Perceptron

The Perceptron Algorithm

Initialize:

- Set all of the weights to small random numbers.

Training:

- For T itereations or until all outputs are correct

  For each input vector

  Compute the activation of each neuron j using

$$y_j = g(\sum_{i=0}^{m} w_{ij}x_i) = \begin{cases} 1 \text{ if } \sum_{i=1}^{m} w_{ij}x_i > 0 \\ 0 \text{ if } \sum_{i=1}^{m} w_{ij}x_i \leq 0 \end{cases}$$

  Update each weight individually using

$$w_{ij} \leftarrow w_{ij} - \eta(y_j - t_j) \cdot x_i.$$

Recall:

Compute the activation of each neuron j using:

$$y_j = g(\sum_{i=0}^{m} w_{ij}x_i) = \begin{cases} 1 \text{ if } w_{ij}x_i > 0 \\ 0 \text{ if } w_{ij}x_i \leq 0 \end{cases}$$

The Perceptron

Perceptron Convergence Theorem

Given a linearly separable dataset, the Perceptron will converge to a solution that separates the classes, and that it will do it after a finite number of iterations. The number of iterations is bounded by $1/\gamma^2$, where γ is the distance between the separating hyperplane and the closest datapoint to it.

Multi-Layer Perceptron

Example

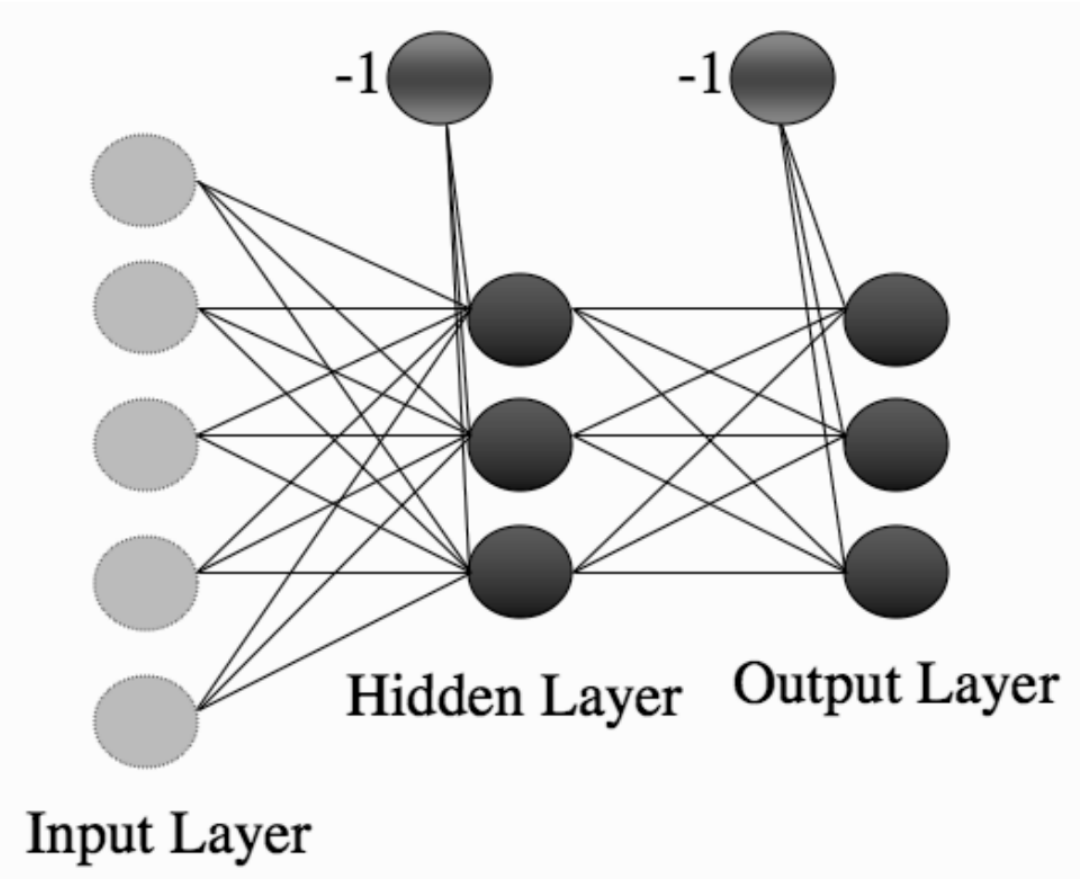

Figure: Multi-Layer Perceptron [Mar14]

MLP

XOR: An Example

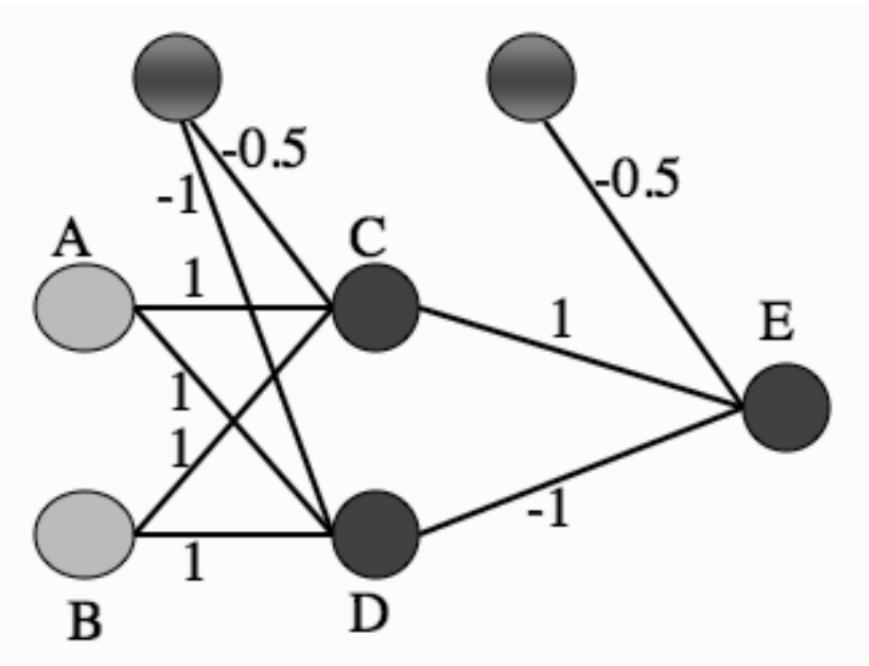

Figure: XOR using MLP [Mar14]

Going Forward

This works in the same way as a perceptron.

References I

[Mar14] Stephen Marsland. *Machine Learning, an algorithmic perspective*. CRC Press, 2014.