

Module 4

BACKPROPAGATION ARCHITECTURE

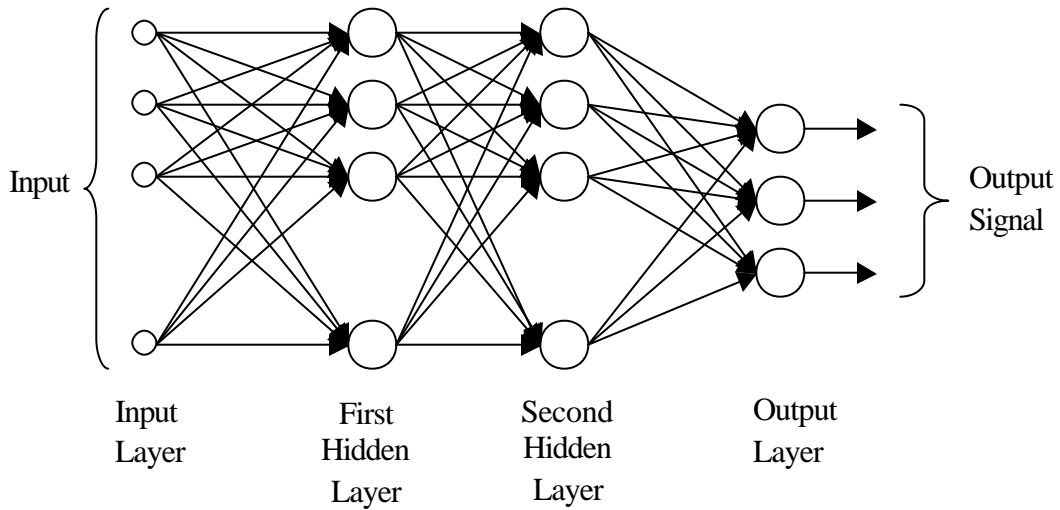


Figure 4.1 Architecture of multilayer perceptron

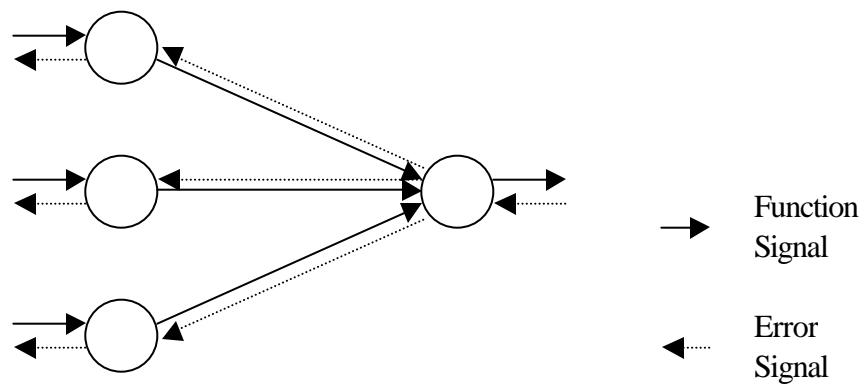


Figure 4.2: Illustration of the directions of two basic signal flows in a multilayer perceptron

BACKPROPAGATION ALGORITHM

We want to adjust the network $w_{ij}^{(n)}$ in order to minimize the sum-square error function

$$E(w_{ij}^{(n)}) = \frac{1}{2} \sum_p \sum_j (d_j^p - o_j^{(n)})^2$$

gradient descent weight update is

$$\Delta w_{kl}^{(m)} = -\eta \frac{\partial E(w_{ij}^{(n)})}{\partial w_{kl}^{(m)}}$$

For a two layer network, the final outputs can be written:

$$\begin{aligned} \text{out}_k^{(2)} &= f\left(\sum_j \text{out}_j^{(1)} w_{jk}^{(2)}\right) \\ &= f\left(\sum_j f\left(\sum_i \text{in}_i w_{ij}^{(1)}\right) w_{jk}^{(2)}\right) \end{aligned}$$

Use chain rules for derivative

$$\frac{\partial E(w_{ij}^{(n)})}{\partial w_{hl}^{(m)}} = -\sum_p \sum_k (d_k - \text{out}_k^{(2)}) \frac{\partial \text{out}_k^{(2)}}{\partial w_{hl}^{(m)}}$$

$$\frac{\partial \text{out}_k^{(2)}}{\partial w_{hl}^{(2)}} = f'\left(\sum_j \text{out}_j^{(1)} w_{jk}^{(2)}\right) \text{out}_h^{(1)} \delta_{kl}$$

$$\frac{\partial \text{out}_k^{(2)}}{\partial w_{hl}^{(2)}} = f'\left(\sum_j \text{out}_j^{(1)} w_{jk}^{(2)}\right) f'\left(\sum_i \text{in}_i w_{il}^{(1)}\right) w_{lk}^{(2)} \text{in}_h$$

$$\Delta w_{hl}^{(2)} = \eta \sum_p (d_k - \text{out}_k^{(2)}) f'\left(\sum_j \text{out}_j^{(1)} w_{jk}^{(2)}\right) \text{out}_h^{(1)}$$

$$\Delta w_{hl}^{(1)} = \eta \sum_p \sum_k (d_k - \text{out}_k^{(2)}) f'\left(\sum_j \text{out}_j^{(1)} w_{jk}^{(2)}\right) f'\left(\sum_j \text{in}_j w_{il}^{(1)}\right) w_{lk}^{(2)} \text{in}_h$$

If the threshold function $f(x)$ is a sigmoid. We can $f'(x) = f(x)(1-f(x))$ to give

$$\Delta w_{hl}^{(2)} = \eta \sum_p (d_k - \text{out}_k^{(2)}) \text{out}_k^{(2)} (1 - \text{out}_k^{(2)}) \text{out}_h^{(1)}$$

$$\Delta w_{hl}^{(1)} = \eta \sum_p \sum_k (d_k - \text{out}_k^{(2)}) \text{out}_k^{(2)} (1 - \text{out}_k^{(2)}) w_{lk}^{(2)} \text{out}_l^{(1)} (1 - \text{out}_l^{(1)}) \text{in}_h$$

These equations constitute the basic Back-Propagation Learning Algorithm.

XOR PROBLEM

It is not possible to find weights that enable Single Layer Perceptrons to deal with non-linearly separable problems like XOR. However, Multi-Layer Perceptrons (MLPs) are able to cope with non-linearly separable problems.

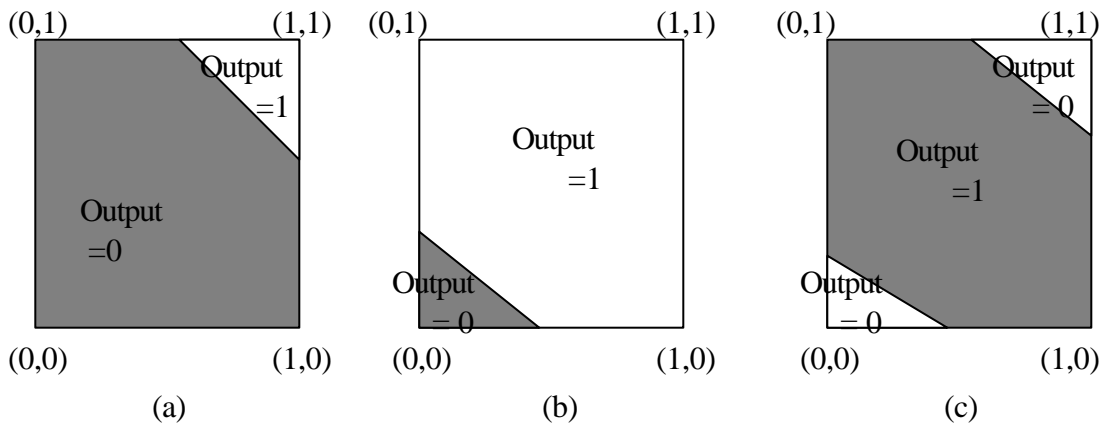
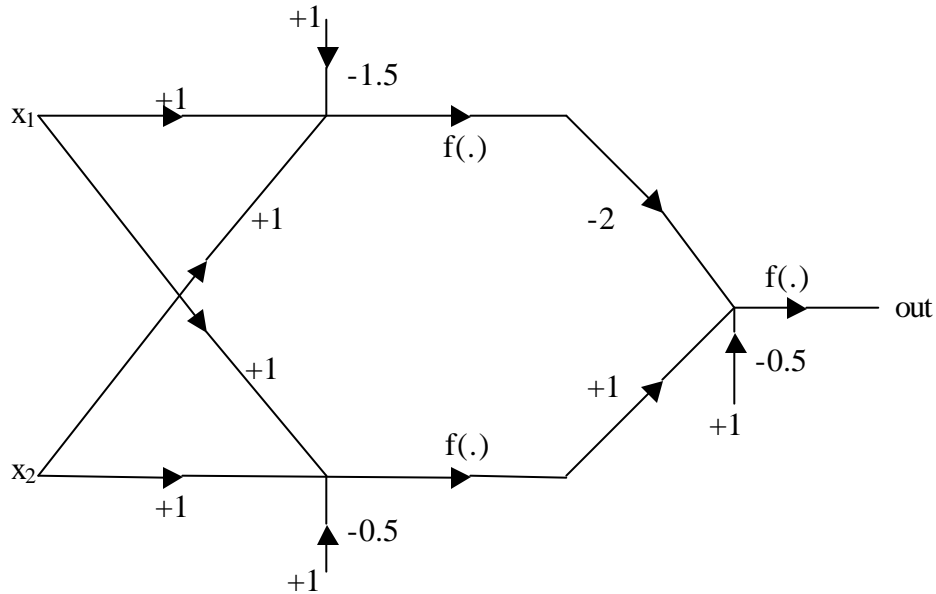


Figure 4.9 (a) Decision boundary constructed by hidden neuron 1. (b) Decision boundary constructed by hidden neuron 2. (c) Decision boundaries constructed by the complete network

HEURISTICS FOR MAKING THE BACKPROPAGATION ALGORITHM PERFORM BETTER

1. Sequence versus batch update
2. maximizing information content
3. Activation function
4. Target value
5. Normalization
6. Initialization
7. Learning from hints
8. Learning rate

VIRTUAL AND LIMITATION OF BACKPROPAGATION LEARNING

1. Connectionism
2. Feature Detection
3. Function Approximation
4. Computational Efficiency
5. Sensitivity Analysis
6. Robustness
7. Convergence
8. Local Minima
9. Scaling

ACCELERATED CONVERGENCE OF BACKPROPAGATION LEARNING

Heuristic 1: Every adjustable network parameter of the cost function should have its own individual learning-rate parameter

Heuristic 2: Every learning rate parameter should allowed to vary from one iteration to the next.

Heuristic 3: When the derivative of the cost function with respect to a synaptic weight has the same algebraic sign for several consecutive iterations of the algorithm, the learning rate parameter for that particular weight should be increased.