Back Propagation Learning:

1) A back-propagation neural n/w is a multilayer, feed-forward neural n/w consisting of an i/p layer, a hidden layer and an o/p layer.
2) The neurons present in the hidden and o/p layers have biases which are the connections from the units whose activation is always 1.
3) The bias terms also act as weights.
4) The architecture of a BPN, depicting only the direⁿ of infoʳᵐ flow for the feed-forward phase but during the back-propagation phase of learning, signals are sent in the reverse dirⁿ.
5) The i/p's are sent to the BPN and the o/p obtained from the net could be either binary (0, 1) or bipolar (-1, +1).
6) The activation funⁿ could be any funⁿ which increases monotonically and is also differentiable.

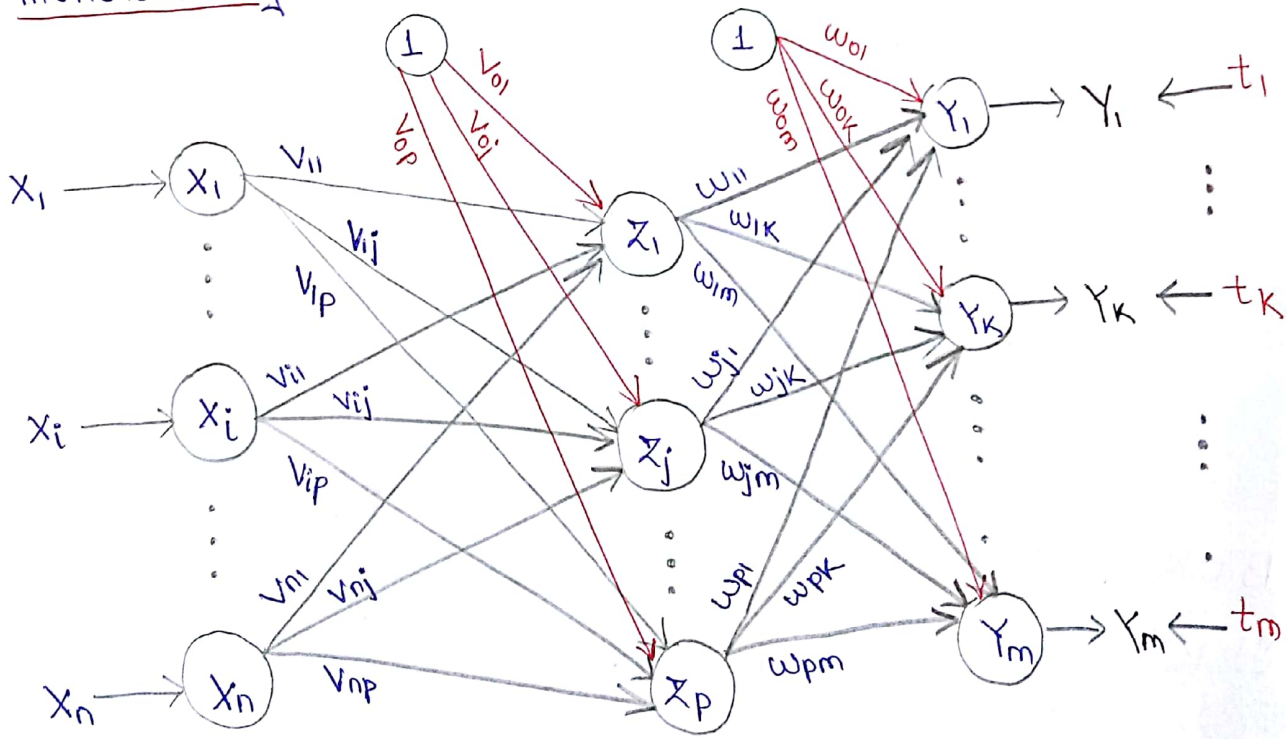

Fig: Architecture of a back-propagation n/w.

Training Algorithm

The error back-propagation learning Algorithm can be outlined in the following algorithm:

Step 0: Initialize weights and learning rate (take some small random values)

Step 1: Perform steps 2-9 when stopping condn is false.

Step 2: Perform steps 3-8 for each training pair.

Feed-forward phase (Phase I):

Step 3: Each i/p unit receives i/p signal x_i and sends it to the hidden unit ($i = 1$ to n).

Step 4: Each hidden unit Z_j ($j = 1$ to p) sums its weighted i/p signals to calculate net i/p:

$$Z_{inj} = V_{0j} + \sum_{i=1}^{n} x_i V_{ij}$$

Calculate o/p of the hidden unit by applying its activation fun^{n}s over Z_{inj} (binary or bipolar sigmoidal activation fun^n):

$$Z_j = f(Z_{inj})$$

and send the o/p signal from the hidden unit to the i/p of output layer units.

Step 5: For each output unit Y_k ($k = 1$ to m). Calculate the net i/p:

$$Y_{ink} = w_{0k} + \sum_{j=1}^{p} Z_j w_{jk}$$

and apply the activation fun^n to compute o/p signal.

$$Y_k = f(Y_{ink})$$

Back-propagation of error (Phase II):

Step 6: Each o/p unit Y_k ($k = 1$ to m) receives a target pattern corresponding to the i/p training pattern & computes the error correction term:

$$\delta_k = (t_k - Y_k) f'(Y_{ink})$$

[for binary sigmoidal — $f'(x) = \lambda f(x)[1 - f(x)]$]

On the basis of the calculated error correction term, update the change in weights & bias.

$$\Delta w_{jk} = \alpha \delta_k Z_j \quad ; \quad \Delta w_{0k} = \alpha \delta_k$$

Also, send δ_K to the hidden layer backwards.

Step 7: Each hidden unit ($z_j, j = 1$ to p) sum its delta ips from the olp units.

$$\delta_{inj} = \sum_{K=1}^{m} \delta_K w_{jK}$$

The term δ_{inj} gets multiplied with the derivative of $f(z_{inj})$ to calculate the error term:

$$\delta_j = \delta_{inj} f'(z_{inj})$$

On the basis of the calculated δ_j, update the change in weights and bias:

$$\Delta v_{ij} = \alpha \delta_j x_i \quad ; \quad \Delta v_{oj} = \alpha \delta_j$$

Weight and bias updation (phase III)

Step 8: Each olp unit ($y_K, K = 1$ to m) updates the bias & weights:

$$w_{jK}(new) = w_{jK}(old) + \Delta w_{jK}$$

$$w_{oK}(new) = w_{oK}(old) + \Delta w_{oK}$$

Each hidden unit ($z_j, j = 1$ to p) updates its bias & weights:

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

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

Step 9: check for the stopping condn. The stopping condn may be certain no. of epochs reached or when the actual olp equals the target olp.

Testing Algorithm of Back-Propagation N/w:

The testing procedure of the BPN is as follows:

Step 0: Initialize the weights. The weights are taken from the training Algorithm.

Step 1: Perform Steps 2-4 for each ip vector.

Step 2: Set the activation of ip unit for x_i ($i = 1$ to n).

Step 3: Calculate the net ip to hidden unit z and its op.

for j = 1 to p,

$$z_{inj} = v_{0j} + \sum_{j=1}^{n} x_i v_{ij}$$

$$z_j = f(z_{inj})$$

Step 4: Now compare the op of the op layer unit. for K=1 to m.

$$y_{ink} = w_{0k} + \sum_{j=1}^{p} z_j w_{jk}$$

$$y_k = f(y_{ink})$$

Use sigmoidal activation functions for calculating the op.

Flowchart of training Process:

The terminologies used in the flowchart and in the training algorithm are as follows:

x = input training vector $(x_1, x_2, \ldots x_i, \ldots, x_n)$

t = target op vector $(t_1, \ldots, t_k, \ldots, t_m)$

α = learning rate parameter

x_i = input unit i. (Since the ip layer uses identity activation fun^n, the ip and op signals here are same.)

v_{0j} = bias on j^{th} hidden unit.

w_{0k} = bias on k^{th} op unit.

z_j = hidden unit j. The net i/p to z_j is

$$z_{inj} = v_{0j} + \sum_{i=1}^{n} x_i v_{ij}$$

and the op is:

$$z_j = f(z_{inj})$$

y_k = op unit K. The net ip to y_k is

$$y_{ink} = w_{0k} + \sum_{j=1}^{p} z_j w_{jk}$$

and the op is:

$$y_k = f(y_{ink})$$

δ_K = error correction weight adjustment for w_{jK} that is due to an error at o/p unit Y_K, which is back-propagated to the hidden units that feed into unit Y_K.

δ_j = error correction weight adjustment for v_{ij} i.e due to the back-propagation of error to the hidden unit Z_j.

* Also, Note that the commonly used Activation fun^{n}s are binary sigmoidal and bipolar sigmoidal activation functions.

* These functions are used in the BPN bcoz of the following characteristics:
  i) Continuity
  ii) differentiability
  iii) non decreasing monotony.

* The range of binary sigmoid is from 0 to 1, and for bipolar sigmoid it is from -1 to +1.

Flow chart