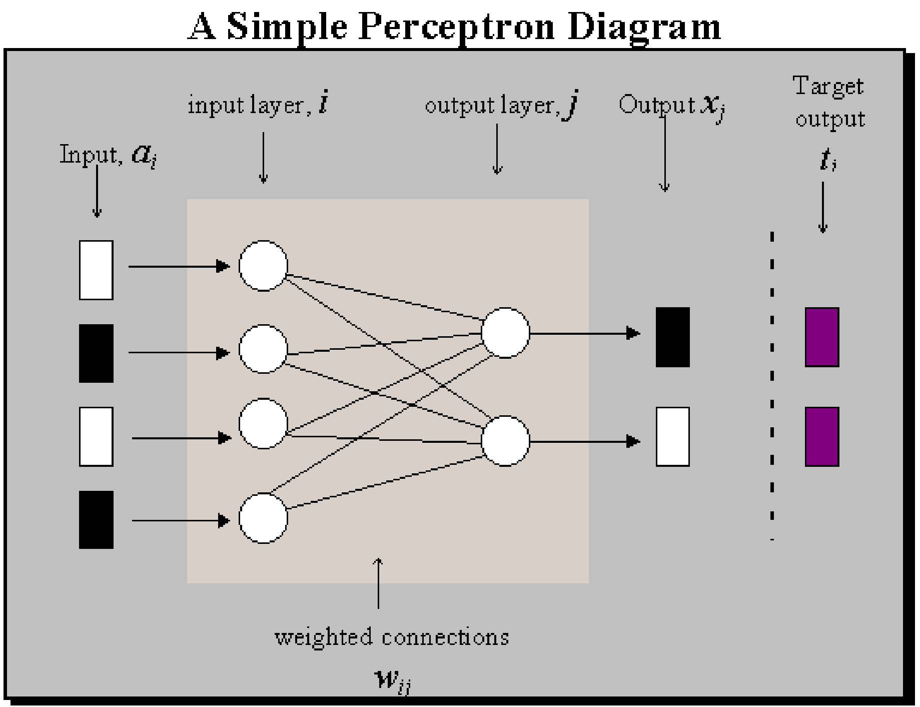
Perceptrons

History

Perceptons where originally investigated by Frank Rosenblatt in 1957 at the Cornell Aeronautical Laboratory at Cornell University. This was an attempt to understand human memory, learning and cognitive processes. His first attempt at this was the Mark I Perceptron, which could learn to recognize and identify optical patterns. Rosenblatt's work was based upon and also advanced the work by Warren McCulloch and Walter Pitts. McCulloch and Pitts were the first to consider the concept of neural networks. They developed the MP neuron which was based upon the human brains concept of a nerve firing an impulse to another nerve, only if a threshold value is exceeded. The MP neuron didn't have the ability to learn, therefore Rosenblatt tried to solve this problem by implementing Donald Hebb's idea that "when an axom of cell A is near enough to excite cell B and repeatedly, or persistently, takes part in firing it, some growth process or metabolic change takes place in one or both cells, such that Aís efficiency as one of the cells firing B is increased". This implied the idea of a network of neurons which could learn by adjusting the weights on connections between neurons. This is the idea which a perceptron work upon.

Single Layer Perceptron

So named because there is an input layer and an output layer, where each layer is fully connected to the other layer. There are no connections within the layers, just like in a complete bipartite graph. A perceptron is the simplest kind of neural network, and is also a feedforward neural network, meaning that there is no directed cycles. In this network, information only moves in one direction, straight from the input layer, through the hidden layer (only in MLP's) and to the output layer. The perceptron works by the input layer sending a signal to the output player through the weighted connections. Each node on the output layer sums up the incoming signal values, and if a threshold is exceeded, the node fires an output signal.



The output signals from each individual node can by summed up by:

Outputs are determined by:

$$S_j = \sum_{i=0}^{n} a_i w_{ij}$$

$$\text{If } S_j > \Theta \text{ then } x_j = 1$$
$$\text{If } S_j \leq \Theta \text{ then } x_j = 0$$

where Θ is a predetermined threshold value

In order to train the perceptron, the weighted connections between the input and output layers could be adjusted so the output would match the desired output. The training of the perceptron can be achieved by giving the input layer a set of predesigned values and checking the output to what was expected. If the output was different, the weighted connection could be altered so the output was closer to what was expected. The new weights for the connections were determined by an error correction formula given below.

Error Correction Formula:

$$w_{ij_{\text{new}}} = w_{ij_{\text{old}}} + C(t_j - x_j)a_i$$

C = learning rate

As can be seen, the new value for the connection is given by multiplying the difference between the actual output (x) and the target output (t) by a learning constant (c). If the input value (a) is a 1 then the connection value would be altered, otherwise if it was a 0, it has no bearing on the output and therefore wouldn't be changed. This training of the perceptron would continue until no improvements could be made, meaning the network has converged. When the convergence point has been met, the perceptron has either learned from the training and can be used in general to produce its own correct results, or it has failed and is deemed no working.

Perceptron networks have limitation though. Firstly, the output value of the perceptron can only be either true or false, meaning that it can only say for definite whether the input is what it is, and not how close it is. Also, a perceptron can only recognize linearly separable patterns. This means that any input pattern can be separated into two distinct classes by drawing a single line. The most famous example of a perceptron's inability to solve problems is that if the Boolean exclusive-or problem.

The linearly separable pattern problem which is the downfall of the single layer perceptron is somewhat solved by the introduction of back error propagation. This extends the single layer perceptron by implementing multiple layers of hidden networks, also know as multiple player perceptrons.

Sources:
http://www.cs.bgu.ac.il/~omri/Perceptron/
http://ei.cs.vt.edu/~history/Perceptrons.Estebon.html
http://www.cs.stir.ac.uk/courses/ITNP4B/lectures/kms/2-Perceptrons.pdf
http://en.wikipedia.org/wiki/Feedforward_neural_network