



Bayes Nets

- Some helpful rules:
 - $P(A|B) = [P(B|A) * P(A)] / P(B)$
 - $P(A|K) = P(A|R) * P(R|K) + P(A|\sim R) * P(\sim R|K)$
 - Where R is between A and K in a network

RIT Computer Science Dept.



Questions

- How should we recognize the contents of an image?
 - What about a noisy image?
 - What about an occluded image?
 - What about an image in shadows?
- Is the symbol hypothesis the way to go?

RIT Computer Science Dept.



We Want

- Classification/Pattern Recognition/Memory Recall
- Prediction
- Robustness
- Generalization
- Abstraction
- Highly Parallel?
- Learning: supervised and unsupervised

RIT Computer Science Dept.



Neural Networks?

“Conventional training methods for multilayer perceptrons (“backprop” nets) can be interpreted in statistical terms as variations on maximum likelihood estimation. The idea is to find a single set of weights for the network that maximize the fit to the training data, perhaps modified by some sort of weight penalty to prevent overfitting.”

– Radford Neal

RIT Computer Science Dept.



Some Applications of neural networks

- Alvin (the neural network that learns to drive a van from camera inputs).
- NETtalk: a network that learns to pronounce English text.
- Recognizing hand-written zip codes.
- Lots of applications in financial time series analysis.

RIT Computer Science Dept.



NETtalk (Sejnowski & Rosenberg, 1987)

- The task is to learn to pronounce English text from examples.
- Training data is 1024 words from a side-by-side English/phoneme source.
- Input: 7 consecutive characters from written text presented in a moving window that scans text.
- Output: phoneme code giving the pronunciation of the letter at the center of the input window.
- Network topology: 7x29 inputs (26 chars + punctuation marks), 80 hidden units and 26 output units (phoneme code). Sigmoid units in hidden and output layer.

RIT Computer Science Dept.



NETtalk (contd.)

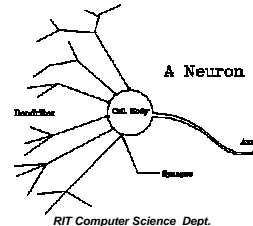
- Training protocol: 95% accuracy on training set after 50 epochs of training by full gradient descent. 78% accuracy on a set-aside test set.
- Comparison against Dectalk (a rule based expert system): Dectalk performs better; it represents a decade of analysis by linguists. NETtalk learns from examples alone and was constructed with little knowledge of the task.

RIT Computer Science Dept.



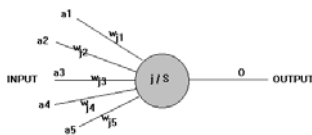
Biological Inspiration

- Neural nets are biologically inspired, but don't operate like real neurons!



In the Beginning

- The McCulloch-Pitts neuron (1943)



RIT Computer Science Dept.



Thresholds



Step function

step(x) = 1, if $x \geq \text{threshold}$
 0, if $x < \text{threshold}$
 (in picture above, threshold = 0)



Sign function

sign(x) = +1, if $x \geq 0$
 -1, if $x < 0$



Logistic function

sigmoid(x) = $1/(1+e^{-x})$

- Various functions are used for thresholds:
 - Sign and a Step functions
 - Logistic or sigmoid function: s-shaped and continuous
 - Hyperbolic tangent (tanh): s-shaped, but symmetric about the origin (continuous)

RIT Computer Science Dept.



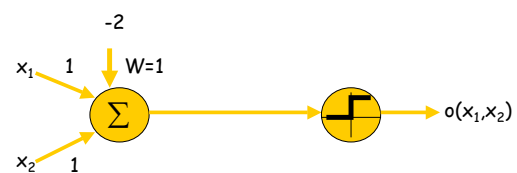
What You Can Do

- In 1943, McCollough and Pitts showed that a synchronous assembly of such neurons is a universal computing machine. That is, any Boolean function can be implemented with threshold (step function) units.

RIT Computer Science Dept.



Implementing AND



$$o(x_1, x_2) = 1 \text{ if } -2 + x_1 + x_2 \geq 0$$

$$= -1 \text{ otherwise}$$

RIT Computer Science Dept.

Implementing OR

$o(x_1, x_2) = 1$ if $-1 + x_1 + x_2 \geq 0$
 $= -1$ otherwise

RIT Computer Science Dept.

Implementing NOT

$o(x_1) = 1$ if $0.5 - x_1 > 0$
 $= 0$ otherwise

RIT Computer Science Dept.

Implementing more complex Boolean functions

RIT Computer Science Dept.

Non-biological?

- Artificial Neural Nets don't account for:
 - Time delays in the system
 - Firing frequency and synchronization

RIT Computer Science Dept.

And Then There Were...

- Perceptrons (1958)

Output (motor neurons)
 Synaptic Connections
 Input (sensor neurons)

RIT Computer Science Dept.

Feed-forward Networks

- Perceptrons are examples of a feed-forward network
 - They are a directed graph
 - There are no cycles in the graph
 - There is no internal state other than the weights

RIT Computer Science Dept.



How it works

- The value of an output is: $O_i = \text{step}(\sum_j W_{ij} I_j)$
- Learning occurs when the actual output is modified to be more similar to the desired output

RIT Computer Science Dept.



The Original Learning Rule

- The original perceptron learning rule:
 1. If output neuron O_i incorrectly produced a 1, then we need to decrease W for i . For each incoming synapse to O_i , set $W_{ij} = W_{ij} - I_j$
 2. If output neuron O_i incorrectly produced a 0, then we need to increase W for i . For each incoming synapse to O_i , set $W_{ij} = W_{ij} + I_j$
 3. If output neuron O_i got the right value, then make no changes to its weights.

RIT Computer Science Dept.



Better Learning Rule

- The learning rule was a special case of:

$$\Delta W_{ij} = \alpha(C_i - O_i)I_j$$

- C_i is the correct output value for neuron i . The alpha value is the learning rate (so we don't overshoot and oscillate around the value)

RIT Computer Science Dept.



The Overall Training Procedure

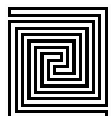
1. Pick an input/expected output vector pair at random.
2. Present the input vector pair to the input neurons.
3. Read the network's output vector.
4. Using the learning rule, modify each weight according to the difference between output vector and the expected output vector.
5. Go to #1.

RIT Computer Science Dept.

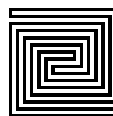


What Can't Perceptron's Do?

- XOR
- Distinguish on the basis of connectivity such figures as:



One Connected Spiral Region

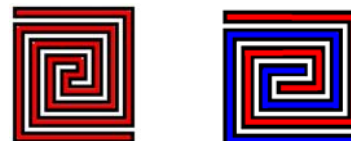


Two Separate Connected Spiral Regions

RIT Computer Science Dept.



Spiral Regions

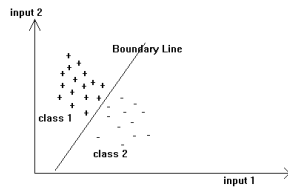


RIT Computer Science Dept.



Linearly Separable

- Perceptrons can only learn linearly separable functions

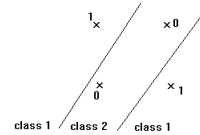


RIT Computer Science Dept.



XOR Problem

- There is no place where the xor problem can be linearly separated!

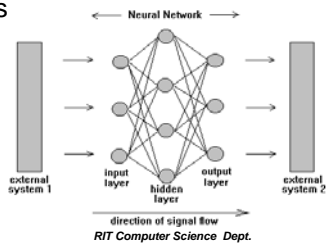


RIT Computer Science Dept.



New Architectures Emerged

- New neural networks from the '80's used an input, output, and 1+ hidden layers



RIT Computer Science Dept.



Description

- Minsky and Papert's description of neural nets with hidden units:
 - GAMBA PERCEPTRON: A number of linear threshold systems have their outputs connected to the inputs of a linear threshold system. Thus we have a linear threshold function of many linear threshold functions.

RIT Computer Science Dept.



What Killed the Field of Neural Nets...

- Minsky and Papert then stated:
 - "Virtually nothing is known about the computational capabilities of this latter kind of machine. We believe that it can do little more than can a low order perceptron. (This, in turn, would mean, roughly, that although they could recognize (sp) some relations between the points of a picture, they could not handle relations between such relations to any significant extent.) That we cannot understand mathematically the Gamba perceptron very well is, we feel, symptomatic of the early state of development of elementary computational theories."

RIT Computer Science Dept.



Multiple Layers

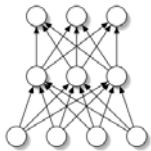
- A 2-layer network can learn any continuous function.
- A three layer neural network can learn any function.
- We will deal with 2-layer networks. But...
 - How do we train them?

RIT Computer Science Dept.



Notation and Output

- Variables we'll use
- The connection to an output neuron O_i from hidden layer neuron H_j is represented as W_{ij}



O: Output Layer

W matrix of synapses

H: Hidden Layer

V matrix of synapses

I: Input layer

$$O_i = \sigma\left(\sum_j W_{ij} H_j\right)$$

$$H_j = \sigma\left(\sum_k V_{jk} I_k\right)$$

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

RIT Computer Science Dept.



Initial Weights

- Perceptron: 0
- NN with Hidden Layer: small random values

RIT Computer Science Dept.



Training

- The rule we use for modifying the weights is known as the **delta rule**, because it changes each weight according to how much say it had in the final outcome (the delta, or partial derivative of the output with respect to the weight).

$$\Delta W_{ij} = \alpha (C_i - O_i) O_i (1 - O_i) H_j$$

$$\Delta V_{jk} = \alpha \sum_i (C_i - O_i) O_i (1 - O_i) W_{ij} H_j (1 - H_j) I_k$$

RIT Computer Science Dept.



What This Means

$$\Delta W_{ij} = \alpha (C_i - O_i) O_i (1 - O_i) H_j$$

$$\Delta V_{jk} = \alpha \sum_i (C_i - O_i) O_i (1 - O_i) W_{ij} H_j (1 - H_j) I_k$$

- The rule is applied to all weights at the same time.
- The alpha value is the learning rate
- Notice the similarity between the learning rule for W and that of the perceptron

$$\Delta W_{ij} = \alpha (C_i - O_i) I_j$$

RIT Computer Science Dept.



Error is Distributed

- Each hidden node is responsible for some fraction of the error in each of the output nodes. This fraction equals the strength of the connection (weight) between the hidden node and the output node.

RIT Computer Science Dept.



The Training Procedure

- Same as the perceptron, but since it operates on continuous values, it will never get the EXACT output
- We just want the output to converge to a correct output within some limit.
- Backpropagation won't always converge: it may get caught in a suboptimal solution, because it is a greedy algorithm

RIT Computer Science Dept.



The Learning Rate

- If it's too high, the we won't converge, but will overshoot
- If it's low we'll take forever to learn
- It's normal to start out with a higher learning rate and lower it slowly to help with convergence

RIT Computer Science Dept.



Momentum

- Sometimes we multiply w_{ij} by a momentum factor α . This allows us to use a high learning rate, but prevent the oscillatory behavior that can sometimes result from a high learning rate.

$$w_{ij} \leftarrow w_{ij} + \eta \delta_j x_{ij}$$

Multiply by momentum α

RIT Computer Science Dept.



The Hidden Layer

- If you provide too many nodes in the hidden layer, it will learn every input at one node
- If you provide too few, then it can't model the data
- When you're between the two extreme's the network will generalize to a certain extent

RIT Computer Science Dept.



More on backpropagation

- Performs gradient descent over the entire network weight vector.
- Will find a local, not necessarily global error minimum.
- Minimizes error over training set; need to guard against overfitting just as with decision tree learning.
- Training takes thousands of iterations (epochs) --- slow!

RIT Computer Science Dept.



When neural nets are appropriate for learning problems

- Instances are represented by attribute-value pairs.
 - Pre-processing required: Continuous input values to be scaled in [0-1] range, and discrete values need to be converted to Boolean features.
- Training examples are noisy.
- Long training times are acceptable.
- Human understandability of learned function is unimportant.
 - However, there is work on converting NNs to rules.

RIT Computer Science Dept.



Network topology

- Designing network topology is an art.
- We can learn the network topology using genetic algorithms. But using GAs is very cpu-intensive. An alternative that people use is hill-climbing.

RIT Computer Science Dept.