COSC 522 – Machine Learning

Lecture 13 – Backpropagation (BP) and Multi-Layer Perceptron (MLP)

Hairong Qi, Gonzalez Family Professor
Electrical Engineering and Computer Science
University of Tennessee, Knoxville
https://www.eecs.utk.edu/people/hairong-qi/
Email: hqi@utk.edu

Course Website: http://web.eecs.utk.edu/~hqi/cosc522/

Roadmap

- Supervised learning
  - Classification
    - Maximum Posterior Probability (MPP): For a given x, if $P(w_1|x) > P(w_2|x)$, then x belongs to class 1, otherwise 2.
      - Parametric Learning
        - Three cases
        - Estimate Gaussian parameters using MLE
      - Nonparametric Learning
        - Parzon window (fixed window size)
        - K-Nearest Neighbor (variable window size)
    - Neural Network
  - Regression (linear regression with nonlinear basis functions)
    - Neural Network
- Unsupervised learning
  - Non-probabilistic approaches
    - kmeans, wta
  - Hierarchical approaches
    - Agglomerative clustering
  - Neural Network
- Supporting preprocessing techniques
  - Dimensionality Reduction
    - Supervised linear (FLD)
    - Unsupervised linear (PCA)
    - Unsupervised nonlinear (t-SNE)
- Supporting postprocessing techniques
  - Classifier Fusion
  - Performance Evaluation
- Optimization techniques
  - Gradient Descent (GD)

Questions

- Differences between feedback and feedforward neural networks
- Limitations of perceptron
- Why go deeper?
- MLP structure
- MLP cost function and optimization method (BP)
- The importance of the threshold function
- Relationship between BPNN and MPP
- Various aspects of practical improvements of BPNN

Types of NN

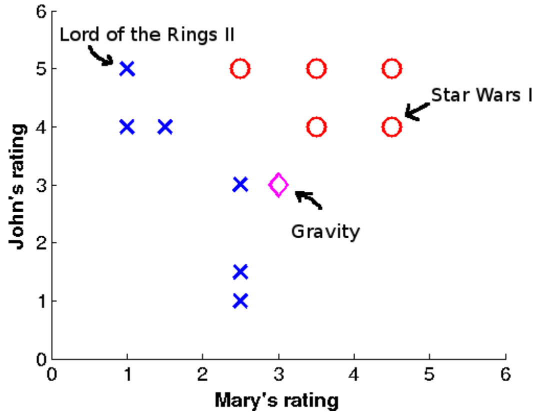
- Recurrent (feedback during operation)
  - Hopfield
  - Kohonen
  - Associative memory
- Feedforward
  - No feedback during operation or testing (only during determination of weights or training)
  - Perceptron
  - Multilayer perceptron and backpropagation

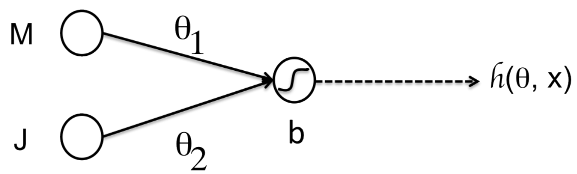
Limitations of Perceptron

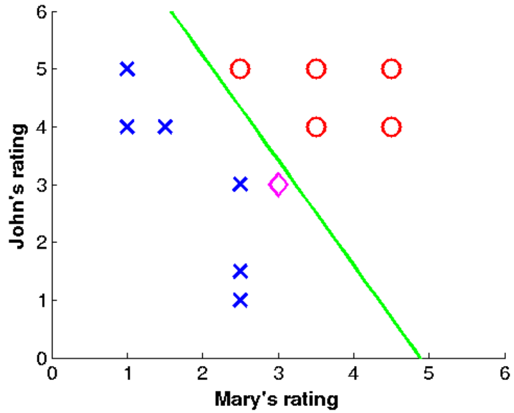
- The output only has two values (1 or 0)
- Can only classify samples which are linearly separable (straight line or straight plane)
- Single layer: can only train AND, OR, NOT
- Can’t train a network functions like XOR

Why deeper?

Movie name	Mary's rating	John's rating	I like?
Lord of the Rings II	1	5	No
...	...	...	...
Star Wars I	4.5	4	Yes
Gravity	3	3	?

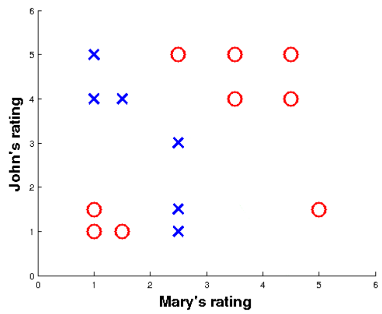


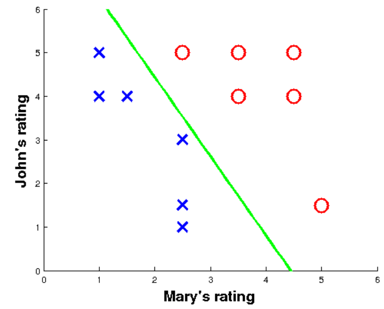


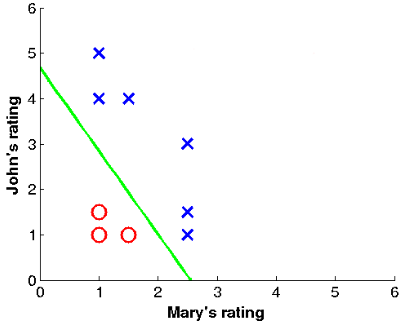


http://ai.stanford.edu/~quocle/tutorial2.pdf

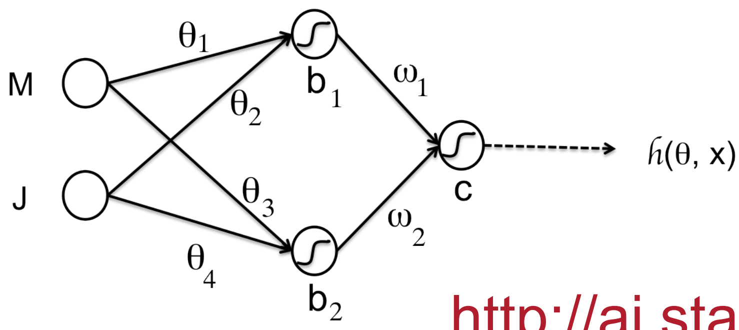
Why deeper?

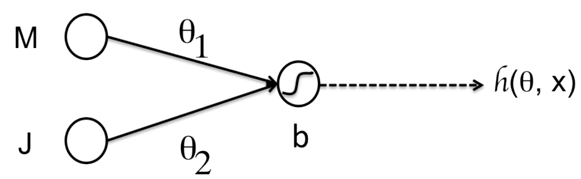






Movie name	Output by decision function h_1	Output by decision function h_2	Susan likes?
Lord of the Rings II	$h_1(x^{(1)})$	$h_2(x^{(2)})$	No
...	...	...	...
Star Wars I	$h_1(x^{(n)})$	$h_2(x^{(n)})$	Yes
Gravity	$h_1(x^{(n+1)})$	$h_2(x^{(n+1)})$	?



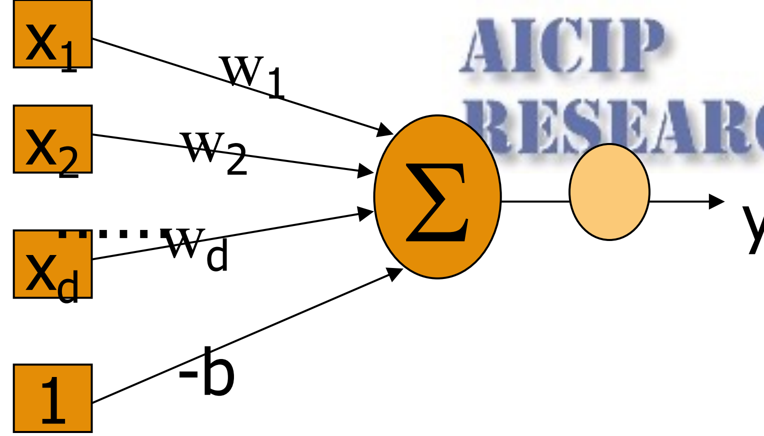


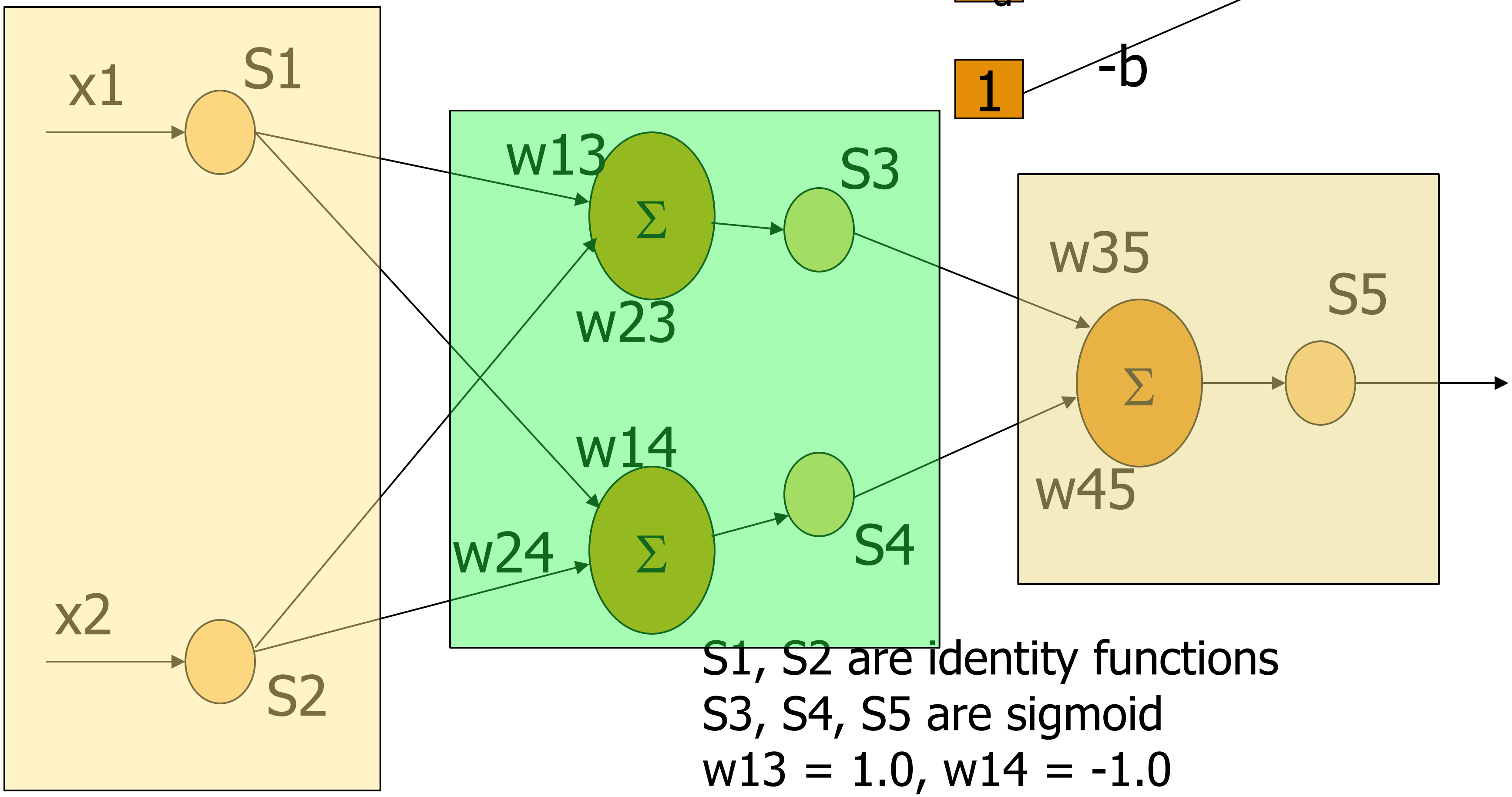
http://ai.stanford.edu/~quocle/tutorial2.pdf

Questions

- Differences between feedback and feedforward neural networks
- Limitations of perceptron
- Why go deeper?
- MLP structure
- MLP cost function and optimization method (BP)
- The importance of the threshold function
- Relationship between BPNN and MPP
- Various aspects of practical improvements of BPNN

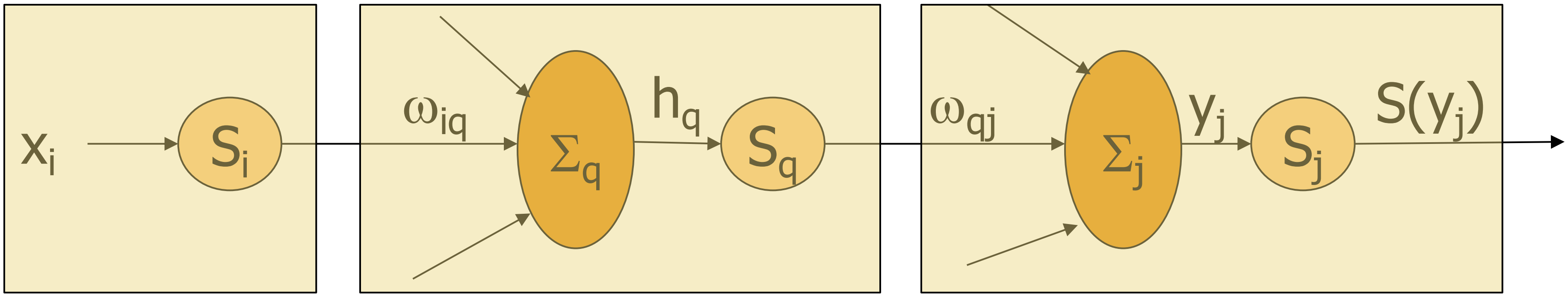
XOR (3-layer NN)





S1, S2 are identity functions
S3, S4, S5 are sigmoid
w13 = 1.0, w14 = -1.0
w24 = 1.0, w23 = -1.0
w35 = 0.11, w45 = -0.1
The input takes on only –1 and 1

MLP – 3-Layer Network



$$E = \frac{1}{2}\sum_j \left(T_j - S\left(y_j\right)\right)^2$$

Choose a set of initial ω_{st}

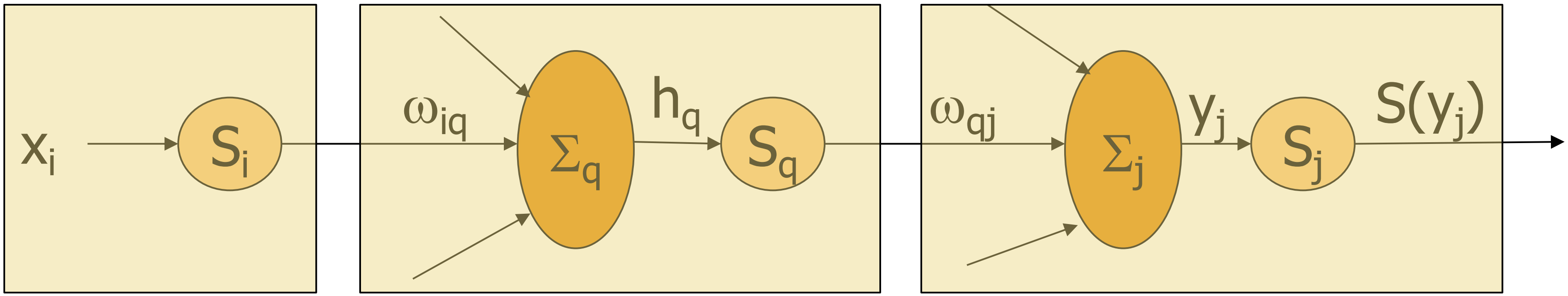
$$\omega_{st}^{k+1} = \omega_{st}^{k} - c^k \frac{\partial E^k}{\partial \omega_{st}^{k}}$$

ω_{st} is the weight connecting input *s* at neuron *t*

The problem is essentially "how to choose weight ω to minimize the error between the expected output and the actual output"

The basic idea behind BP is gradient descent

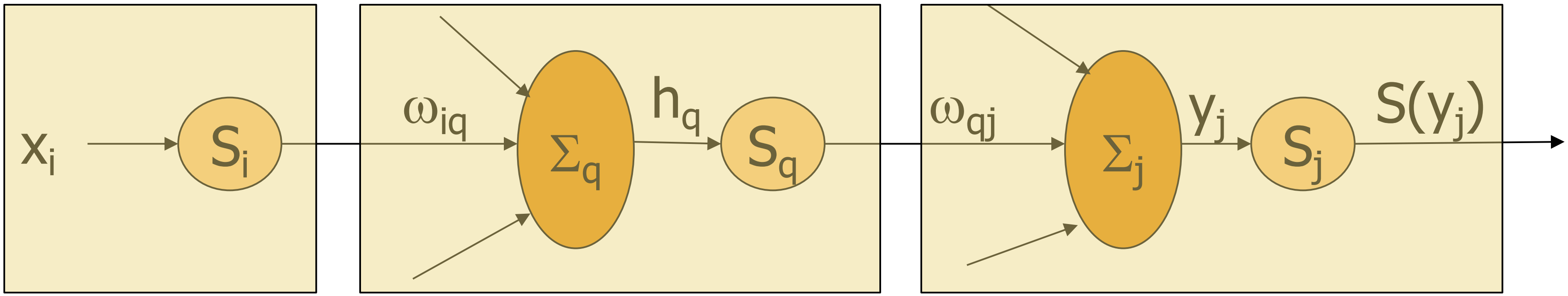
Exercise



$$y_j = \sum_q S_q(h_q)\omega_{qj} \Rightarrow \frac{\partial y_j}{\partial S_q} = \omega_{qj} \quad \text{and} \quad \frac{\partial y_j}{\partial \omega_{qj}} = S_q(h_q)$$

$$h_q = \sum_i x_i \omega_{iq} \Rightarrow \frac{\partial h_q}{\partial x_i} = \omega_{iq} \quad \text{and} \quad \frac{\partial h_q}{\partial \omega_{iq}} = x_i$$

The Derivative – Chain Rule



$$\Delta\omega_{qj} = -\frac{\partial E}{\partial \omega_{qj}} = -\frac{\partial E}{\partial S_j}\frac{\partial S_j}{\partial y_j}\frac{\partial y_j}{\partial \omega_{qj}}$$

$$= -\left(T_j - S_j\right)\left(S'_j\right)\left(S_q\left(h_q\right)\right)$$

$$\Delta\omega_{iq} = -\frac{\partial E}{\partial \omega_{iq}} = \left[\sum_j \frac{\partial E}{\partial S_j}\frac{\partial S_j}{\partial y_j}\frac{\partial y_j}{\partial S_q}\right]\frac{\partial S_q}{\partial h_q}\frac{\partial h_q}{\partial \omega_{iq}}$$

$$= \left[\sum_j \left(T_j - S_j\right)\left(S_j'\right)\left(\omega_{qj}\right)\right]\left(S_q'\right)\left(x_i\right)$$

Threshold Function

- Traditional threshold function as proposed by McCulloch-Pitts is binary function
- The importance of differentiable
- A threshold-like but differentiable form for S (25 years)
- The sigmoid

$$S(x) = \frac{1}{1 + \exp(-x)}$$

BP vs. MPP

$$E(\omega) = \sum_{\mathbf{x}} [g_k(\mathbf{x};\mathbf{w}) - T_k]^2 = \sum_{\mathbf{x}\in\omega_k} [g_k(\mathbf{x};\mathbf{w}) - 1]^2 + \sum_{\mathbf{x}\notin\omega_k} [g_k(\mathbf{x};\mathbf{w}) - 0]^2$$

$$= n\left\{ \frac{n_k}{n} \frac{1}{n_k} \sum_{\mathbf{x}\in\omega_k} [g_k(\mathbf{x};\mathbf{w}) - 1]^2 + \frac{n-n_k}{n} \frac{1}{n-n_k} \sum_{\mathbf{x}\notin\omega_k} [g_k(\mathbf{x};\mathbf{w})]^2 \right\}$$

$$\lim_{n\to\infty} \frac{1}{n} E(\mathbf{w}) = P(\omega_k) \int [g_k(\mathbf{x};\mathbf{w}) - 1]^2 p(\mathbf{x} \mid \omega_k) d\mathbf{x} + P(\omega_{i\neq k}) \int g_k^2(\mathbf{x};\mathbf{w}) p(\mathbf{x} \mid \mathbf{w}_{i\neq k}) d\mathbf{x}$$

$$= \int [g_k{}^2(\mathbf{x};\mathbf{w}) - 2g_k(\mathbf{x};\mathbf{w}) + 1] p(\mathbf{x},\omega_k) d\mathbf{x} + \int g_k^2(\mathbf{x};\mathbf{w}) p(\mathbf{x},\mathbf{w}_{i\neq k}) d\mathbf{x}$$

$$= \int g_k{}^2(\mathbf{x};\mathbf{w}) p(\mathbf{x}) d\mathbf{x} - 2\int g_k(\mathbf{x};\mathbf{w}) p(\mathbf{x},\omega_k) d\mathbf{x} + \int p(\mathbf{x},\omega_k) d\mathbf{x}$$

$$= \int [g_k(\mathbf{x};\mathbf{w}) - P(\omega_k \mid \mathbf{x})]^2 p(\mathbf{x}) d\mathbf{x} + C$$

Questions

- Differences between feedback and feedforward neural networks
- Limitations of perceptron
- Why go deeper?
- MLP structure
- MLP cost function and optimization method (BP)
- The importance of the threshold function
- Relationship between BPNN and MPP
- Various aspects of practical improvements of BPNN

Activation (Threshold) Function

- The signum function

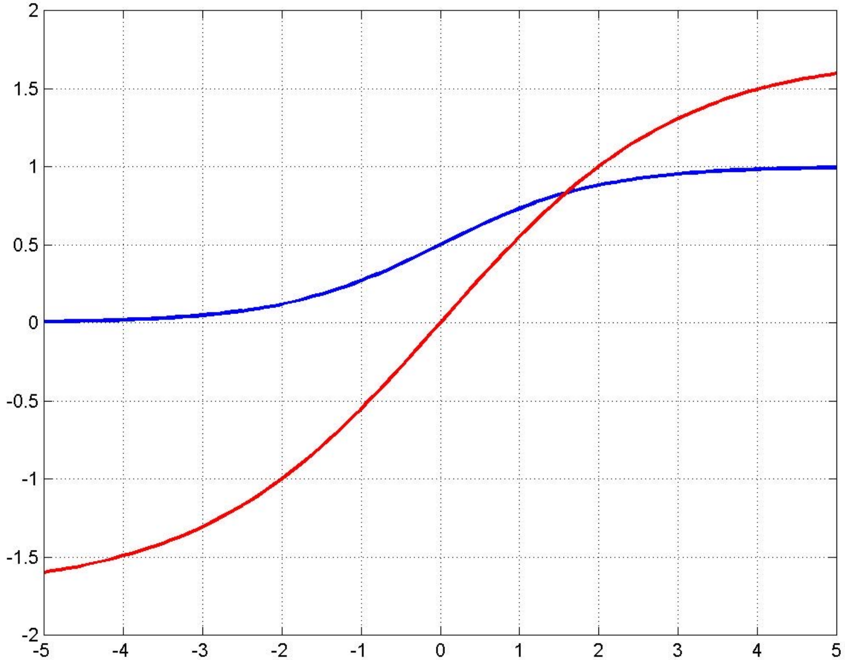
$$S(x) = signum(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ -1 & \text{if } x < 0 \end{cases}$$

- The sigmoid function
  - Nonlinear
  - Saturate
  - Continuity and smoothness
  - Monotonicity (so S' (x) > 0)

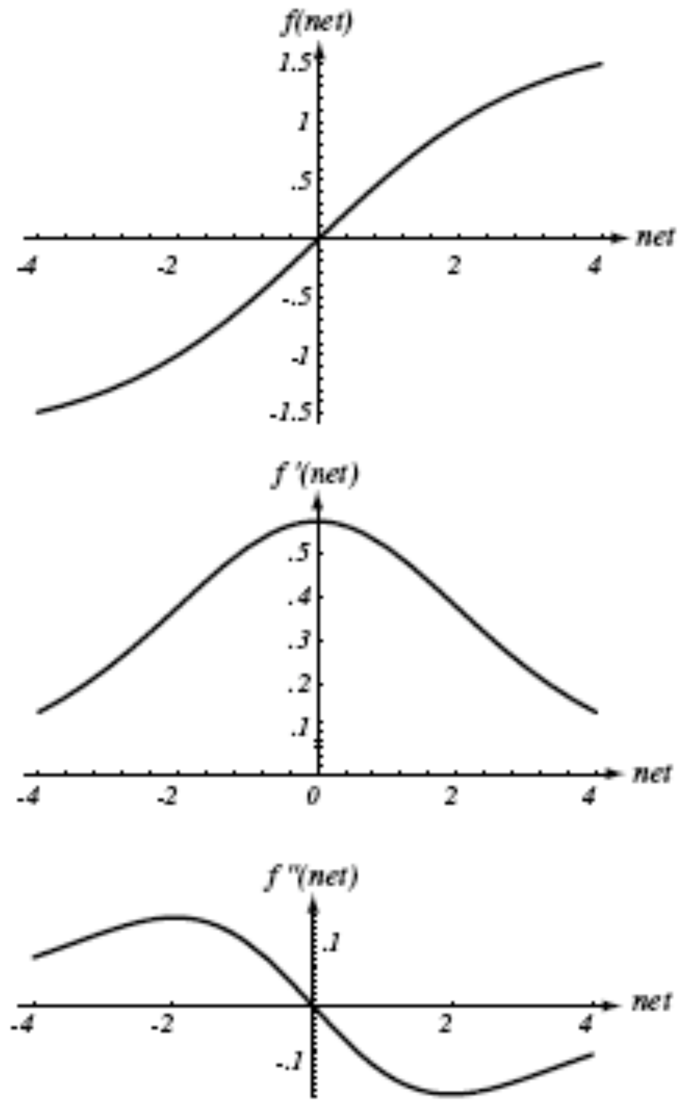
$$S(x) = sigmoid(x) = \frac{1}{1 + \exp(-x)}$$

- Improved
  - Centered at zero
  - Antisymmetric (odd) – leads to faster learning
  - a = 1.716, b = 2/3, to keep S' (0) -> 1, the linear range is –1<x<1, and the extrema of S'' (x) occur roughly at x -> 2

$$S(x) = sigmoid(x) = \frac{2a}{1 + \exp(-bx)} - a$$



2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-5
-4
-3
-2
-1
0
1
2
3
4
5



f(net)
1.5
1
.5
-.5
-1
-1.5
net
-4
-2
2
4
f '(net)
.5
.4
.3
.2
.1
net
0
f ''(net)
.1
-.1
net

Data Standardization

- Problem in the units of the inputs
  - Different units cause magnitude of difference
  - Same units cause magnitude of difference
- Standardization – scaling input
  - Shift the input pattern
    - The average over the training set of each feature is zero
  - Scale the full data set
    - Have the same variance in each feature component (around 1.0)

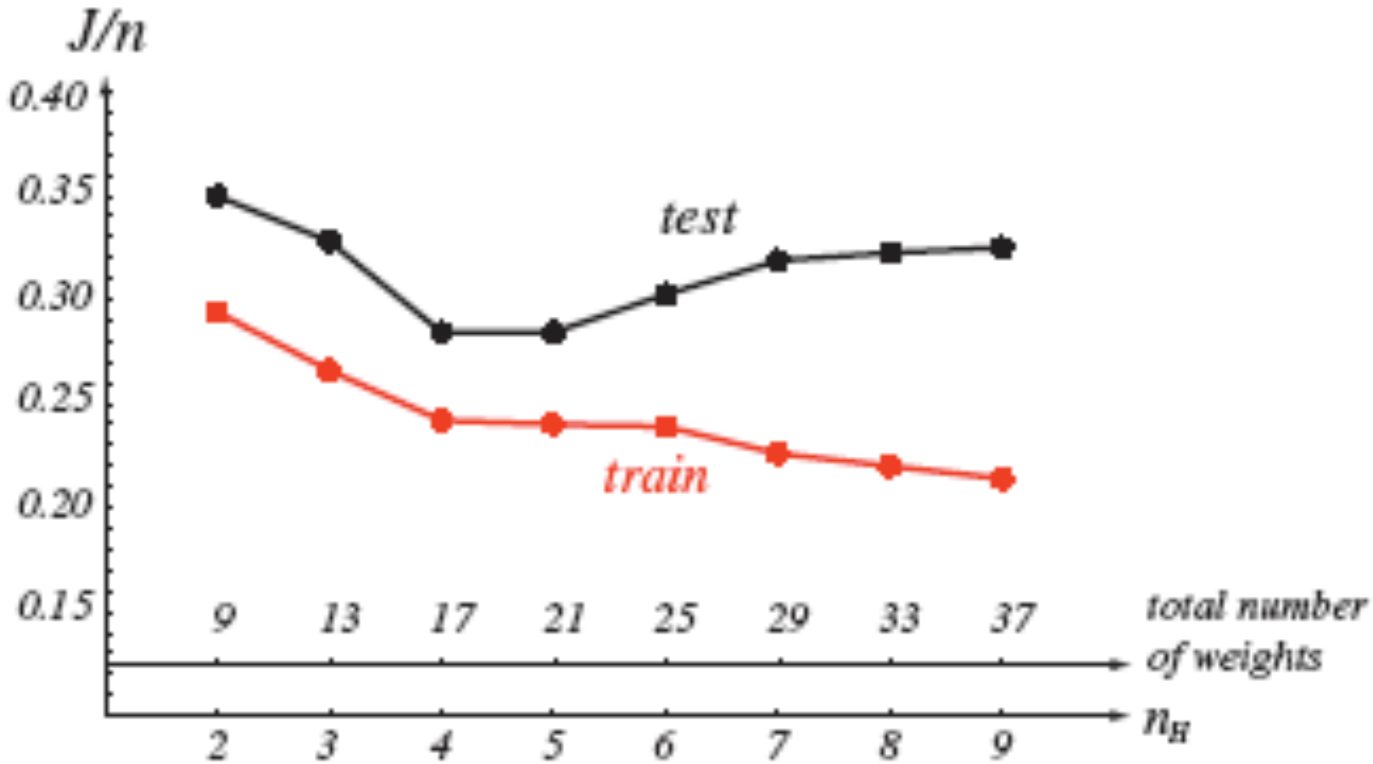
Target Values (output)

- Instead of one-of-c (c is the number of classes), we use +1/-1
  - +1 indicates target category
  - -1 indicates non-target category
- For faster convergence

Number of Hidden Layers

- The number of hidden layers governs the expressive power of the network, and also the complexity of the decision boundary
- More hidden layers -> higher expressive power -> better tuned to the particular training set -> poor performance on the testing set
- Rule of thumb
  - Choose the number of weights to be roughly n/10, where n is the total number of samples in the training set
  - Start with a “large” number of hidden units, and “decay”, prune, or eliminate weights

Number of Hidden Layers



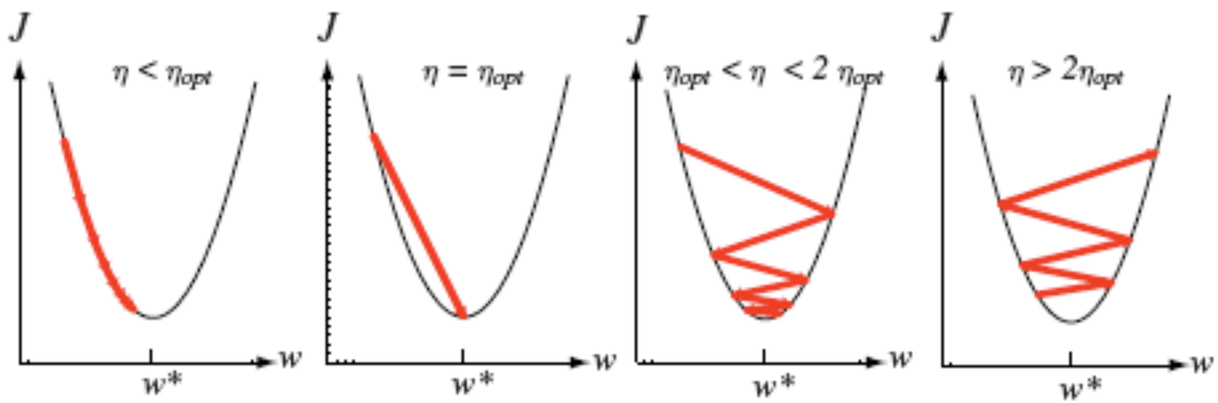
Initializing Weight

- Can't start with zero
- Fast and uniform learning
  - All weights reach their final equilibrium values at about the same time
  - Choose weights randomly from a uniform distribution to help ensure uniform learning
  - Equal negative and positive weights
  - Set the weights such that the integration value at a hidden unit is in the range of –1 and +1
  - Input-to-hidden weights: (-1/sqrt(d), 1/sqrt(d))
  - Hidden-to-output weights: (-1/sqrt(n_H), 1/sqrt(n_H)), n_H is the number of connected units

Learning Rate

$$c_{opt} = \left(\frac{\partial^2 MSE}{\partial \omega^2} \right)^{-1}$$

- The optimal learning rate
  - Calculate the 2nd derivative of the objective function with respect to each weight
  - Set the optimal learning rate separately for each weight
  - A learning rate of 0.1 is often adequate

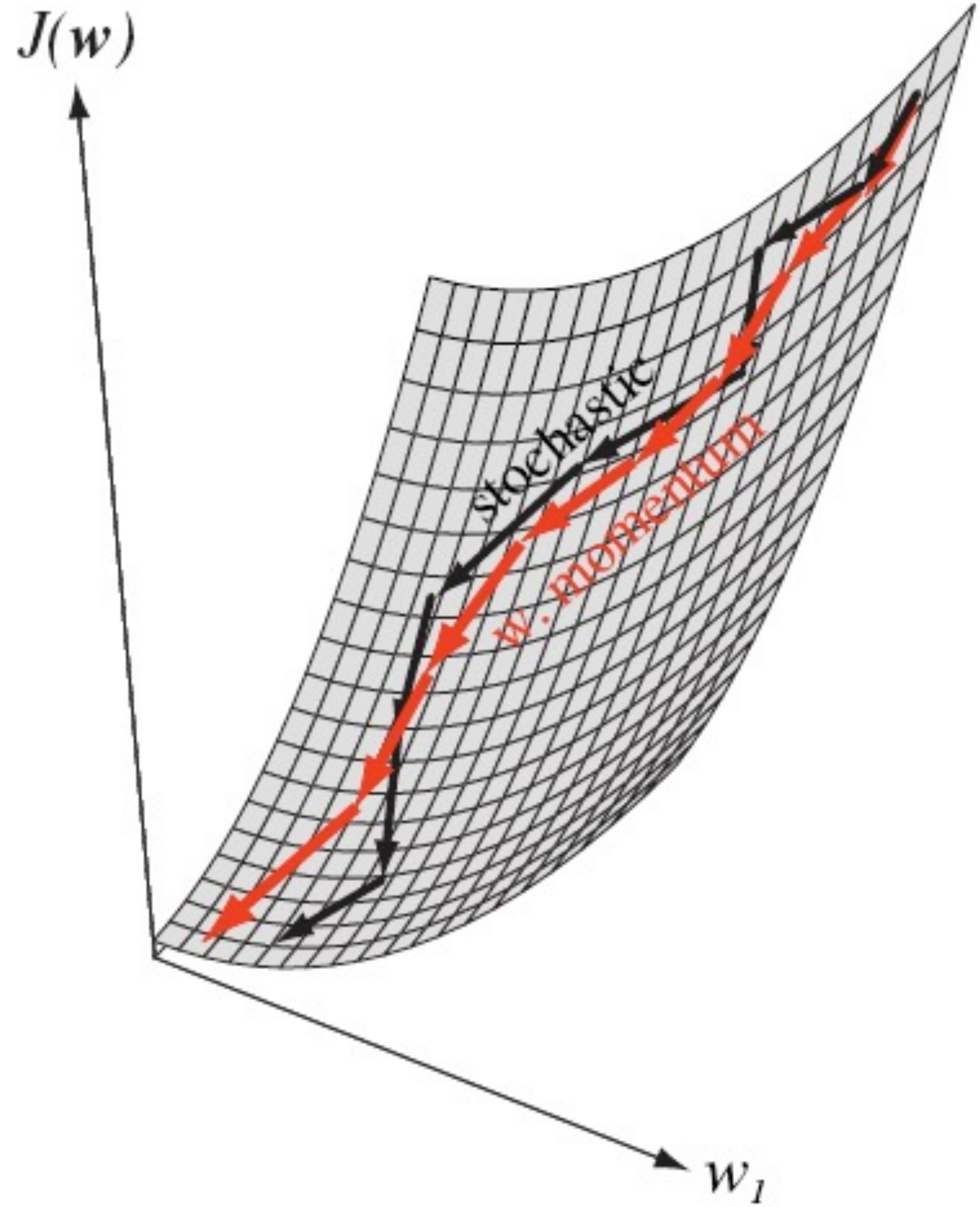


Plateaus or Flat Surface in S'

- Plateaus
  - Regions where the derivative $\dfrac{\partial E}{\partial \omega_{st}}$ is very small
  - When the sigmoid function saturates
- Momentum
  - Allows the network to learn more quickly when plateaus in the error surface exist

$$\omega_{st}^{k+1} = \omega_{st}^{k} - c^{k} \frac{\partial E^{k}}{\partial \omega_{st}^{k}}$$

$$\omega_{st}^{k+1} = \omega_{st}^{k} + (1-\alpha^{k})\Delta\omega_{bp}^{k} + \alpha^{k}(\omega_{st}^{k} - \omega_{st}^{k-1})$$


J(w)
stochastic
w. momentum
w1

Weight Decay

- Should almost always lead to improved performance

$$\omega^{new} = \omega^{old}(1-\varepsilon)$$

Batch Training vs. On-line Training

- Batch training
  - Add up the weight changes for all the training patterns and apply them in one go
  - GD
- On-line training
  - Update all the weights immediately after processing each training pattern
  - Not true GD but faster learning rate

Further Discussions

AICIP RESEARCH

- How to draw the decision boundary of BPNN?
- How to set the range of valid output
  - 0-0.5 and 0.5-1?
  - 0-0.2 and 0.8-1?
  - 0.1-0.2 and 0.8-0.9?
- The importance of having symmetric initial input