

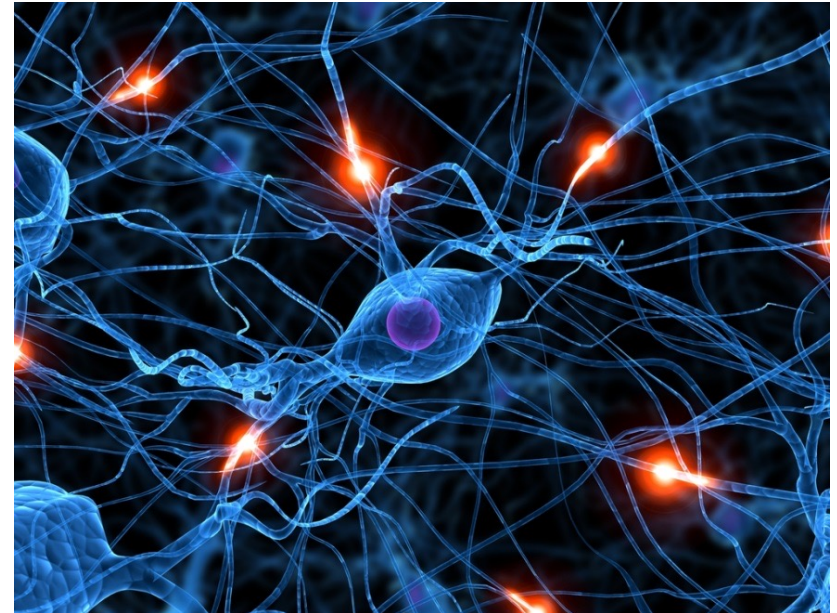
430.457

Introduction to Intelligent Systems

Prof. Songhwai Oh
ECE, SNU

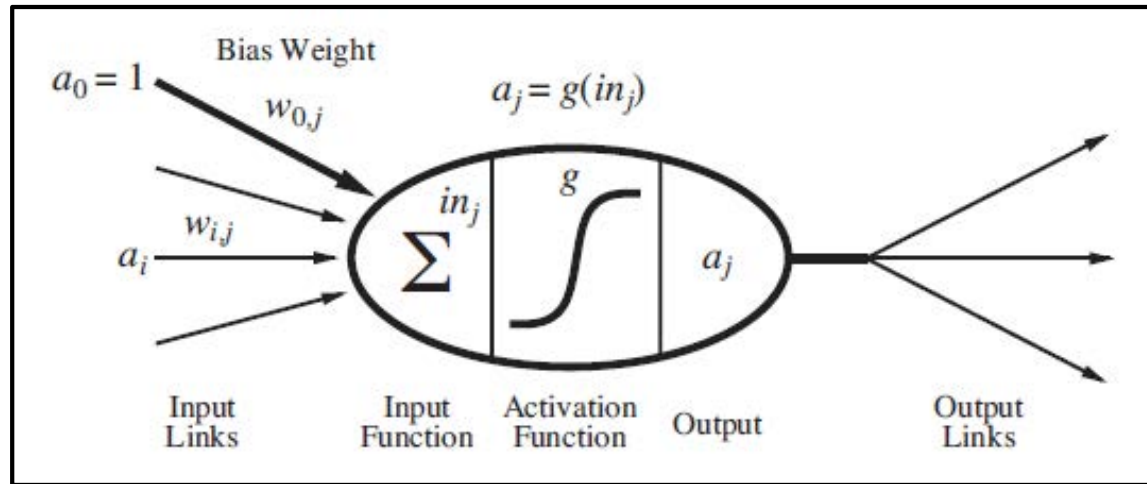
Human brain

- 100 billion neurons
- 100 to 500 trillion synapses



ARTIFICIAL NEURAL NETWORKS (ANN)

Neural Network Structure

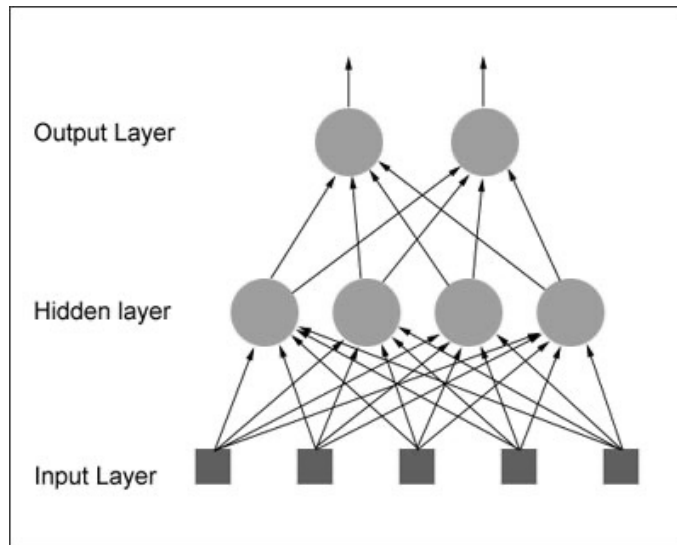


$$in_j = \sum_{i=0}^n w_{i,j} a_i$$

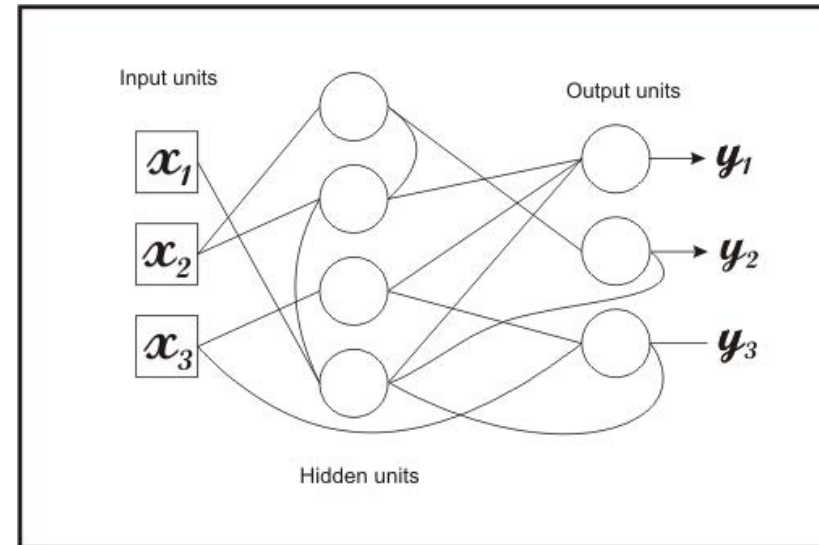
$$a_j = g(in_j) = g \left(\sum_{i=0}^n w_{i,j} a_i \right)$$

- Perceptron: hard thresholding
- Sigmoid perceptron: soft thresholding, e.g., logistic function

Feed-forward network

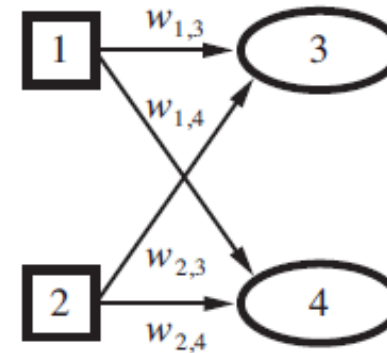


Recurrent network

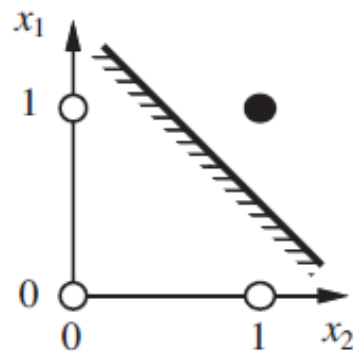


Single-Layer Feed-Forward Neural Networks

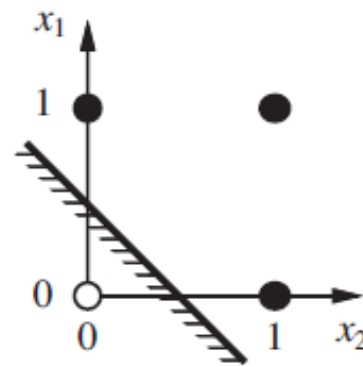
x_1	x_2	y_3 (carry)	y_4 (sum)
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



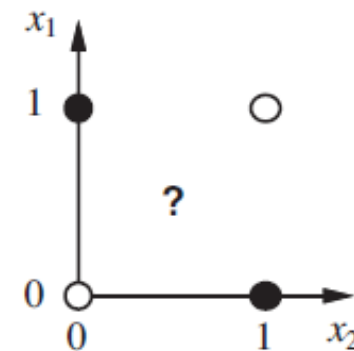
- Perceptron learning rule
- Logistic regression



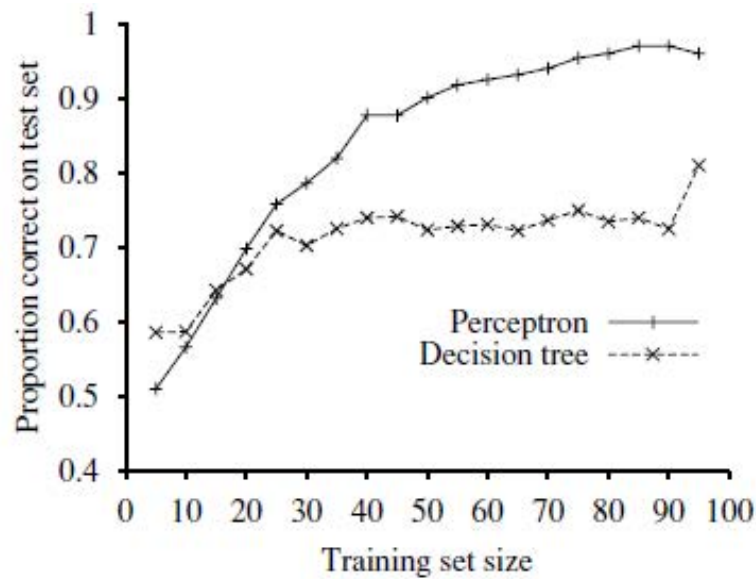
(a) x_1 and x_2



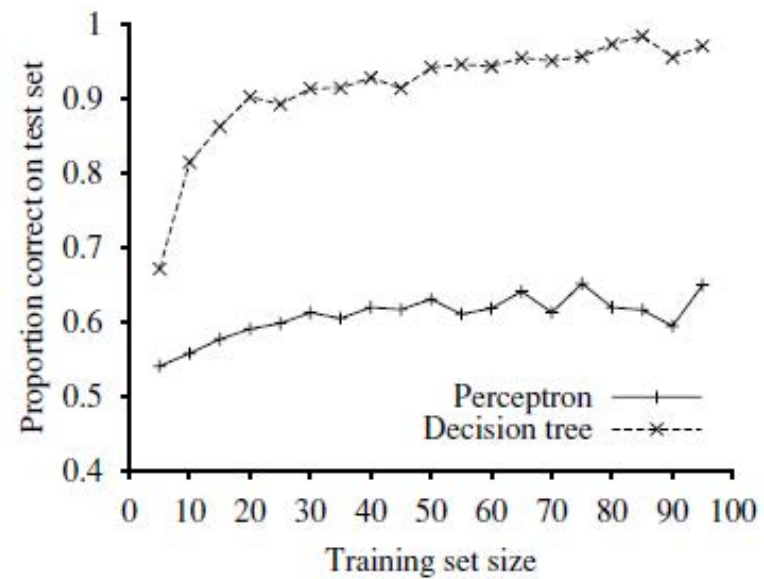
(b) x_1 or x_2



(c) x_1 xor x_2

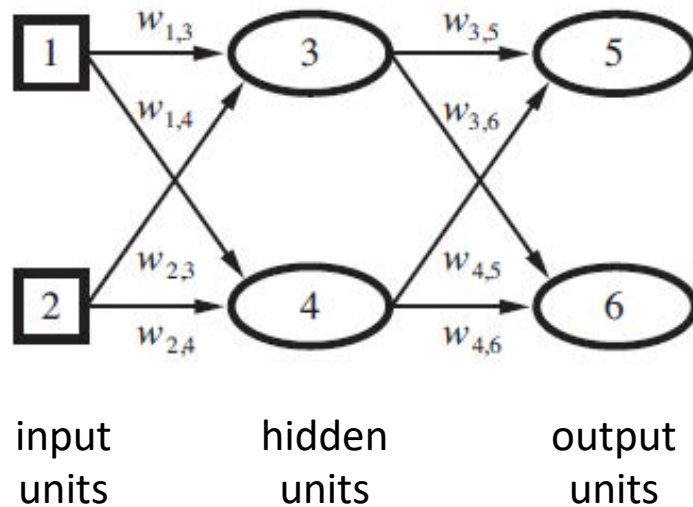


Majority function
(11 Boolean inputs)



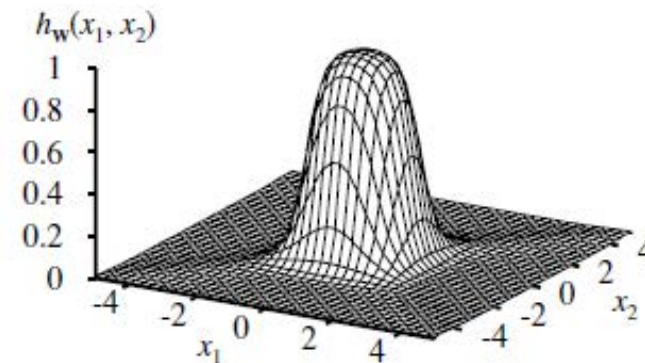
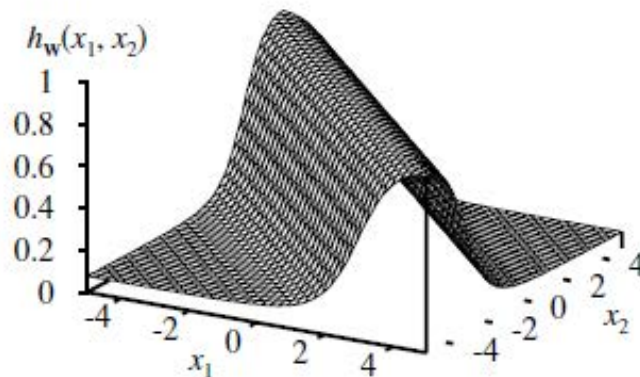
WillWait
(Restaurant example)

Multilayer Feed-Forward Neural Networks



Input units: $(a_1, a_2) = (x_1, x_2)$

$$\begin{aligned}
 a_5 &= g(w_{0,5} + w_{3,5} a_3 + w_{4,5} a_4) \\
 &= g(w_{0,5} + w_{3,5} g(w_{0,3} + w_{1,3} a_1 + w_{2,3} a_2) \\
 &\quad + w_{4,5} g(w_{0,4} + w_{1,4} a_1 + w_{2,4} a_2)) \\
 &= g(w_{0,5} + w_{3,5} g(w_{0,3} + w_{1,3} x_1 + w_{2,3} x_2) \\
 &\quad + w_{4,5} g(w_{0,4} + w_{1,4} x_1 + w_{2,4} x_2))
 \end{aligned}$$



An ANN with a single (sufficiently large) hidden layer can represent any continuous function.

Back-Propagation

- Let $\mathbf{h}_{\mathbf{w}}$ be a nonlinear function representing a multilayer feed-forward neural network, where $\mathbf{w}$ is a weight vector.
- We can learn a neural network by finding weights using a gradient descent algorithm.
- Gradient descent update rule:

$$w_i \leftarrow w_i - \alpha \frac{\partial}{\partial w_i} \text{Loss}(\mathbf{w}).$$

$$\frac{\partial}{\partial w} \text{Loss}(\mathbf{w}) = \frac{\partial}{\partial w} |\mathbf{y} - \mathbf{h}_{\mathbf{w}}(\mathbf{x})|^2 = \frac{\partial}{\partial w} \sum_k (y_k - a_k)^2 = \sum_k \frac{\partial}{\partial w} (y_k - a_k)^2$$

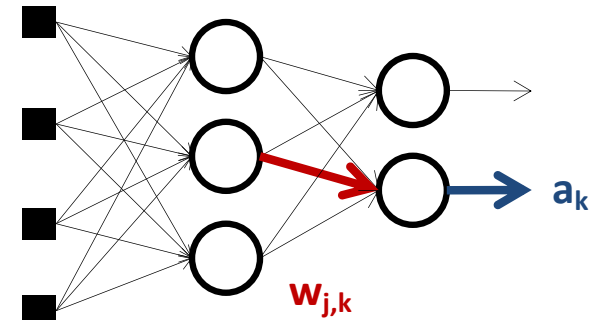
from the j^{th} hidden unit to the k^{th} output

$$Loss_k = (y_k - a_k)^2$$

$$\frac{\partial Loss_k}{\partial w_{j,k}} = -2(y_k - a_k) \frac{\partial a_k}{\partial w_{j,k}} = -2(y_k - a_k) \frac{\partial g(in_k)}{\partial w_{j,k}}$$

$$= -2(y_k - a_k) g'(in_k) \frac{\partial in_k}{\partial w_{j,k}} = -2(y_k - a_k) g'(in_k) \frac{\partial}{\partial w_{j,k}} \left(\sum_j w_{j,k} a_j \right)$$

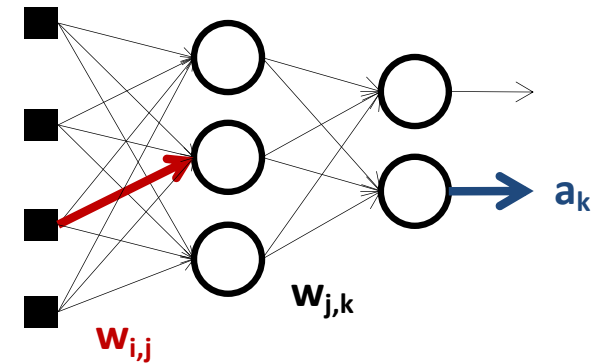
$$= -2 \underbrace{(y_k - a_k) g'(in_k)}_{\Delta_k} a_j = -a_j \Delta_k ,$$



from the i^{th} input to the j^{th} hidden unit

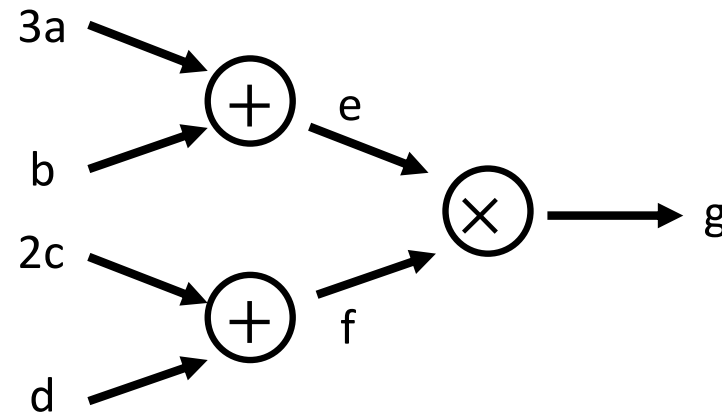
$$Loss_k = (y_k - a_k)^2$$

$$\begin{aligned}\frac{\partial Loss_k}{\partial w_{i,j}} &= -2(y_k - a_k) \frac{\partial a_k}{\partial w_{i,j}} = -2(y_k - a_k) \frac{\partial g(in_k)}{\partial w_{i,j}} \\ &= -2(y_k - a_k) g'(in_k) \frac{\partial in_k}{\partial w_{i,j}} = -2\Delta_k \frac{\partial}{\partial w_{i,j}} \left(\sum_j w_{j,k} a_j \right) \\ &= -2\Delta_k w_{j,k} \frac{\partial a_j}{\partial w_{i,j}} = -2\Delta_k w_{j,k} \frac{\partial g(in_j)}{\partial w_{i,j}} \\ &= -2\Delta_k w_{j,k} g'(in_j) \frac{\partial in_j}{\partial w_{i,j}} \\ &= -2\Delta_k w_{j,k} g'(in_j) \frac{\partial}{\partial w_{i,j}} \left(\sum_i w_{i,j} a_i \right) \\ &= -2\Delta_k w_{j,k} g'(in_j) a_i = \underbrace{-a_i \Delta_j}_{\Delta_j},\end{aligned}$$



Example

Computation
graph



$$\begin{aligned}g &= ef \\e &= 3a + b \\f &= 2c + d\end{aligned}$$

$$g = (3a + b)(2c + d)$$

$$Loss = (y - g)^2$$

$$\frac{\partial Loss}{\partial a} = \frac{\partial}{\partial a} (y - g)^2$$

$$\begin{aligned}\frac{\partial Loss}{\partial a} &= -2(y - g) \frac{\partial g}{\partial a} = -2(y - g) \frac{\partial g}{\partial e} \frac{\partial e}{\partial a} = -2(y - g) \cdot f \cdot 3 \\&= -6(y - g)(2c + d)\end{aligned}$$

Issues

- Overfitting
 - Complex model
 - Not enough data
- Vanishing/exploding gradient problem
 - Cannot train many layers of a network
- Other competing methods
 - Support vector machines
 - Bayesian networks
- Breakthroughs
 - Faster computers, GPUs
 - Cheap memory (enabling large data)
 - New techniques