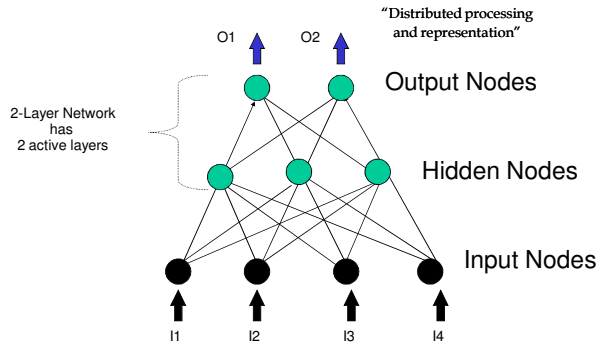
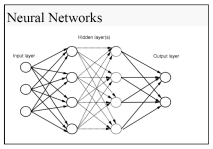
Multi-Layer Networks



N-layer Feed Forward Network



Layer 0 is input nodes

Layers 1 to N-1 are hidden nodes

Layer N is output nodes

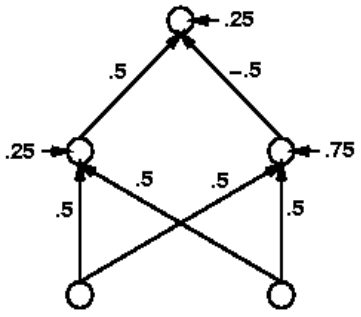
All nodes at any layer k are connected to all nodes at layer k+1

There are no cycles

Theorem:

Given an arbitrary number of hidden units, any Boolean function can be computed with a single hidden layer.

XOR Solution



Multi-Layer Networks

Behaviour of an artificial neural network to any particular input depends upon:

structure of each node (activation function)
structure of the network (architecture)
weights on each of the connections

.... these must be learned !

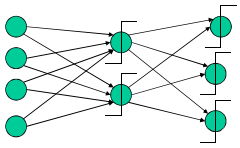
Multi-Layer Networks Built from Perceptron Units

Perceptrons are not able to learn certain concepts

Can only learn linearly separable functions

But they can be the basis for larger structures

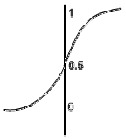
- Which can learn more sophisticated concepts
- Say that the networks have "perceptron units"



Problem With Perceptron Units

- Needs the output of a unit to be a differentiable function
- That is: The learning rule relies on minimizing the error. Finding minima by differentiating.
- Step functions aren't differentiable. They are not continuous
- Alternative threshold function are to be used
  - Must be differentiable
  - Must be similar to step function
- Sigmoid units used for backpropagation
  (There are other alternatives that may be used)

Sigmoid Units



Take in weighted sum of inputs, S

Then the out isoutput:

$$\sigma(S) = \frac{1}{1+e^{-S}}$$

Advantages:

- Looks very similar to the step function
- Is differentiable
- Derivative easily expressible in terms of σ itself:

$$\frac{d\sigma(S)}{dS} = \sigma(S)(1-\sigma(S))$$

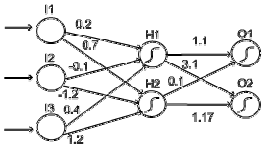
Example ANN with Sigmoid Units

Feed forward network

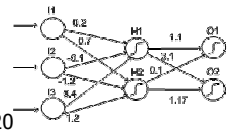
- Feed inputs in on the left, propagate numbers forward

Suppose we have this ANN

- With weights set arbitrary



Propagation of Example



With an example E:

Suppose the input to this ANN is 10, 30, 20

First calculate weighted sums to hidden layer:

S_{H1} = (0.2*10) + (-0.1*30) + (0.4*20) = 2-3+8 = 7

S_{H2} = (0.7*10) + (-1.2*30) + (1.2*20) = 7-36+24= -5

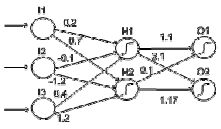
Next calculate the output from the hidden layer Using:

$\sigma(S) = 1/(1 + e^{-S})$

$\sigma(S_{H1}) = 1/(1 + e^{-7}) = 1/(1+0.000912) = 0.999$

$\sigma(S_{H2}) = 1/(1 + e^{5}) = 1/(1+148.4) = 0.0067$

Propagation of Example



Next calculate the weighted sums into the output layer:

S_{O1} = (1.1 * 0.999) + (0.1 * 0.0067) = 1.0996

S_{O2} = (3.1 * 0.999) + (1.17 * 0.0067) = 3.1047

Finally, calculate the output from the ANN

$\sigma(S_{O1}) = 1/(1+e^{-1.0996}) = 1/(1+0.333) = 0.750$

$\sigma(S_{O2}) = 1/(1+e^{-3.1047}) = 1/(1+0.045) = 0.957$

Output from O2 > output from O1

So, the ANN predicts category associated with O2

For the example input (10,30,20)