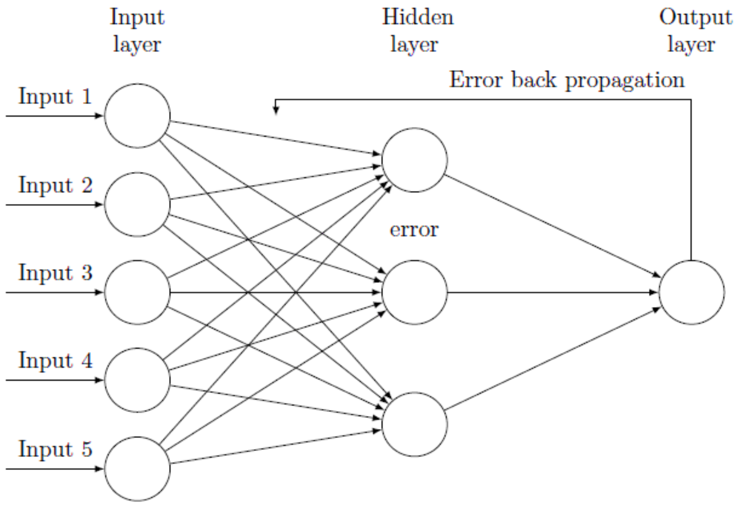
Backpropagation Neural Network Learning Algorithm



1. Randomly initialize the weights of input layer w_i, weights between input and hidden layer w_{ij}, and weights between hidden and output layer w_{jk}.

2. Present input set $x_0, x_1, \dots, x_n$ and desired output set of y.

$REPEAT$

3. Calculation between input and hidden layer—

$$net_{aj} = \sum_{i=1}^{n} w_{ij} * O_{ai}$$

$$active_j = net_{aj} + uh_j$$

$$O_{aj} = \frac{1}{1 + e^{-k_1 * active_j}}$$

Here, k_1 is an arbitrary constant and uh_j is the threshold value of hidden layer.

O_{ai} = Output from input layer.

O_{aj} = Output from hidden layer.

4. Calculation between hidden and output layer—

$$net_{ak} = \sum_{j=1}^{m} w_{jk} * O_{aj}$$

$$active_k = net_{ak} + uO_k$$

$$O_{ak} = \frac{1}{1 + e^{-k_2 * active_k}}$$

Here, k_2 is an arbitrary constant and uO_j is the threshold value of output layer.

O_{aj} = Output from hidden layer.

O_{ak} = Output from output layer.

5. Calculation of error—

$$\delta_{ak} = t_{ak} - O_{ak}$$

Here, t_{ak} = Desired output

O_{ak} = Actual output

6. Update weights in between hidden and output layer—

$$\Delta w_{jk} = \eta_2 * k_2 * \delta_{ak} * O_{ak} * O_{aj} * (1 - O_{aj})$$

$$w_{jk} := w_{jk} + \Delta w_{jk}$$

7. Update thresholds in between hidden and output layer—

$$\Delta uO_k = \eta_2 * k_2 * \delta_{ak} * (1 - O_{ak})$$

$$uO_k := uO_k + \Delta uO_k$$

8. Update weights in between input and hidden layer—

$$\Delta w_{ij} = \eta_1 * k_1 * O_{ai} * O_{aj} * \left(1 - O_{aj}\right) \sum_{j=1}^{m} \delta_{ak} * w_{jk}$$

$$w_{ij} := w_{ij} + \Delta w_{ij}$$

9. Update thresholds in between input and hidden layer—

$$\Delta uh_j = \eta_1 * k_1 * \delta_{ak} * \left(1 - O_{aj}\right) \sum_{j=1}^{m} \delta_{ak} * w_{jk}$$

$$uh_k := uh_j + \Delta uh_j$$

$END\ REPEAT$