Neural Networks

CPS 270
Ron Parr

Why Neural Networks?

- Maybe computers should be more brain-like:

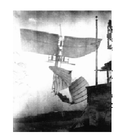
	Computers	Brains
Computational Units	10^9 gates/CPU	10^{11} neurons
Storage Units	10^{10} bits RAM 10^{13} bits HD	10^{11} neurons 10^{14} synapses
Cycle Time	10^{-9} S	10^{-3} S
Bandwidth	10^{10} bits/s*	10^{14} bits/s
Compute Power	10^{10} Ops/s	10^{14} Ops/s

Comments on Blue Gene

- Blue Gene: World's Fastest Supercomputer
- 360 Teraflops
- Currently at 131,000 processors
- 10^{13} -> 10^{14} Ops/s (brain level?)
- 16 TB memory (10^{13} bits)
- 14 Megawatts power ($1M/year in electricity)
- 2500 sq ft size (nice sized house)
- Pictures and other details:
- http://domino.research.ibm.com/comm/pr.nsf/pages/rsc.bluegene_2004.html

Neural Network Motivation

- Individual neurons are slow, boring
- Brains succeed by using massive parallelism
- Idea: Copy what works



- Raises many issues:
  - Is the computational metaphor suited to the computational hardware?
  - How do we know if we are copying the important part?
  - Are we aiming too low?

Artificial Neural Networks

- Develop *abstraction* of function of actual neurons
- Simulate large, massively parallel artificial neural networks on conventional computers
- Some have tried to build the hardware too
- Try to approximate human learning, robustness to noise, robustness to damage, etc.

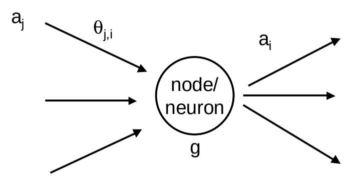
Use of neural networks

- Trained to pronounce English
  - Training set: Sliding window over text, sounds
  - 95% accuracy on training set
  - 78% accuracy on test set
- Trained to recognize handwritten digits
  - >99% accuracy
- Trained to drive
  (Pomerleau's no-hands across America)

Neural Network Lore

- Neural nets have been adopted with an almost religious fervor within the AI community - several times
- Often ascribed near magical powers by people, usually those who know the least about computation or brains
- For most AI people, magic is gone, but neural nets remain extremely interesting and useful mathematical objects

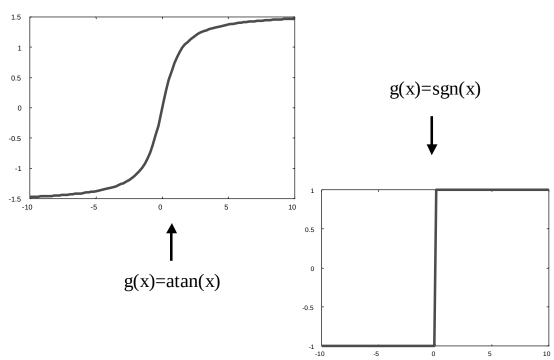
Artificial Neurons



$$a_i = g(\sum_j w_{j,i} a_j)$$

g can be any function, but usually a smoothed step function

Threshold Functions



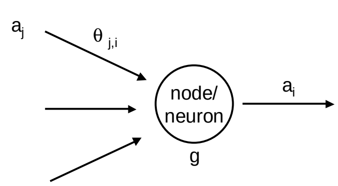
Network Architectures

- Cyclic vs. Acyclic
  - Cyclic is tricky, but more biologically plausible
    - Hard to analyze in general
    - May not be stable
    - Need to assume latches to avoid race conditions
  - Hopfield nets: special type of cyclic net useful for associative memory
- Single layer (perceptron)
- Multiple layer

Feedforward Networks

- We consider acyclic networks
- One or more computational layers
- Entire network can be viewed as computing a complex non-linear function
- Typical uses in learning:
  - Classification (usually involving complex patterns)
  - General continuous function approximation

Perceptron



For now, we assume that g is a simple step function (sgn)
Assume only 1 neuron and one output.

Perceptron Learning

- We are given a set of inputs $A_1 \ldots A_n$
- $T_1 \ldots T_n$ is a set of target outputs +1/-1
- θ is our set of weights
- Net(A_i, θ)=output of perceptron given input A_i and weights W.
- Error(A_i, θ) = T_i-Net(A_i, θ)
- Goal: Pick θ to optimize:

$$\min_{\theta} \sum_i error(A_i, \theta)$$

Update Rule

Repeat until convergence:

$$\forall i \forall j : \theta_j \leftarrow \theta_j + \alpha a_j Error(A_i, \theta)$$

"Learning Rate"

- i iterates over samples
- j iterates over weights

http://neuron.eng.wayne.edu/java/Perceptron/New38.html

Perceptron Learning Properties

- Good news:
  - If there exists a set of weights that will correctly classify every example, the perceptron learning rule will find it
- Bad news:
  - Perceptrons can represent only a small class of functions, "linearly separable," functions

Linearly Separable Functions

What is a perceptron really doing?

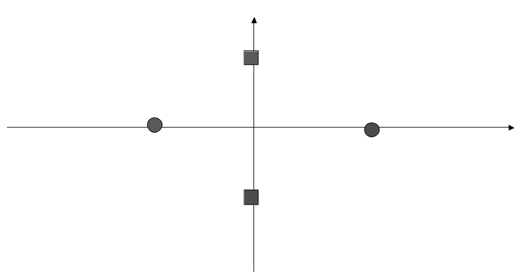
It checks if a linear combination of the inputs is greater than a threshold.

$$\theta_1 a_1 + \theta_2 a_2 \ldots \theta_n a_n > 0?$$

Perceptron asks: What side of a hyperplane does A lie on?

Q: How can we change from >0 to >C for arbitrary C?

Visualizing Linearly Separable Functions



Is red linearly separable from green?
Are the circles linearly separable from the squares?

Observations

- Linear separability is fairly weak
- We have other tricks:
  - Functions that are not linearly separable in one space, may be linearly separable in another space
  - If we engineer our inputs to our neural network, then we change the space in which we are constructing linear separators
  - Every function has a linear separator (in some space)
- Perhaps other network architectures will help

Multilayer Networks

- Once people realized how simple perceptrons were, they lost interest in neural networks for a while
- Multilayer networks turn out to be much more expressive (with a smoothed step function)
  - Use sigmoid, e.g., atanh($\theta^T x$)
  - With 2 layers, can represent any continuous function
  - With 3 layers, can represent many discontinuous functions
- Tricky part: How to adjust the weights

Smoothing Things Out

- Consider single-layer case first
- Idea: Do gradient descent on a smooth error function
- Error function is sum of squared errors

$$E = 0.5\sum_i error(X^{(i)}, \theta)^2$$

- i iterates over samples
- j iterates over weights

$$\frac{\partial E}{\partial \theta_j} = \sum_i error(X^{(i)}, \theta)\frac{\partial g(\sum_k \theta_k x_k^{(i)})}{\partial \theta_j}$$

input k for sample i

$$= \sum_i error(X^{(i)}, \theta) g'(\sum_k \theta_k x_k^{(i)}) x_j^{(i)}$$

Multilayer Case

- Gradient calculation, parameter update have recursive formulation
- Decomposes into:
  - Local message passing
  - No transcendentals
- Highly parallelizable
- Biologically plausible(?)
- Leads to celebrated *backpropagation* algorithm

Back-prop Issues

- Backprop = gradient descent on error function
- Function is nonlinear (= powerful)
- Function is nonlinear (= local minima)
- Big nets:
  - Many parameters
    - Many optima
    - Slow gradient descent
  - Biological plausibility ≠ Electronic plausibility
- Many NN experts became experts in numerical analysis (by necessity)

Neural Nets in Practice

- Many applications for pattern recognition tasks
- Very powerful representation
  - Can overfit
  - Can fail to fit with too many parameters, poor features
- Very widely deployed AI technology, but
  - Few open research questions
  - Connection to biology still uncertain
  - Results are hard to interpret
- “Second best way to solve any problem”
  - Can do just about anything w/enough twiddling
  - Now third or fourth to SVMs, boosting, and ???