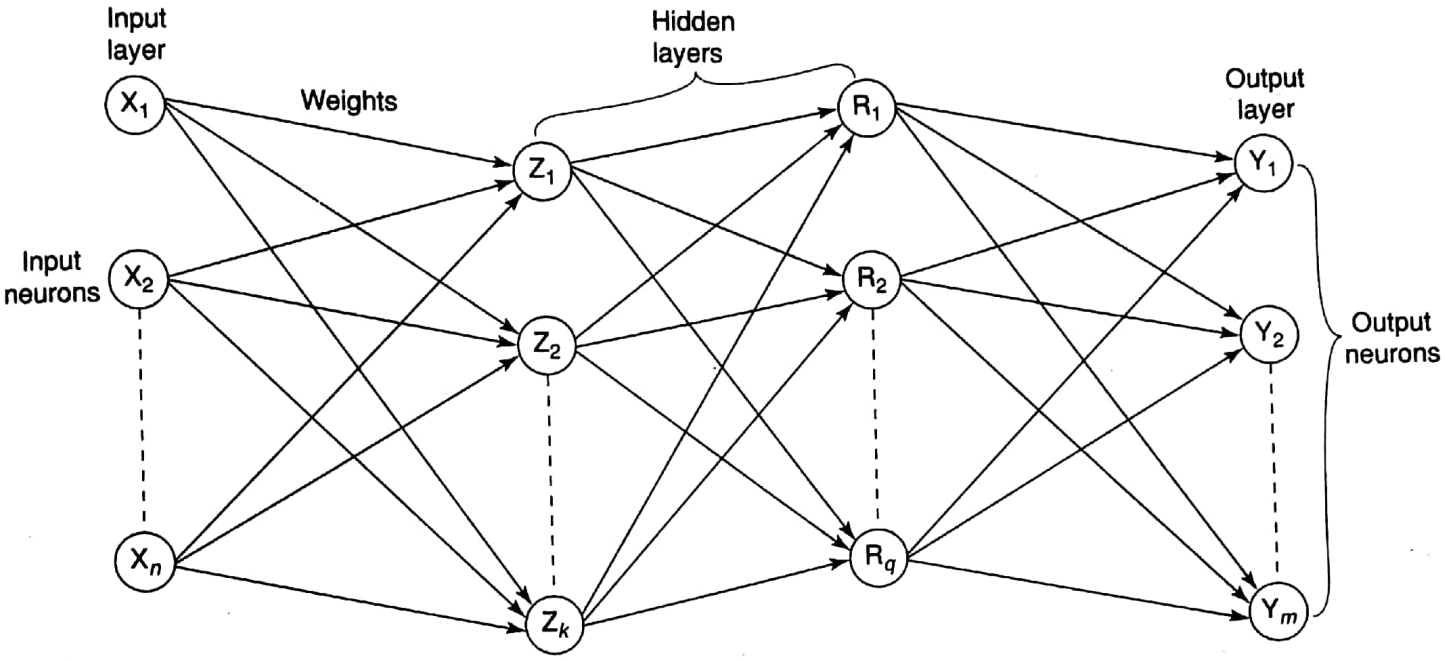

Figure 2-7 Multilayer feed-forward network.

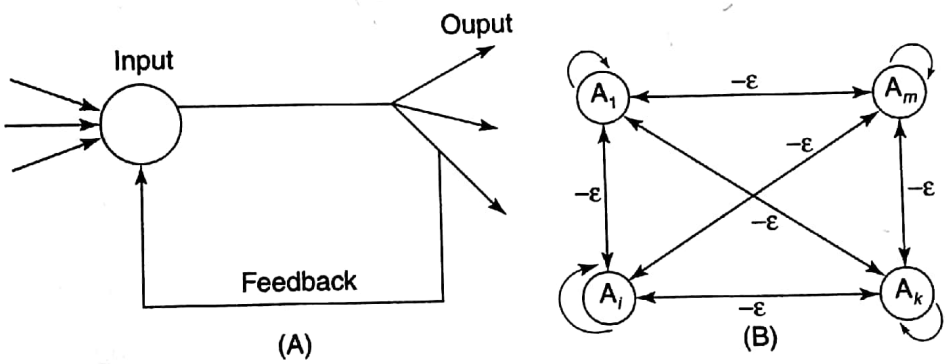

Figure 2-8 (A) Single node with own feedback. (B) Competitive nets.

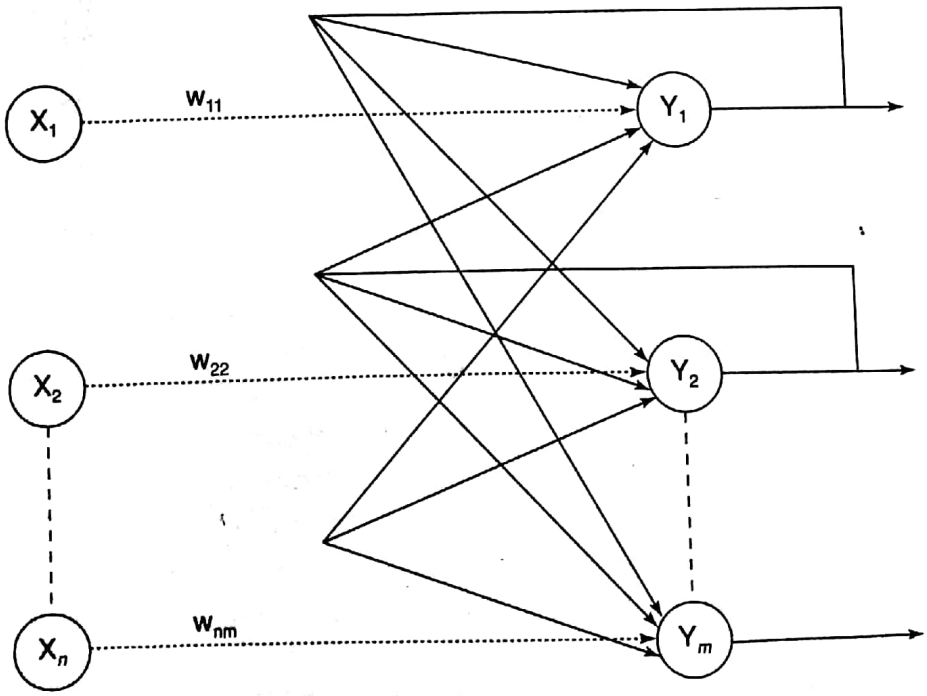

Figure 2-9 Single-layer recurrent network.

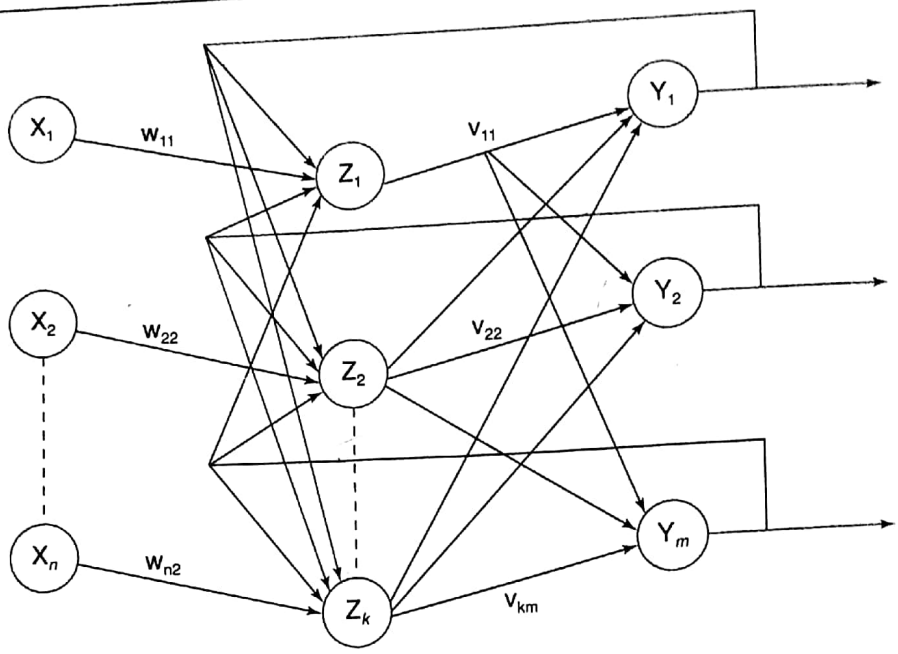

Figure 2-10 Multilayer recurrent network.

buffering the input signal. The output layer generates the output of the network. Any layer that is formed between the input and output layers is called *hidden layer*. This hidden layer is internal to the network and has no direct contact with the external environment. It should be noted that there may be zero to several hidden layers in an ANN. More the number of the hidden layers, more is the complexity of the network. This may, however, provide an efficient output response. In case of a fully connected network, every output from one layer is connected to each and every node in the next layer.

A network is said to be a feed-forward network if no neuron in the output layer is an input to a node in the same layer or in the preceding layer. On the other hand, when outputs can be directed back as inputs to same or preceding layer nodes then it results in the formation of *feedback networks*.

If the feedback of the output of the processing elements is directed back as input to the processing elements in the same layer then it is called *lateral feedback*: Recurrent networks are feedback networks with closed loop. Figure 2-8(A) shows a simple recurrent neural network having a single neuron with feedback to itself. Figure 2-9 shows a single-layer network with a feedback connection in which a processing element's output can be directed back to the processing element itself or to the other processing element or to both.

The architecture of a competitive layer is shown in Figure 2-8(B), the competitive interconnections having fixed weights of $-\varepsilon$. This net is called Maxnet, and will be discussed in the unsupervised learning network category. Apart from the network architectures discussed so far, there also exists another type of architecture with lateral feedback, which is called the *on-center-off-surround* or *lateral inhibition structure*. In this structure, each processing neuron receives two different classes of inputs – "excitatory" input from nearby processing elements and "inhibitory" inputs from more distantly located processing elements. This type of interconnection is shown in Figure 2-11.

In Figure 2-11, the connections with open circles are excitatory connections and the links with solid connective circles are inhibitory connections. From Figure 2-10, it can be noted

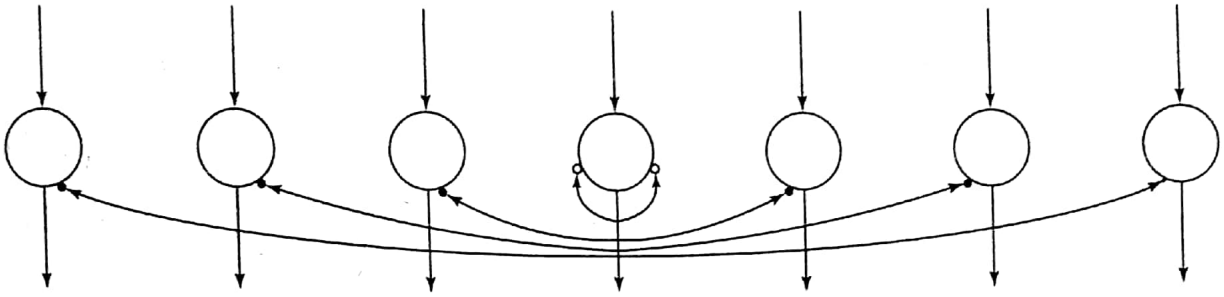

Figure 2-11 Lateral inhibition structure.

that a processing element output can be directed back to the nodes in a preceding layer, forming a *multilayer recurrent network*. Also, in these networks, a processing element output can be directed back to the processing element itself and to other processing elements in the same layer. Thus, the various network architectures as discussed from Figures 2-6–2-11 can be used for giving effective solution to a problem by using ANN.

2.3.2 Learning

The main property of an ANN is its capability to learn. Learning or training is a process by means of which a neural network adapts itself to a stimulus by making proper parameter adjustments, resulting in the production of desired response. Broadly, there are two kinds of learning in ANNs:

1. Parameter learning: It updates the connecting weights in a neural net.
2. Structure learning: It focuses on the change in network structure (which includes the number of processing elements as well as their connection types).

The above two types of learning can be performed simultaneously or separately. Apart from these two categories of learning, the learning in an ANN can be generally classified into three categories as

1. supervised learning;
2. unsupervised learning;
3. reinforcement learning.

Let us discuss these learning types in detail.

2.3.2.1 Supervised Learning

The learning here is performed with the help of a teacher. Let us take the example of the learning process of a small child. The child doesn't know how to read/write. He/she is being taught by the parents at home and by the teacher in school. The children are trained and molded to recognize the alphabets, numerals, etc. Their each and every action is supervised by a teacher. Actually, a child works on the basis of the output that he/she has to produce. All these real-time events involve supervised learning methodology. Similarly, in ANNs following the supervised learning, each input vector requires a corresponding target vector, which represents the desired output. The input vector along with the target vector is called *training pair*. The network here is informed precisely about what should be emitted as output. The block diagram of Figure 2-12 depicts the working of a supervised learning network.

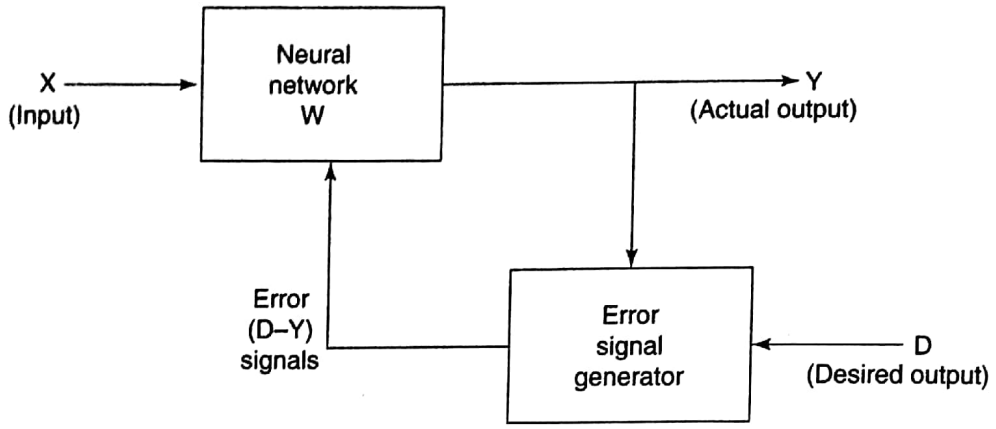

Figure 2-12 Supervised learning.

During training, the input vector is presented to the network, which results in an output vector. This output vector is the actual output vector. Then the actual output vector is compared with the desired (target) output vector. If there exists a difference between the two output vectors then an error signal is generated by the network. This error signal is used for adjustment of weights until the actual output matches the desired (target) output. In this type of training, a supervisor or teacher is required for error minimization. Hence, the network trained by this method is said to be using supervised training methodology. In supervised learning, it is assumed that the correct "target" output values are known for each input pattern.

2.3.2.2 *Unsupervised Learning*

The learning here is performed without the help of a teacher. Consider the learning process of a tadpole, it learns by itself, that is, a child fish learns to swim by itself, it is not taught by its mother. Thus, its learning process is independent and is not supervised by a teacher. In ANNs following unsupervised learning, the input vectors of similar type are grouped without the use of training data to specify how a member of each group looks or to which group a number belongs. In the training process, the network receives the input patterns and organizes these patterns to form clusters. When a new input pattern is applied, the neural network gives an output response indicating the class to which the input pattern belongs. If for an input, a pattern class cannot be found then a new class is generated. The block diagram of unsupervised learning is shown in Figure 2-13.

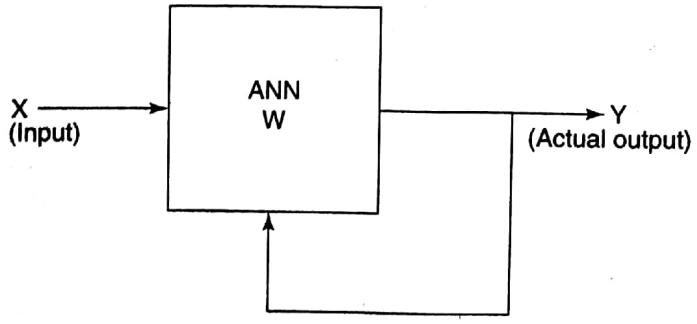

Figure 2-13 Unsupervised learning.

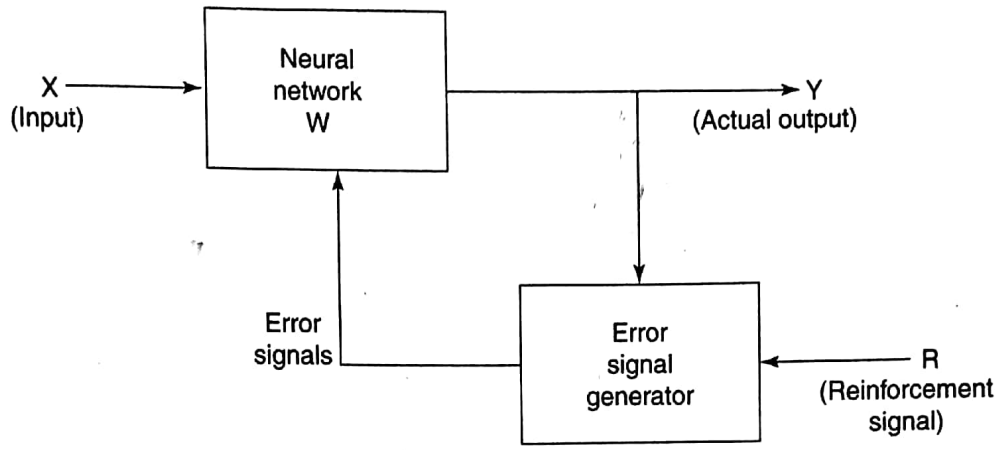

Figure 2-14 Reinforcement learning.

From Figure 2-13 it is clear that there is no feedback from the environment to inform what the outputs should be or whether the outputs are correct. In this case, the network must itself discover patterns, regularities, features or categories from the input data and relations for the input data over the output. While discovering all these features, the network undergoes change in its parameters. This process is called *self-organizing* in which exact clusters will be formed by discovering similarities and dissimilarities among the objects.

2.3.2.3 *Reinforcement Learning*

This learning process is similar to supervised learning. In the case of supervised learning, the correct target output values are known for each input pattern. But, in some cases, less information might be available. For example, the network might be told that its actual output is only "50% correct" or so. Thus, here only critic information is available, not the exact information. The learning based on this critic information is called *reinforcement learning* and the feedback sent is called *reinforcement signal.*

The block diagram of reinforcement learning is shown in Figure 2-14.

The reinforcement learning is a form of supervised learning because the network receives some feedback from its environment. However, the feedback obtained here is only evaluative and not instructive. The external reinforcement signals are processed in the critic signal generator, and the obtained critic signals are sent to the ANN for adjustment of weights properly so as to get better critic feedback in future. The reinforcement learning is also called learning with a critic as opposed to learning with a teacher, which indicates supervised learning.

So, now you've a fair understanding of the three generalized learning rules used in the training process of ANNs.

2.3.3 Activation Functions

To better understand the role of the activation function, let us assume a person is performing some work. To make the work more efficient and to obtain exact output, some force or activation may be given. This activation helps in achieving the exact output. In a similar way, the activation function is applied over the net input to calculate the output of an ANN.

The information processing of a processing element can be viewed as consisting of two major parts: input and output. An integration function (say f) is associated with the input

of a processing element. This function serves to combine activation, information or evidence from an external source or other processing elements into a net input to the processing element. The nonlinear activation function is used to ensure that a neuron's response is bounded – that is, the actual response of the neuron is conditioned or dampened as a result of large or small activating stimuli and is thus controllable.

Certain nonlinear functions are used to achieve the advantages of a multilayer network from a single-layer network. When a signal is fed through a multilayer network with linear activation functions, the output obtained remains same as that could be obtained using a single-layer network. Due to this reason, nonlinear functions are widely used in multilayer networks compared to linear functions.

There are several activation functions. Let us discuss a few in this section:

1. *Identity function*: It is a linear function and can be defined as

$$f(x) = x \quad \text{for all } x$$

The output here remains the same as input. The input layer uses the identity activation function.

2. *Binary step function*: This function can be defined as

$$f(x) = \begin{cases} 1 & \text{if } x \geqslant \theta \\ 0 & \text{if } x < \theta \end{cases}$$

where θ represents the threshold value. This function is most widely used in single-layer nets to convert the net input to an output that is a binary (1 or 0).

3. *Bipolar step function*: This function can be defined as

$$f(x) = \begin{cases} 1 & \text{if } x \geqslant \theta \\ -1 & \text{if } x < \theta \end{cases}$$

where θ represents the threshold value. This function is also used in single-layer nets to convert the net input to an output that is bipolar (+1 or −1).

4. *Sigmoidal functions*: The sigmoidal functions are widely used in back-propagation nets because of the relationship between the value of the functions at a point and the value of the derivative at that point which reduces the computational burden during training. Sigmoidal functions are of two types:
   - *Binary sigmoid function:* It is also termed as logistic sigmoid function or unipolar sigmoid function. It can be defined as

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

where λ is the steepness parameter. The derivative of this function is

$$f'(x) = \lambda f(x)[1 - f(x)]$$

Here the range of the sigmoid function is from 0 to 1.

   - *Bipolar sigmoid function*: This function is defined as

$$f(x) = \frac{2}{1 + e^{-\lambda x}} - 1 = \frac{1 - e^{-\lambda x}}{1 + e^{-\lambda x}}$$

where λ is the steepness parameter and the sigmoid function range is between −1 and +1. The derivative of this function can be

$$f'(x) = \frac{\lambda}{2}[1 + f(x)][1 - f(x)]$$

The bipolar sigmoidal function is closely related to hyperbolic tangent function, which is written as

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

$$h(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}}$$

The derivative of the hyperbolic tangent function is

$$h'(x) = [1 + h(x)][1 - h(x)]$$

If the network uses a binary data, it is better to convert it to bipolar form and use the bipolar sigmoidal activation function or hyperbolic tangent function.

5. *Ramp function*: The ramp function is defined as

$$f(x) = \begin{cases} 1 & \text{if } x > 1 \\ x & \text{if } 0 \leq x \leq 1 \\ 0 & \text{if } x < 0 \end{cases}$$

The graphical representations of all the activation functions are shown in Figure 2-15(A)–(F).

2.4 Important Terminologies of ANNs

This section introduces you to the various terminologies related with ANNs.

2.4.1 Weights

In the architecture of an ANN, each neuron is connected to other neurons by means of directed communication links, and each communication link is associated with weights. The weights contain information about the input signal. This information is used by the net to solve a problem. The weight can be represented in terms of matrix. The weight matrix can also be called *connection matrix*. To form a mathematical notation, it is assumed that there are "*n*" processing elements in an ANN and each processing element has exactly "*m*" adaptive weights. Thus, the weight matrix W is defined by

$$W = \begin{bmatrix} w_1^T \\ w_2^T \\ . \\ . \\ . \\ w_n^T \end{bmatrix} = \begin{bmatrix} w_{11} & w_{12} & . & . & . & w_{1m} \\ w_{21} & w_{22} & . & . & . & w_{2m} \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ w_{n1} & w_{n2} & . & . & . & w_{nm} \end{bmatrix}$$

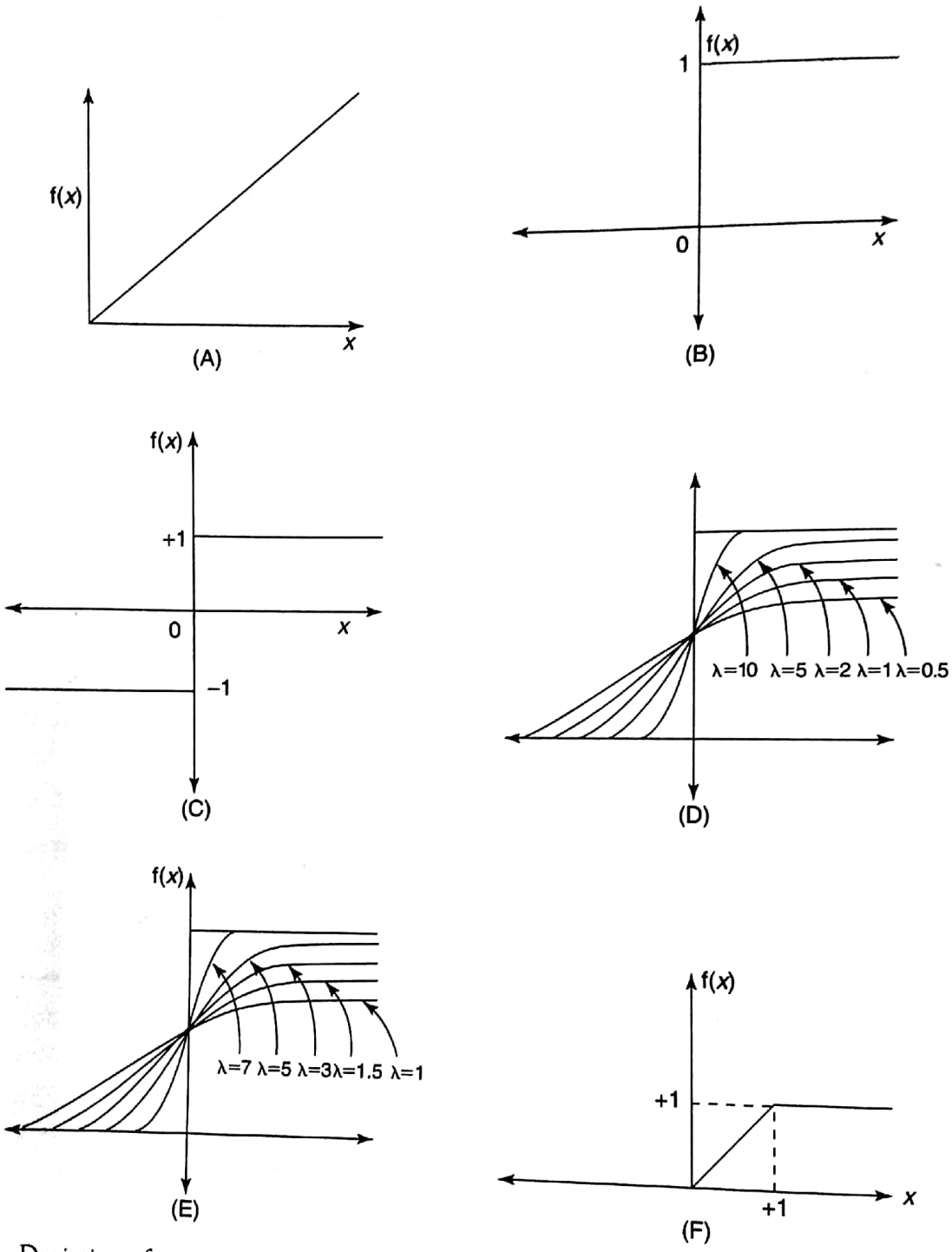

Figure 2-15 Depiction of activation functions: (A) identity function; (B) binary step function; (C) bipolar step function; (D) binary sigmoidal function; (E) bipolar sigmoidal function; (F) ramp function.

where $w_i = [w_{i1}, w_{i2}, \ldots, w_{im}]^T, i = 1, 2, \ldots, n$, is the weight vector of processing element and w_{ij} is the weight from processing element "i" (source node) to processing element "j" (destination node).

If the weight matrix W contains all the adaptive elements of an ANN, then the set of all W matrices will determine the set of all possible information processing configurations for this ANN. The ANN can be realized by finding an appropriate matrix W. Hence, the weights

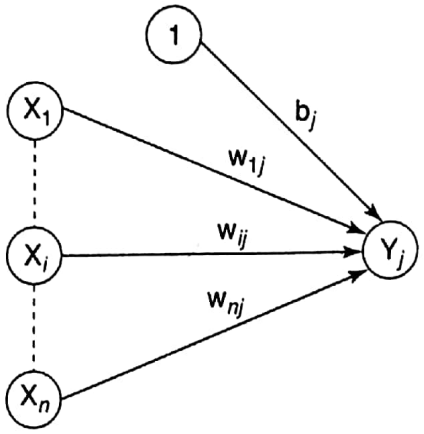

Figure 2-16 Simple net with bias.

encode long-term memory (LTM) and the activation states of neurons encode short-term memory (STM) in a neural network.

2.4.2 Bias

The bias included in the network has its impact in calculating the net input. The bias is included by adding a component $x_0 = 1$ to the input vector X. Thus, the input vector becomes

$$X = (1, X_1, \ldots, X_i, \ldots, X_n)$$

The bias is considered like another weight, that is, $w_{0j} = b_j$.

Consider a simple network shown in Figure 2-16 with bias. From Figure 2-16, the net input to the output neuron Y_j is calculated as

$$\begin{aligned} y_{inj} &= \sum_{i=0}^{n} x_i w_{ij} \\ &= x_0 w_{0j} + x_1 w_{1j} + x_2 w_{2j} + \cdots + x_n w_{nj} \\ &= w_{0j} + \sum_{i=1}^{n} x_i w_{ij} \end{aligned}$$

$$y_{inj} = b_j + \sum_{i=1}^{n} x_i w_{ij}$$

The activation function discussed in Section 2.3.3 is applied over this net input to calculate the output.

The bias can also be explained as follows:

Consider an equation of straight line,

$$y = mx + c$$

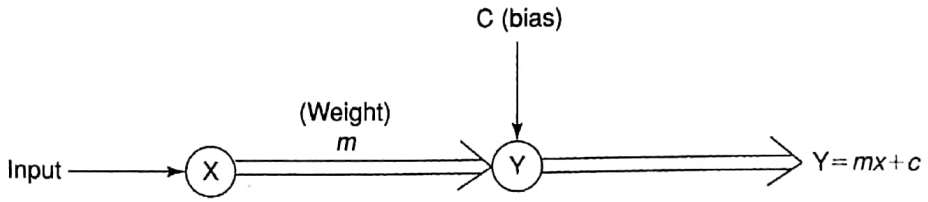

Figure 2-17 Block diagram for straight line.

where,
x is the input,
m is the weight,
c is the bias
y is the output.

The above equation can also be represented as a block diagram shown in Figure 2-17. Thus, bias plays a major role in determining the output of the network.

The bias can be of two types: (i) positive bias and (ii) negative bias. The positive bias helps in increasing the net input of the network and the negative bias helps in decreasing the net input of the network. Thus, as a result of the bias effect, the output of the network can be varied.

2.4.3 Threshold

Threshold is a set value based upon which the final output of the network may be calculated. The threshold value is used in the activation function. A comparison is made between the calculated net input and the threshold to obtain the network output. For each and every application, there is a threshold limit. Consider a direct current (DC) motor. If its maximum speed is 1500 rpm then the threshold based on the speed is 1500 rpm. If the motor is run on a speed higher than its set threshold, it may damage motor coils. Similarly, in neural networks, based on the threshold value, the activation functions are defined and the output is calculated. The activation function using threshold can be defined as

$$f(\text{net}) = \begin{cases} 1 & \text{if net} \geqslant \theta \\ -1 & \text{if net} < \theta \end{cases}$$

where θ is the fixed threshold value.

2.4.4 Learning Rate

The learning rate is denoted by "α." It is used to control the amount of weight adjustment at each step of training. The learning rate, ranging from 0 to 1, determines the rate of learning at each time step.

2.4.5 Momentum Factor

Convergence is made faster if a momentum factor is added to the weight updation process. This is generally done in the back propagation network. If momentum has to be used, the weights from one or more previous training patterns must be saved. Momentum helps the net in reasonably large weight adjustments until the corrections are in the same general direction for several patterns.

2.4.6 Vigilance Parameter

The vigilance parameter is denoted by "ρ." It is generally used in adaptive resonance theory (ART) network. The vigilance parameter is used to control the degree of similarity required for patterns to be assigned to the same cluster unit. The choice of vigilance parameter ranges approximately from 0.7 to 1 to perform useful work in controlling the number of clusters.

2.4.7 Notations

The notations mentioned in this section have been used in this textbook for explaining each network.

x_i – Activation of unit X_i, input signal.
y_i – Activation of unit Y_j, $y_j = f(y_{inj})$
w_{ij} – Weight on connection from unit X_i to unit Y_j.
b_j – Bias acting on unit j. Bias has a constant activation of 1.
W – Weight matrix, $W = \{w_{ij}\}$
y_{inj} – Net input to unit Y_j:

$$y_{inj} = b_j + \sum_i x_i w_{ij}$$

$\|x\|$ – Norm of magnitude vector X.
θ_j – Threshold for activation of neuron Y_j.
S – Training input vector, $S = (s_1, \ldots, s_i, \ldots, s_n)$
T – Training output vector, $T = (t_1, \ldots, t_j, \ldots, t_n)$
X – Input vector, $X = (x_1, \ldots, x_i, \ldots, x_n)$
Δw_{ij} – change in weights:

$$\Delta w_{ij} = w_{ij}(\text{new}) - w_{ij}(\text{old})$$

α – Learning rate; it controls the amount of weight adjustment at each step of training.

2.5 McCulloch–Pitts Neuron

2.5.1 Theory

The McCulloch–Pitts neuron was the earliest neural network discovered in 1943. It is usually called as *M–P neuron*. The M–P neurons are connected by directed weighted paths. It should be noted that the activation of a M–P neuron is binary, that is, at any time step the neuron may fire or may not fire. The weights associated with the communication links may be excitatory (weight is positive) or inhibitory (weight is negative). All the excitatory connected weights entering into a particular neuron will have same weights. The threshold plays a major role in M–P neuron. There is a fixed threshold for each neuron, and if the net input to the neuron is greater than the threshold then the neuron fires. Also, it should be noted that any nonzero inhibitory input would prevent the neuron from firing. The M–P neurons are most widely used in the case of logic functions.

2.5.2 Architecture

A simple M–P neuron is shown in Figure 2-18.

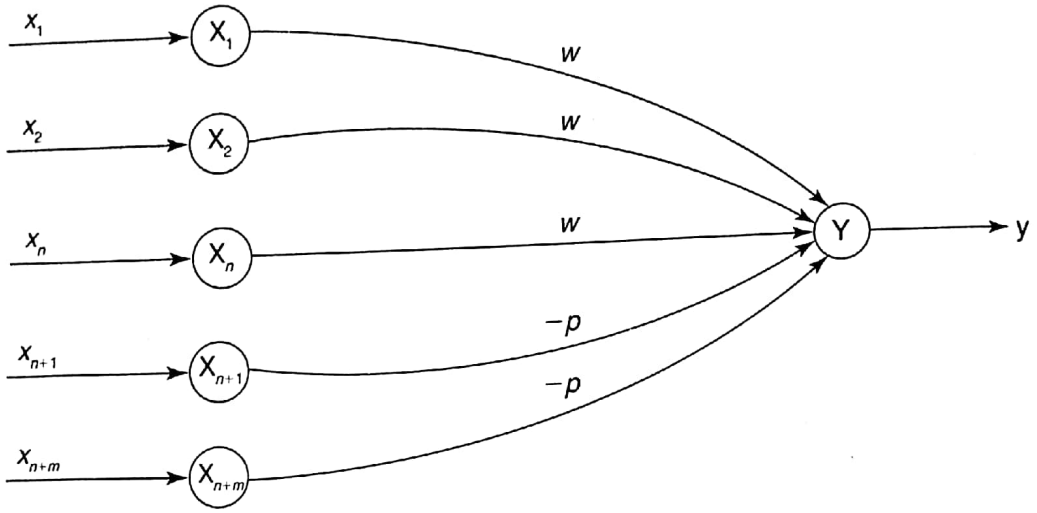

Figure 2-18 McCulloch–Pitts neuron model.

As already discussed, the M–P neuron has both excitatory and inhibitory connections. It is excitatory with weight ($w > 0$) or inhibitory with weight $-p(p < 0)$. In Figure 2-18, inputs from x_1 to x_n possess excitatory weighted connections and inputs from x_{n+1} to x_{n+m} possess inhibitory weighted interconnections. Since the firing of the output neuron is based upon the threshold, the activation function here is defined as

$$f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} \geq \theta \\ 0 & \text{if } y_{in} < \theta \end{cases}$$

For inhibition to be absolute, the threshold with the activation function should satisfy the following condition:

$$\theta > nw - p$$

The output will fire if it receives say "k" or more excitatory inputs but no inhibitory inputs, where

$$kw \geq \theta > (k-1)w$$

The M–P neuron has no particular training algorithm. An analysis has to be performed to determine the values of the weights and the threshold. Here the weights of the neuron are set along with the threshold to make the neuron perform a simple logic function. The M–P neurons are used as building blocks on which we can model any function or phenomenon, which can be represented as a logic function.

2.6 Linear Separability

An ANN does not give exact solution for a nonlinear problem. However, it provides possible approximation solutions to nonlinear problems. Linear separability is the concept wherein the separation of the input space into regions is based on whether the network response is positive or negative.