already seen the benefit of the middle-hidden layer in an artificial neural network. We understand that the hidden layer allows ANN to develop its own internal representation of this mapping. Such a rich and complex internal representation capability allows the hierarchical network to learn any mapping and not just linearly separable ones. Let us consider the three-layer network with input layer having 'l' nodes, hidden layer having 'm' nodes, and an output layer with 'n' nodes. We consider sigmoidal functions for activation functions for the hidden and output layers and linear activation function for input layer. The number of neurons in the hidden layer may be chosen to lie between l and 2l. The basic algorithm loop structure is given as

```
Initialize the weights
Repeat
    For each training pattern
      Train on that pattern
    End
Until the error is acceptably low
```

Algorithm 3.1 illustrates the step by step procedure of the backpropagation algorithm.

Algorithm 3.1 (Backpropagation Learning Algorithm)

Algorithm BPN( )

Step 1 : Normalize the inputs and outputs with respect to their maximum values. It is proved that the neural networks work better if input and outputs lie between 0-1. For each training pair, assume there are 'l' inputs given by $\underset{l \times 1}{\{I\}_I}$ and 'n' outputs $\underset{n \times 1}{\{O\}_O}$ in a normalised form.

Step 2 : Assume the number of neurons in the hidden layer to lie between $l < m < 2l$

Step 3 : $[V]$ Represents the weights of synapses connecting input neurons and hidden neurons and $[W]$ represents weights of synapses connecting hidden neurons and output neurons. Initialize the weights to small random values usually from -1 to 1. For general problems, λ can be assumed as 1 and the threshold values can be taken as zero.

$$[V]^0 = [\text{random weights}]$$

$$[W]^0 = [\text{random weights}]$$

$$[\Delta V]^0 = [\Delta W]^0 = [0] \tag{3.51}$$

Step 4 : For the training data, present one set of inputs and outputs. Present the pattern to the input layer $\{I\}_I$ as inputs to the input layer. By using linear activation function, the output of the input layer may be evaluated as

$$\underset{l \times 1}{\{O\}_I} = \underset{l \times 1}{\{I\}_I} \tag{3.52}$$

Step 5 : Compute the inputs to the hidden layer by multiplying corresponding weights of synapses as

$$\underset{m \times 1}{\{I\}_H} = \underset{m \times l}{[V]^T} \underset{l \times 1}{\{O\}_I} \tag{3.53}$$

Step 6 : Let the hidden layer units evaluate the output using the sigmoidal function as

$$\{O\}_H = \begin{Bmatrix} \cdot \\ \cdot \\ \dfrac{1}{(1 + e^{-I_{Hi}})} \\ \cdot \\ \cdot \end{Bmatrix} \tag{3.54}$$

$m \times 1$

Step 7 : Compute the inputs to the output layer by multiplying corresponding weights of synapses as

$$\underset{n \times 1}{\{I\}_O} = \underset{n \times m}{[W]^T} \underset{m \times 1}{\{O\}_H} \tag{3.55}$$

Step 8 : Let the output layer units evaluate the output using sigmoidal function as

$$\{O\}_O = \begin{Bmatrix} \vdots \\ \dfrac{1}{\left(1 + e^{-I_{oj}}\right)} \\ \vdots \end{Bmatrix} \tag{3.56}$$

The above is the network output.

Step 9: Calculate the error and the difference between the network output and the desired output as for the *i*th training set as

$$E^p = \frac{\sqrt{\Sigma\,(T_j - O_{oj})^2}}{n} \tag{3.57}$$

Step 10: Find $\{d\}$ as

$$\{d\} = \begin{Bmatrix} \vdots \\ (T_k - O_{Ok})O_{Ok}(1 - O_{Ok}) \\ \vdots \end{Bmatrix} \tag{3.58}$$

$n \times 1$

Step 11: Find $[Y]$ matrix as

$$[Y] = \{O\}_H \langle d \rangle \tag{3.59}$$

$m \times n \quad m \times 1 \; 1 \times n$

Step 12: Find $[\Delta W]^{t+1} = \alpha[\Delta W]^t + \eta[Y]$ (3.60)

$m \times n \; m \times n \; m \times n$

Step 13: Find $\{e\} = [W]\{d\}$ (3.61a)

$m \times 1 \quad m \times n \quad n \times 1$

$$\{d^*\} = \begin{Bmatrix} \vdots \\ e_i(O_{Hi})(1 - O_{Hi}) \\ \vdots \end{Bmatrix} \tag{3.61b}$$

$m \times 1 \quad m \times 1$

Find $[X]$ matrix as

$$[X] = \{O\}_I \langle d^* \rangle = \{I\}_I \langle d^* \rangle \quad (3.62)$$
$$1 \times m \quad 1 \times 1 \; 1 \times m \quad 1 \times 1 \; 1 \times m$$

Step 14 : Find $[\Delta V]^{t+1} = \alpha[\Delta V]^t + \eta[X]$ (3.63)
$1 \times m \quad 1 \times m \quad 1 \times m$

Step 15 : Find

$$[V]^{t+1} = [V]^t + [\Delta V]^{t+1}$$
$$[W]^{t+1} = [W]^t + [\Delta W]^{t+1} \quad (3.64)$$

Step 16 : Find error rate as

$$\text{error rate} = \frac{\Sigma E_p}{\text{nset}} \quad (3.65)$$

Step 17 : Repeat steps 4-16 until the convergence in the error rate is less than the tolerance value.

End Algorithm BPN

Once the process converges, the final weights should be stored in a file. Now, we are ready to test the neural net. Given any other input, we will be able to get the outputs and this is known as *inference session*. Hence, it is seen that training of an artificial neural network involves two passes. In the forward pass, the input signals propagate from the network input to the output. In the reverse pass, the calculated error signals propagate backwards through the network where they are used to adjust the weights. The calculation of output is carried out layer by layer in the forward direction. The output of one layer in weighted manner will be the input to the next layer. In the reverse pass, the weights of the output neuron layer are adjusted first since the target value of each output neuron is available to guide the adjustment of associated weights.

3.3 ILLUSTRATION

Consider that for a particular problem there are five training sets as shown in Table 3.3.

Table 3.3 Training sets

S. no.	Inputs		Output
	I_1	I_2	O
1	0.4	–0.7	0.1
2	0.3	–0.5	0.05
3	0.6	0.1	0.3
4	0.2	0.4	0.25
5	0.1	–0.2	0.12