Introduction to Neural Networks

Natural language and ARtificial intelligence Group

Overview

- Apologies
- History Lesson
  - Perceptrons
  - Neural Networks
  - Backpropagation
- Theory

Ancient History

- Seminal paper published in 1943 by McCulloch and Pitts
- Outlined a classifier system based on a single neuron deemed a "perceptron"
- Strictly a mathematical abstraction
- Worked like a linear classifier by weight twiddling

Age of Enlightenment

- Late 1940s Hebb published "The Organization of Behavior"
- Further reinforced the idea of McCulloch and Pitts
- "What fires together wires together"
- Rosenblatt makes a two layer neural network in the 1950s

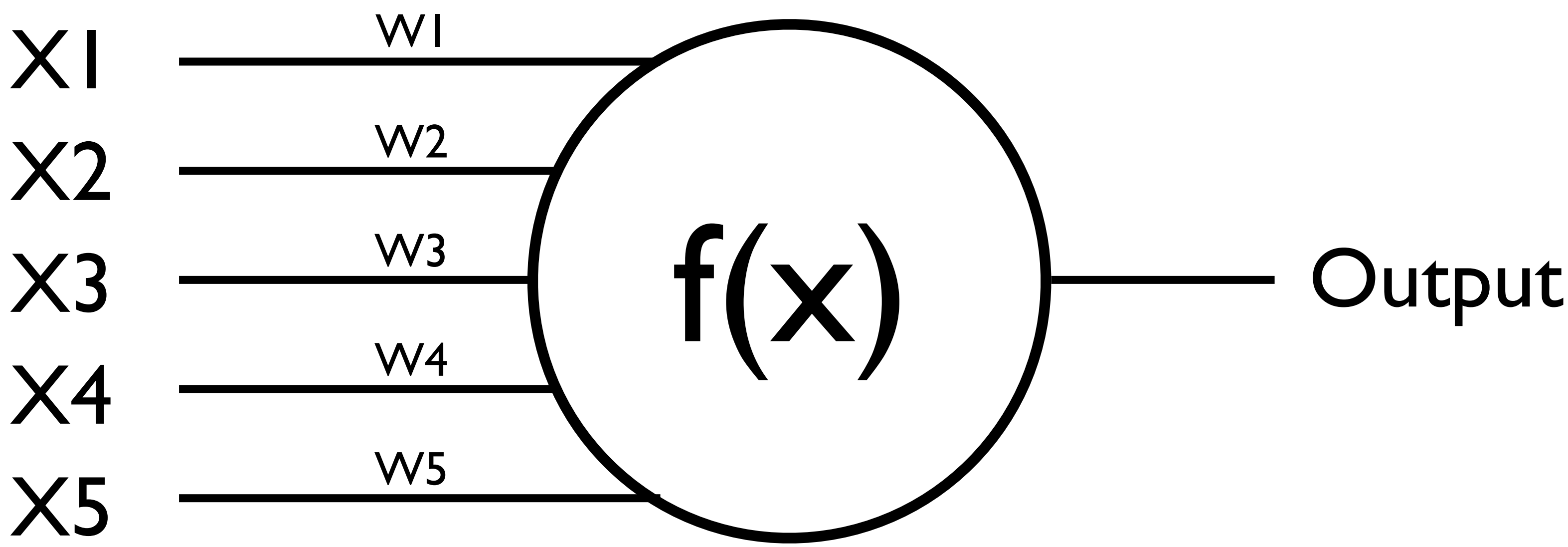
Dark Ages

- Neural networks were proven unable to reproduce all of boolean logic
- Failure to replicate XOR threw neural networks into the dark ages
- Backpropagation developed in 1969, but ignored by researchers until 1986 (said Norvig)

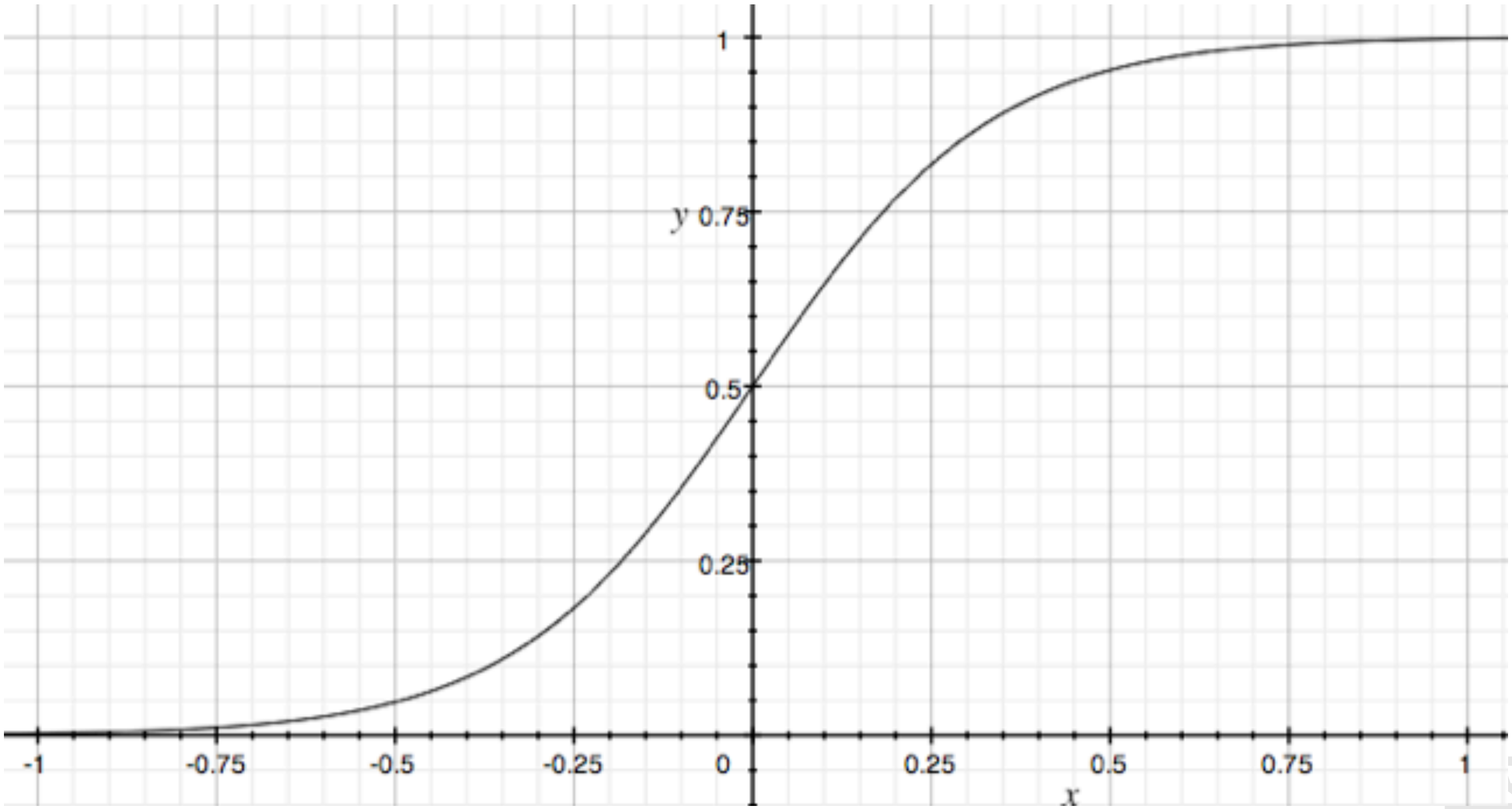
Modern Era (80s-Today)

- Backpropagation rediscovered and helps push neural networks back into the spotlight
- Recurrent neural networks are born
    - Echo State Machines
    - Hopfield Networks
    - Long-Short Term Memory

Perceptron



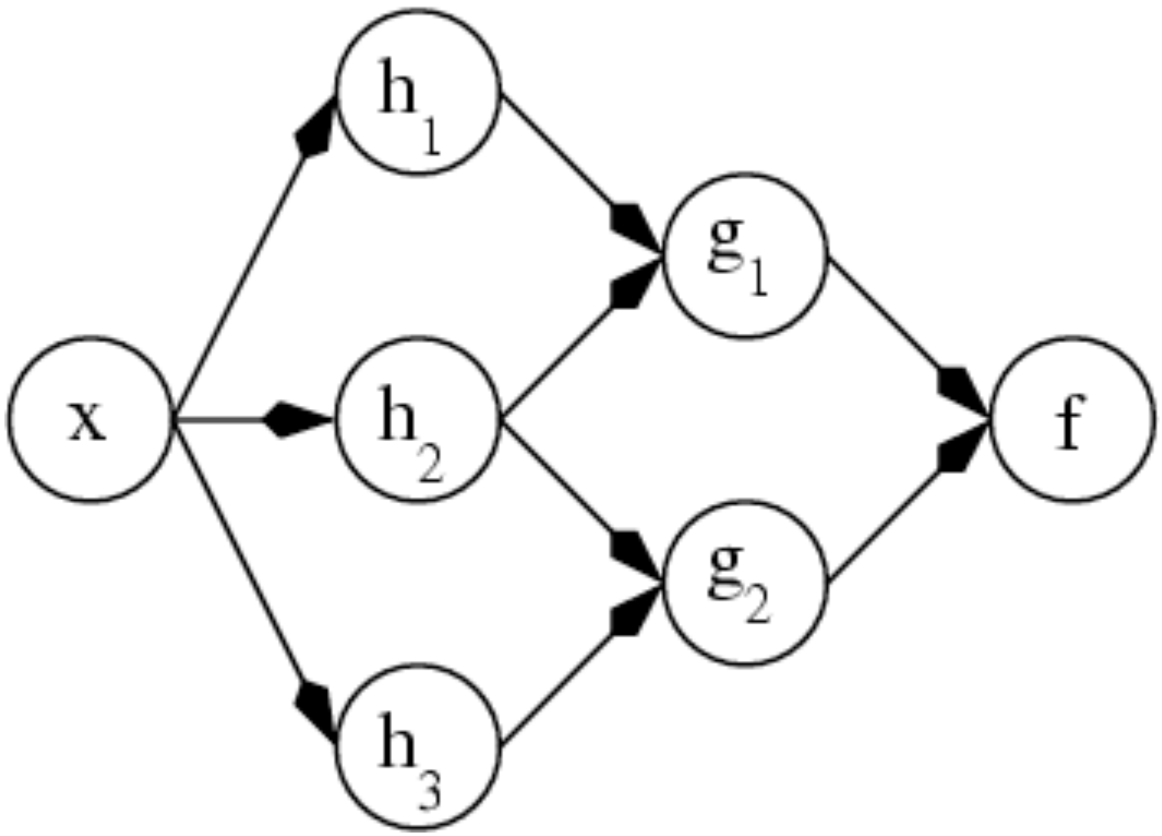
Activation Function



Threshold

~.95

Neural Network



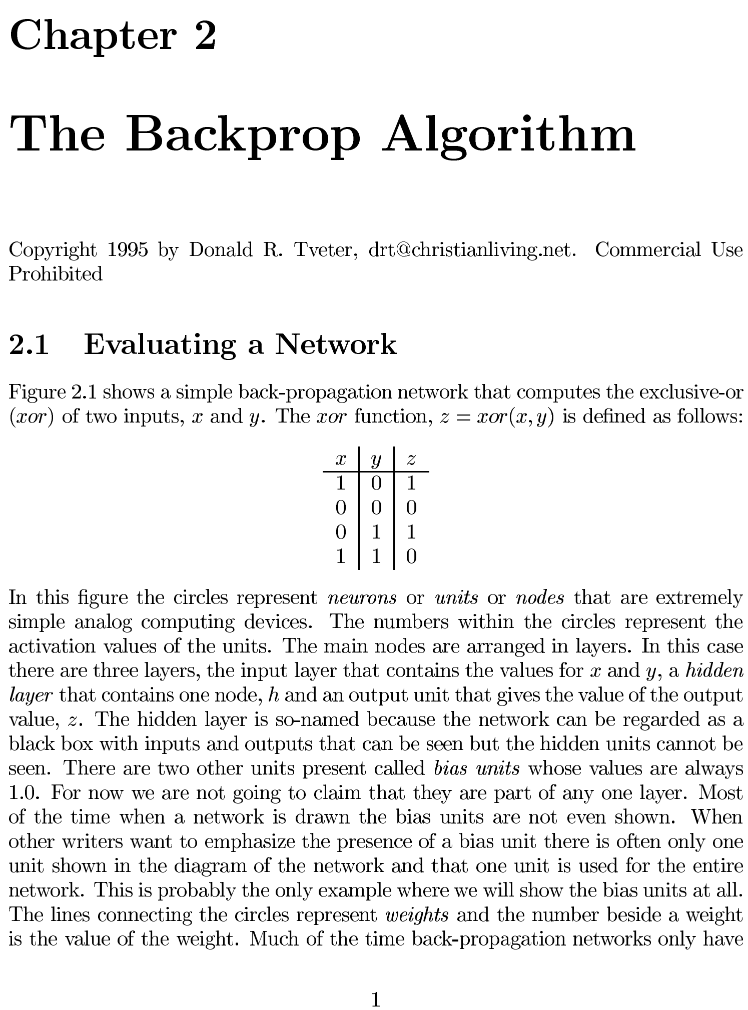
Traditional Networks

- 3+ layers: Input, Hidden Layer(s), Output
- Networks are traditionally feed-forward (i.e. no cycles on their graphs)
- No more than 5 hidden layers are ever used with backpropagation
- Bias neurons are often added to help with the generalized delta rule

Backpropagation

- Present training data
- Compare network output to desired output
- For each output neuron, calculate what the output should be and add or subtract this to the weights (delta rule)
- For each hidden input recursively figure out error levels and modify the weights accordingly (generalized delta rule)

No code... Sorry. Read this instead!


Chapter 2

The Backprop Algorithm

2.1 Evaluating a Network

Figure 2.1 shows a simple back-propagation network that computes the exclusive-or (xor) of two inputs, x and y. The xor function, $z = xor(x, y)$ is defined as follows:

x	y	z
1	0	1
0	0	0
0	1	1
1	1	0

In this figure the circles represent *neurons* or *units* or *nodes* that are extremely simple analog computing devices. The numbers within the circles represent the activation values of the units. The main nodes are arranged in layers. In this case there are three layers, the input layer that contains the values for x and y, a *hidden layer* that contains one node, h and an output unit that gives the value of the output value, z. The hidden layer is so-named because the network can be regarded as a black box with inputs and outputs that can be seen but the hidden units cannot be seen. There are two other units present called *bias units* whose values are always 1.0. For now we are not going to claim that they are part of any one layer. Most of the time when a network is drawn the bias units are not even shown. When other writers want to emphasize the presence of a bias unit there is often only one unit shown in the diagram of the network and that one unit is used for the entire network. This is probably the only example where we will show the bias units at all. The lines connecting the circles represent *weights* and the number beside a weight is the value of the weight. Much of the time back-propagation networks only have

1

Happy Birthday HacDC!

Love NARG