Multi-Layer Perceptrons

Perceptrons

- Late 50's to mid 60's – Rosenblatt's Perceptrons (Original paper: The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain, Psychological Review, 65:386-408)

- Original perceptron formulation used a threshold instead of a sigmoid:

$$g(a) = \begin{vmatrix} 1 \; if \; a > 0 \\ 0 \; if \; a \leq 0 \end{vmatrix}$$

- Learning rule:

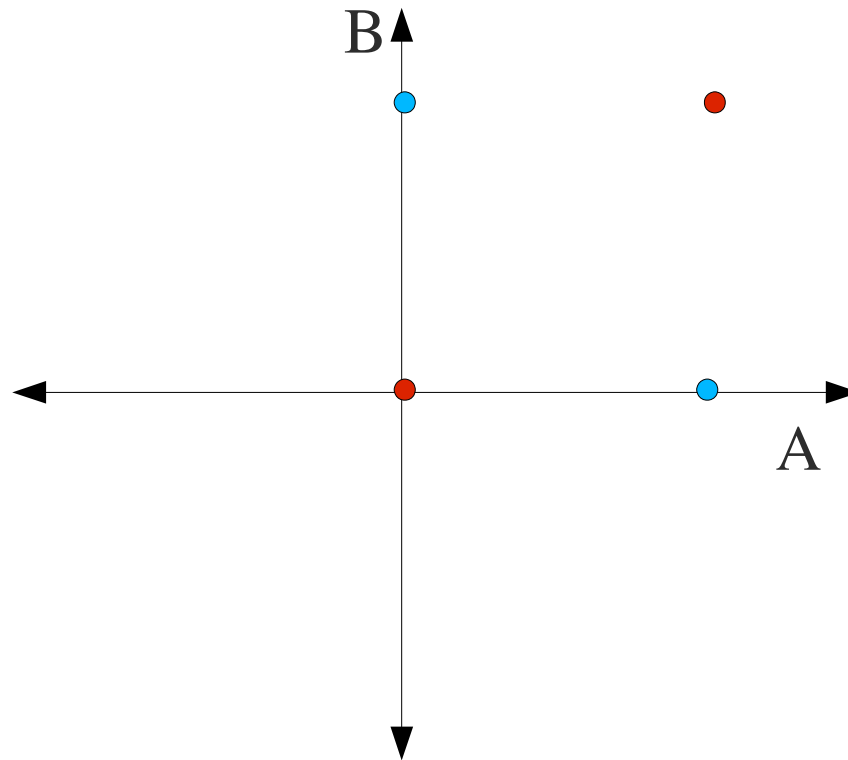$$w \leftarrow w + \alpha (t - g(w^T x)) x$$

The Rise and Fall of Perceptrons

- 1969 – Minsky and Papert write <u>Perceptrons</u>.
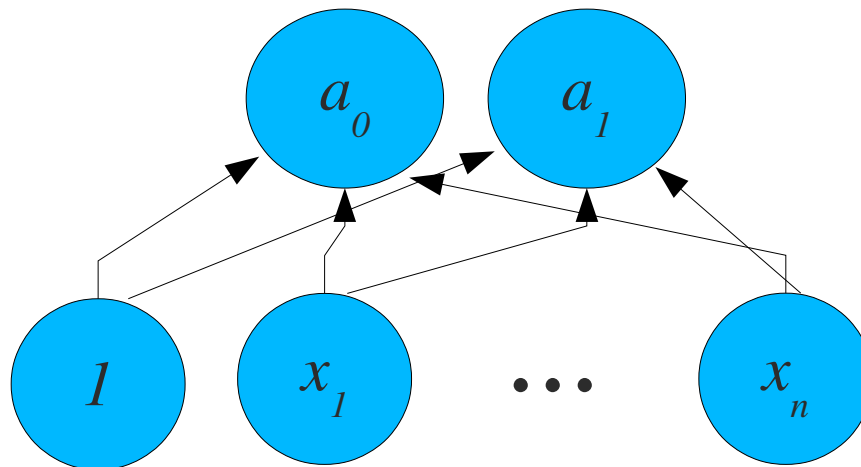 - Pretty much kills off neural network research.

The Problem...

- The perceptron only works if the classes are linearly separable.

- XOR is a problem:

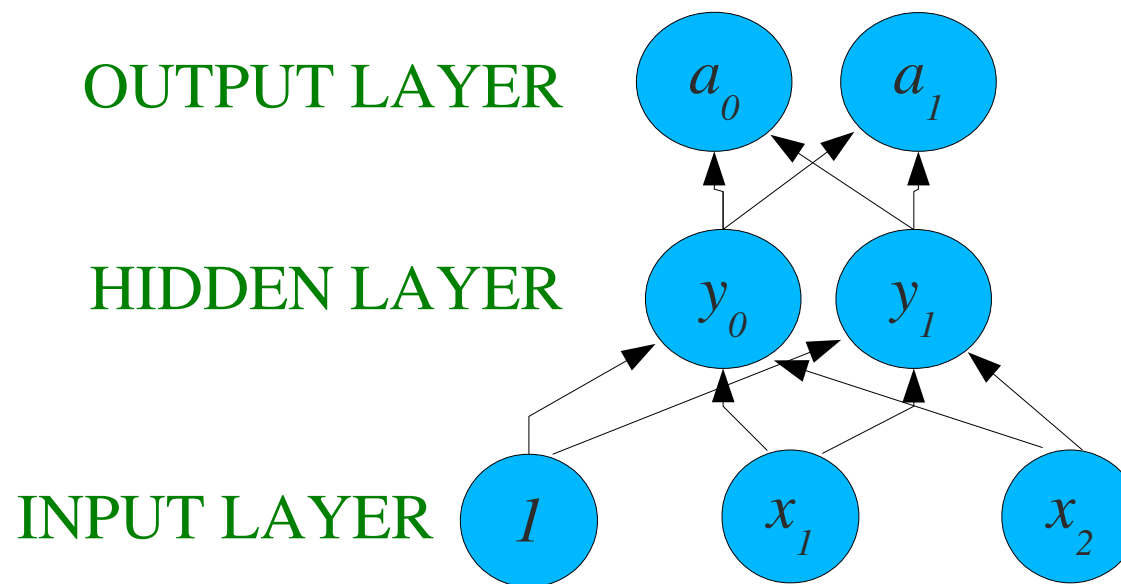| A	B	OUT
0	0	0
0	1	1
1	0	1
1	1	0

Multiple Output Units

- A network with M output units is nothing more than M independent perceptrons.

The Solution

- Multi-layer perceptrons can represent arbitrary functions.

OUTPUT LAYER a_0 a_1

HIDDEN LAYER y_0 y_1

INPUT LAYER 1 x_1 x_2

- We already know how to train the weights at the output layer – this is just a single layer perceptron.

- What about the weights at the hidden layer?

Resurgence of Neural Networks

- Rumelhart, D. E. and J. L. McClelland, Ed. (1986). <u>Parallel distributed processing: Explorations in the microstructure of cognition</u>.

- Backpropagation!

Backpropagation

- Activation at the output layer:

$$a_k = g\left(\sum_j W_{j,k} \ g\left(\sum_i V_{i,j} x_i\right)\right)$$

- here V and W are weight matrices. Units at the input layer are indexed with i, hidden with j and output with k.

- Error metric, assuming multiple output units:

$$E = \frac{1}{2}\sum_k (t_k - a_k)^2$$

- Now just compute $\dfrac{\partial E}{\partial W_{j,k}}$ and $\dfrac{\partial E}{\partial V_{i,j}}$.

Backprop Update Rules

- Let's define a new error term: $\delta_k = Err_k \times g'(in_k)$

- Where: $Err_k = (t_k - y_k) = (t_k - g(in_k))$

$$in_k = \sum_j W_{j,k} y_j$$

- So our perceptron learning rule becomes:

$$W_{j,k} \leftarrow W_{j,k} + \alpha \times \delta_k \times y_j$$

Training the Hidden Layer

- What should the error at the hidden layer be?

$$\delta_j = g'(in_j) \sum_k W_{j,k} \delta_k$$

- This is the backpropagation in backpropagation
- Now the learning rule at the hidden layer is:

$$V_{i,j} \leftarrow V_{i,j} + \alpha \times \delta_j \times x_i$$

The Backprop Learning Algorithm

- Set all weights to small random values
  - **For each training point:
    - FORWARD PASS – compute activations for each layer, starting from the input layer, and working to the output layer.
    - BACKWARD PASS – compute error signals, starting at the output layer and working to the input layer.
    - UPDATE WEIGHTS
  - If the average error is small enough terminate. Otherwise goto **.

Some Issues/Terminology

- Local Minima

- Epoch

- Stochastic Gradient Descent

- Momentum

- Overfitting

- Autoencoding

- Recurrent Neural Networks

Back to Biology

- Do real neurons do backpropagation?
- Probably not.
- Inspired by neural computation, not a model of neural computation.
- Issues:
  - Propagating errors.
  - Real valued signals.