Artificial Intelligence Programming

Neural Networks

Chris Brooks

Department of Computer Science
University of San Francisco

23-1: Neural networks

- Much of what we've studied so far can be classified as **symbolic** AI.
- Focus on symbols and relations between them.
  - Search, logic, decision trees.
- The underlying assumption is that manipulation of symbols is the key requirement for intelligent behavior.
- Neural networks focus on **subsymbolic** behavior.
- Intelligent behavior emerges from the interaction of simple components.

23-2: Biology vs Computer Science

- In biological neurons, signals are received by dendrites and propagated to other neurons via the axon.
- Signaling and firing is very complex.
- Thought and behavior are produced through the interaction of thousands of neurons.

23-3: Biology vs Computer Science

- Computational neural networks are related to biological neural networks primarily by analogy
  - Computational neuroscience studies the modeling of biologically plausible neurons.
- AI researchers are often more interested in developing effective algorithms.
  - As with GAs, we draw upon ideas that are successful in nature and take the parts that are useful.

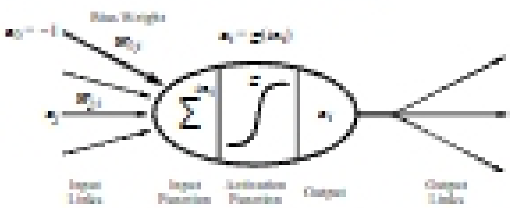
23-4: Computational Neural Networks

- Neural networks are composed of nodes.
- These nodes are connected by links
  - Abstraction of axons
- Each link has an associated weight that indicates the strength of the signal.
- Each node has a nonlinear activation function
  - Governs node's output as a function of the weighted sum of its inputs.

23-5: Appropriate tasks for neural learning

- Many attribute-value pairs.
- Real-valued inputs
- Real or discrete target value
- Noisy or error-containing data
- Long training time OK
- Fast evaluation of test cases needed
- Ability of humans to understand the learned hypothesis is not important.

23-7: Computational Neural Networks

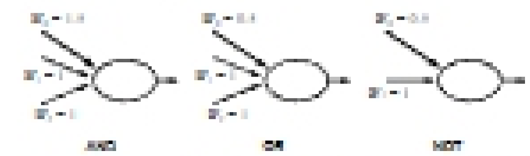


- Bias unit is used to control the *threshold* value
  - How strong the weighted input signal must be for the node to fire.

23-8: Activation functions

- Any nonlinear function can be used in principle.
- Two most common functions are:
  - Step function (threshold function) - Outputs 1 if input positive, zero otherwise.
  - Sigmoid/logistic function: $\frac{1}{1+e^{-x}}$
    - Continuously differentiable
    - Rapid change near threshold, gradual at extremes.

23-9: Examples



- Neural nets can easily be built to perform some standard logical operations using the threshold activation function.
- Change the threshold depending on the function needed

23-10: Types of nodes

- We can distinguish between three types of nodes:
  - Input nodes
  - Output nodes
  - Hidden nodes
- We can also distinguish between types of networks
  - Feed-forward networks: signals flow in one direction, no cycles.
  - Recurrent networks: Cycles in signal propagation
- We'll focus primarily on feedforward networks

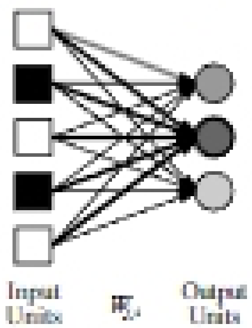
23-11: NNs as Function Approximators

- Feedforward NNs fall into a family of algorithms called *nonlinear function approximators*
- The output of a NN is a function of its inputs
- Nonlinear activation function allows the representation of complex functions.
- By adjusting weights, we change the function being represented
- NNs are often used to efficiently approximate complex functions from data.

23-12: Classification with Neural Networks

- NNs also perform classification very well.
- Map inputs into one or more outputs.
- Output range is split into discrete "classes"
- Very useful for learning tasks where "what to look for" is not known
  - Face recognition, handwriting recognition, driving a car

23-13: Perceptrons



- Single-layer networks (perceptrons) are the simplest form of NN.
- Easy to understand, but computationally limited.
- Each input unit is directly connected to one or more output units.

23-14: Perceptrons

- Output is thresholded weighted sum of the inputs.
- Threshold firing function used here.
- $o(x_1, \ldots, x_n) = 1$ if $w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n > 0$
- -1 otherwise

23-15: Representational power of perceptrons

- Output of net: $\sum_{j=0}^{n} W_j I_j > 0$
- Perceptrons are capable of representing any linearly separable function.
- Unfortunately, many common functions (XOR, parity) are not linearly separable.
- In the early days of AI, perceptrons were popular, due to the fact that their weights could be easily learned.

23-16: Perceptron Training Algorithm