人工智能
——神经网络模型

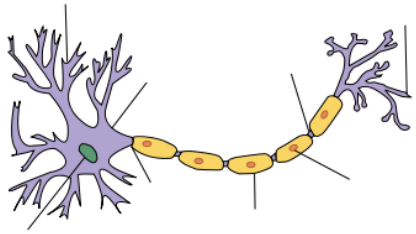
Yanghui Rao
Assistant Prof., Ph.D
School of Data and Computer Science,
Sun Yat-sen University
raoyangh@mail.sysu.edu.cn

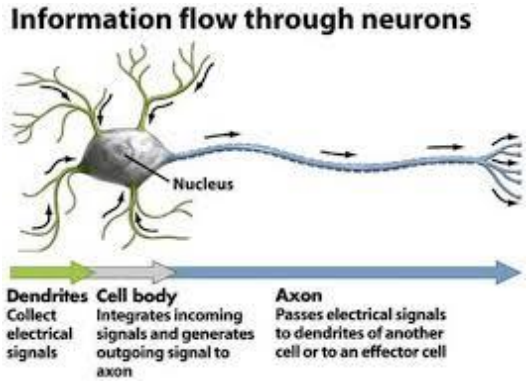
神经网络模型

- 人工神经网络的研究是受到模拟生物神经系统的尝试的启发
- 人脑主要由称为神经元(neurons)的神经细胞组成
- 神经元通过称为轴突(axons)的纤维链连接在一起
- 每当神经元被刺激时，轴突被用来将神经冲动从一个神经元传递到另一个神经元

神经网络模型

- 树突是神经元细胞体的延伸，神经元通过树突延伸到其他神经元的轴突上
- 树突和轴突之间的接触点称为突触(synapse)
- 人类的大脑通过改变神经元之间的突触联系的强度来反复刺激相同的冲动





神经网络模型

- 神经网络模型由大量简单的交互节点组成(人工神经元).
- 知识由这些节点之间的连接强度来表示.
- 知识是通过学习过程调整连接而获得的.
- 所有的神经元同时并独立地处理他们的输入.

人工神经元

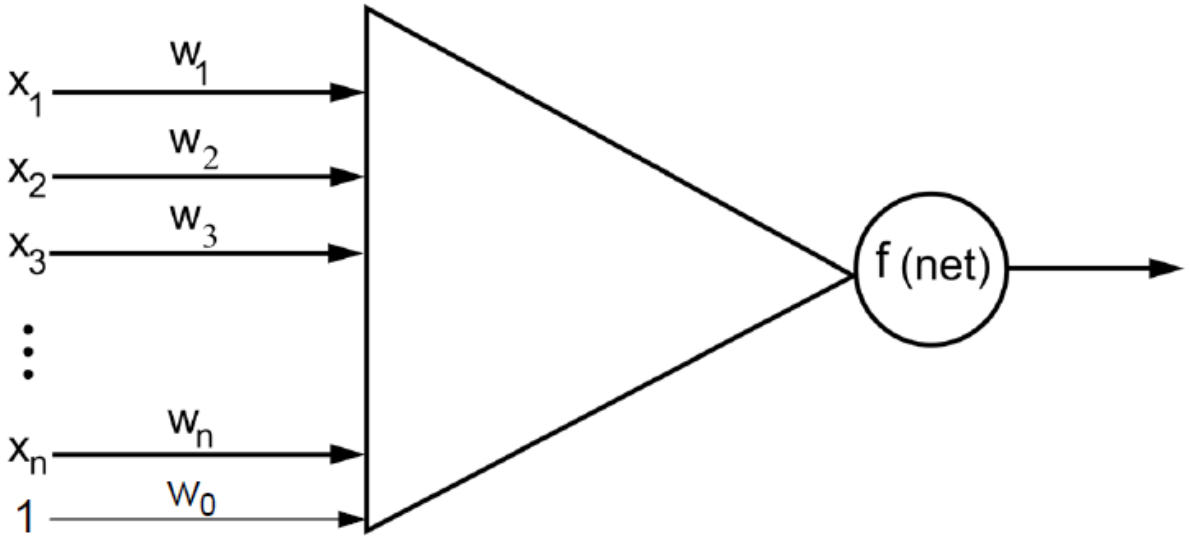
- 神经网络中的计算单位是人工神经元.

- 人工神经元的组成部分为:
  - 输入信号 x_i. 这些信号代表来自环境的数据或其他神经元的激活(activation).
  - 一组实值(real-value)权重 w_i. 这些权重的值代表连接强度.
  - 一个激活层 $\sum_i w_i x_i$. 神经元的激活层由加权输入的总和确定.
  - 一个阈值函数 f. 该函数通过确定激活是低于还是高于阈值来计算最终输出.

人工神经元

- 给定一个激活值 $net = \sum_i w_i x_i$，该神经元的输出为：

$$f(net) = \begin{cases} +1 & if \sum_i w_i x_i \geq 0 \\ -1 & if \sum_i w_i x_i < 0 \end{cases}$$



示例

- 可以使用人工神经元来计算逻辑“与”功能.
  - 神经元有三个输入
  - x_1 和 x_2 表示原始输入.
  - 第三个是具有固定值+1的偏置输入.
  - 输入数据和偏差分别具有+1，+1和-2的权重.
- 那么逻辑或功能呢?

人工神经元

- 感知机学习算法（PLA）可用于调整人工神经元的权重.
- 调整权重，直到神经元的输出与训练样例的真实输出一致.
- 使用以下规则

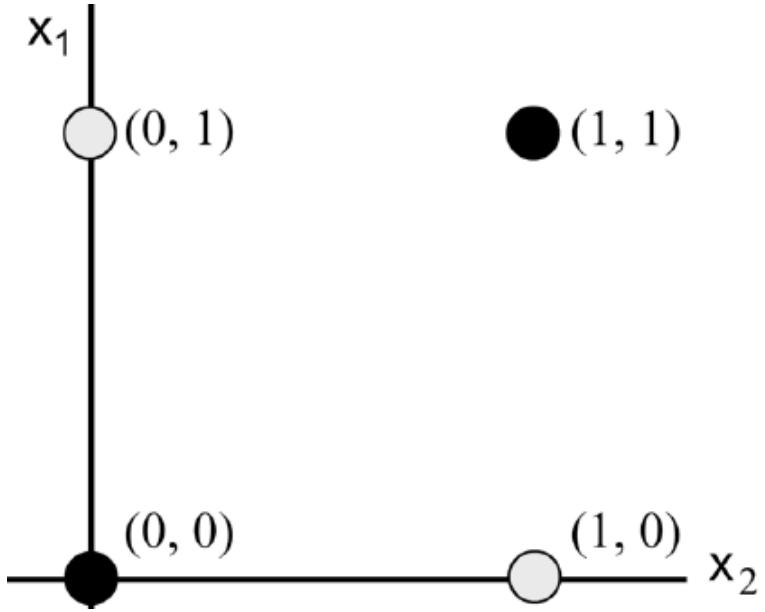
$$\mathbf{w}_{(t+1)} \leftarrow \mathbf{w}_{(t)} + y_{n(t)}\mathbf{x}_{n(t)}$$

人工神经元

- 感知机学习算法不能解决模式不可线性分离的问题.
- 一个简单的例子是异或问题.
- 需要多层网络来解决这类问题.

示例

x_1	x_2	Output
1	1	-1
1	0	1
0	1	1
0	0	-1



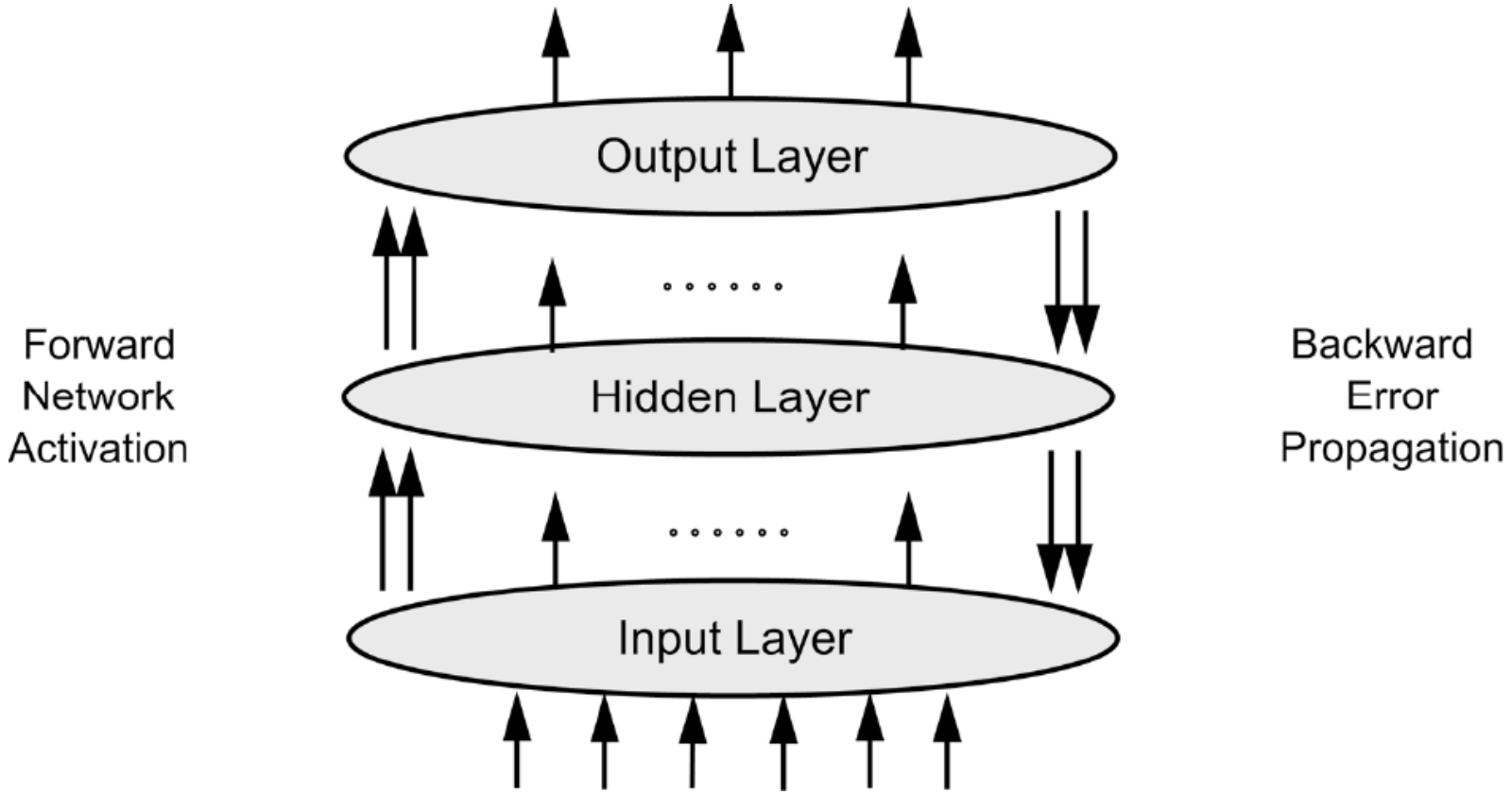
神经网络模型

- 输入和输出节点之间的附加层称为隐藏层.
- 嵌入在这些层中的节点被称为隐藏节点.
- 一个前馈神经网络(feedforward neural networks)，其中一层中的节点仅连接到下一层中的节点.
- 反向传播学习算法是专门为多层神经网络设计的。

神经网络模型

- 训练算法的每次迭代中有两个阶段
  - 前向传递阶段(forward phase)
  - 反向传递阶段(backward phase)

神经网络模型


Output Layer
Hidden Layer
Input Layer
Forward
Network
Activation
Backward
Error
Propagation

神经网络模型

- 在前向传递阶段，从前一次迭代获得的权重被用来计算每个神经元的输出值.

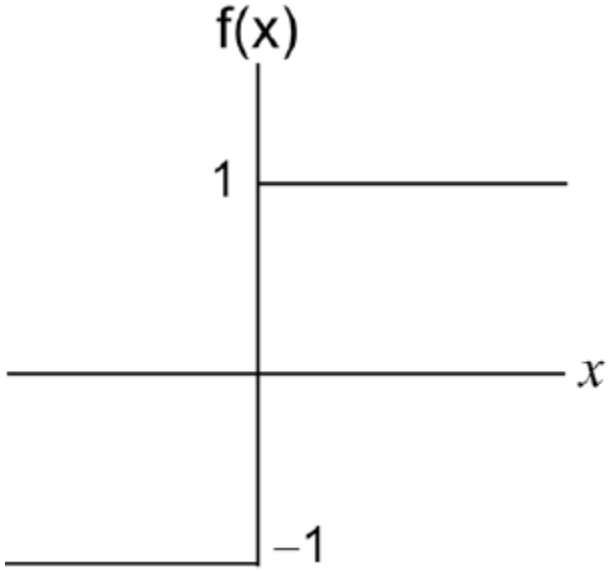
- 在计算第$(l+1)$层的输出之前计算第(l)层处的神经元的输出.

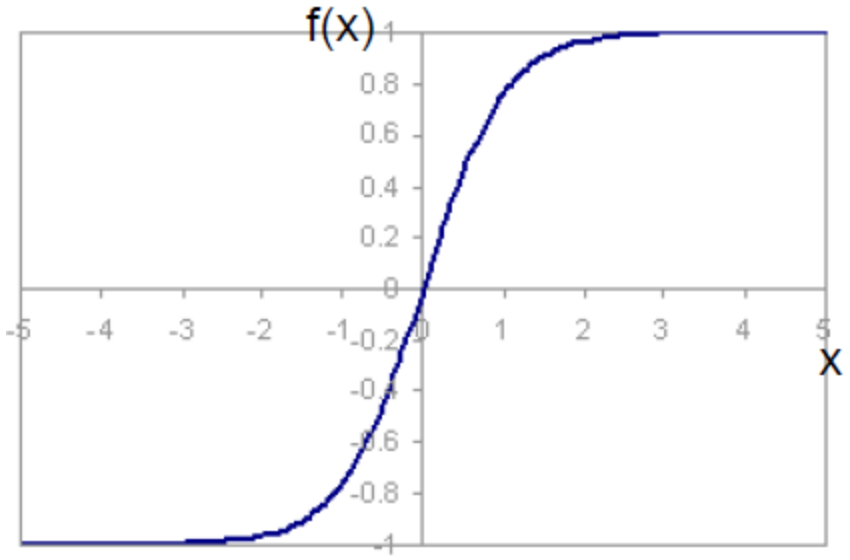
神经网络模型

- 在反向传递阶段，权重更新方程应用于相反的方向.
- 换句话说，第$(l+1)$层的权重在更新第(l)层的权重之前被更新.
- 这种学习算法允许我们使用第$(l+1)$层神经元的误差来估计第(l)层神经元的误差.

神经网络模型

- 对于这种类型的网络，不采用PLA的阈值函数，而是使用另一个激活函数(activation function).





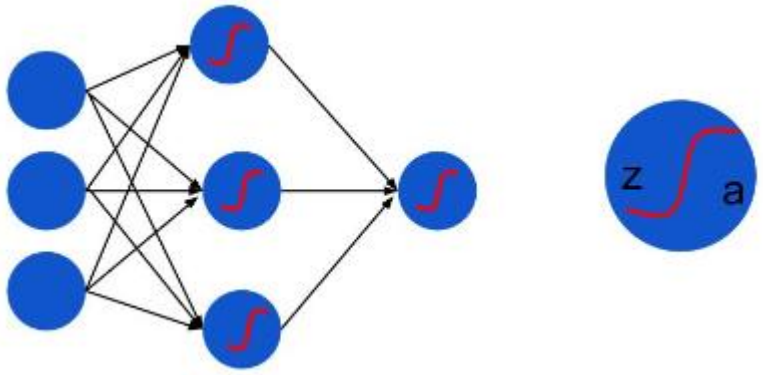
神经网络模型

- 常见的激活函数是双曲正切函数（tanh）

$$f(x)=\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}=\frac{e^{2x}-1}{e^{2x}+1}$$

- 这个函数的一个重要属性是可求导的

$$f'(x)=1-f(x)^{2}$$



神经网络模型

- 另一个常见的激活函数是sigmoid函数

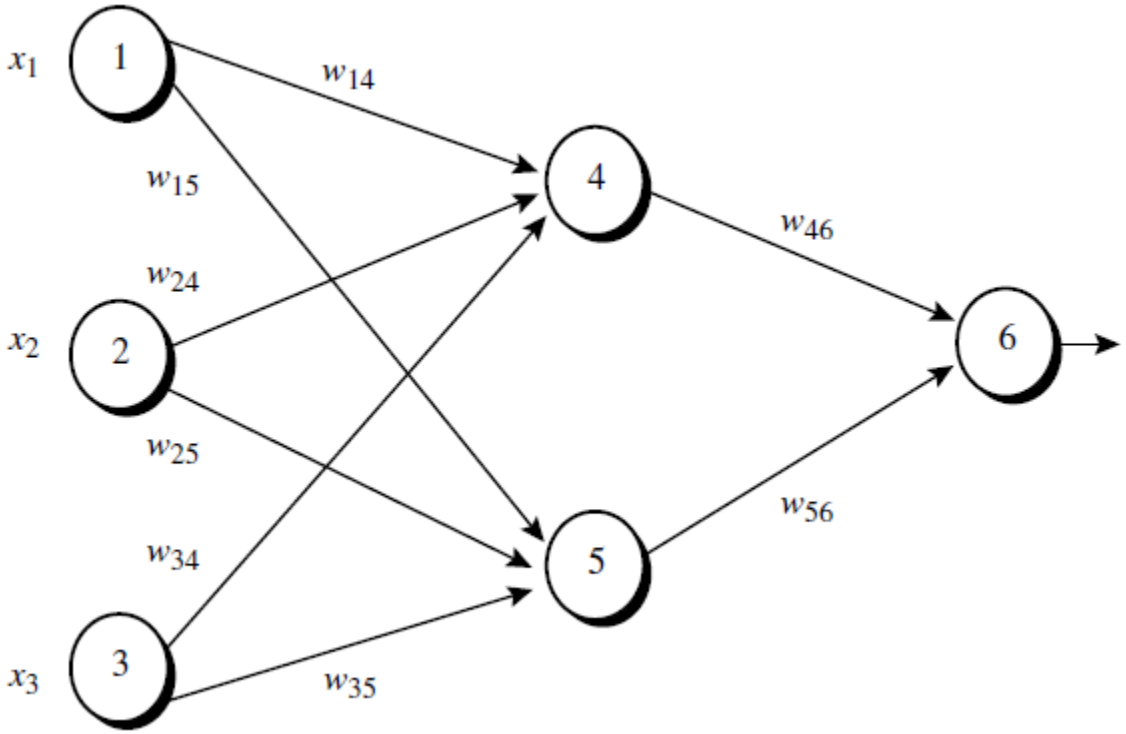
$$f(x)=\frac{1}{1+e^{-x}}$$

- 这个函数也具备可求导的属性

$$f'(x)=f(x)(1-f(x))$$

示例

- 一个训练元组**X**=(1,0,1), 标签为1.



示例

Initial input, weight, and bias values.

x_1	x_2	x_3	w_{14}	w_{15}	w_{24}	w_{25}	w_{34}	w_{35}	w_{46}	w_{56}	θ_4	θ_5	θ_6
1	0	1	0.2	−0.3	0.4	0.1	−0.5	0.2	−0.3	−0.2	−0.4	0.2	0.1

The net input and output calculations.

Unit j	*Net input,* I_j	*Output,* O_j
4	$0.2+0-0.5-0.4=-0.7$	$1/(1+e^{0.7})=0.332$
5	$-0.3+0+0.2+0.2=0.1$	$1/(1+e^{-0.1})=0.525$
6	$(-0.3)(0.332)-(0.2)(0.525)+0.1=-0.105$	$1/(1+e^{0.105})=0.474$

示例

Calculation of the error at each node.

Unit j	*Err$_j$*
6	$(0.474)(1-0.474)(1-0.474)=0.1311$
5	$(0.525)(1-0.525)(0.1311)(-0.2)=-0.0065$
4	$(0.332)(1-0.332)(0.1311)(-0.3)=-0.0087$

Calculations for weight and bias updating.

Weight or bias	*New value*
w_{46}	$-0.3+(0.9)(0.1311)(0.332)=-0.261$
w_{56}	$-0.2+(0.9)(0.1311)(0.525)=-0.138$
w_{14}	$0.2+(0.9)(-0.0087)(1)=0.192$
w_{15}	$-0.3+(0.9)(-0.0065)(1)=-0.306$
w_{24}	$0.4+(0.9)(-0.0087)(0)=0.4$
w_{25}	$0.1+(0.9)(-0.0065)(0)=0.1$
w_{34}	$-0.5+(0.9)(-0.0087)(1)=-0.508$
w_{35}	$0.2+(0.9)(-0.0065)(1)=0.194$
θ_6	$0.1+(0.9)(0.1311)=0.218$
θ_5	$0.2+(0.9)(-0.0065)=0.194$
θ_4	$-0.4+(0.9)(-0.0087)=-0.408$

神经网络模型

前向传播输入

- 给定隐藏层或输出层的单元 j, 单位j的净输入I_j为

$$I_j = \sum_i w_{ij} O_i + \theta_j$$

w_{ij}是从上一层单元i到单元j的连接权重; O_i是上一层单元i的输出; θ_j 单元的偏置.

- 给定单元j的输入I_j，则单位j的输出O_j的公式为:

$$O_j = \frac{1}{1+e^{-I_j}}$$

反向传播误差

- 对于输出层中的单元k，误差Err_k由下式计算:

$$Err_k = O_k(1-O_k)(T_k - O_k)$$

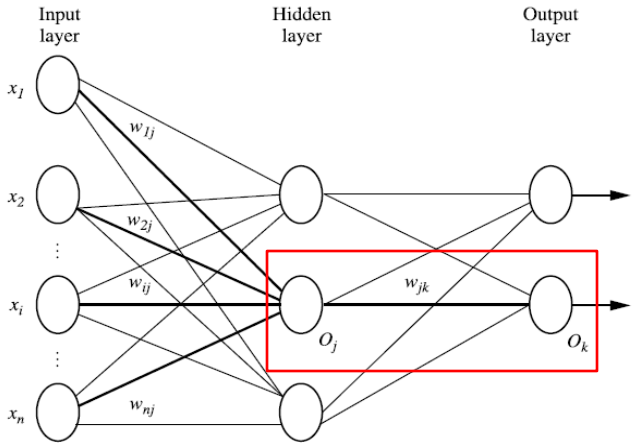
- 隐藏层单元 j 的误差为:

$$Err_j = O_j(1-O_j)\sum_k Err_k w_{jk}$$

- 权重的更新公式为

$$w_{jk} = w_{jk} + \eta Err_k O_j$$

$$\theta_k = \theta_k + \eta Err_k$$

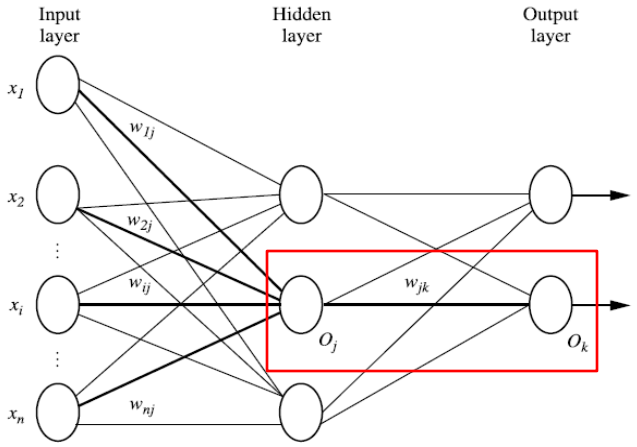


神经网络模型

- 最小化节点O_k的误差
- 我们将其定义为 $E = \frac{1}{2}e^2 = \frac{1}{2}(T-O)^2$
- 为了调整权重 w_{jk}，我们首先计算在w_{jk}上E的偏导数

$$\frac{\partial E}{\partial w_{jk}} = \frac{\partial E}{\partial e} \times \frac{\partial e}{\partial O_k} \times \frac{\partial O_k}{\partial w_{jk}}$$
$$= -(e) \times (O_k(1-O_k)) \times (O_j)$$
$$= -(T_k - O_k)O_k(1-O_k)O_j$$

- 然后我们使用梯度下降法



神经网络模型

- 反向传播学习算法是基于错误曲面的思想.
- 表面代表了作为网络权重函数的数据集上的累积误差.
- 曲面表示数据集上的累积误差每个可能的网络权重配置由表面上的点表示为网络权重的函数.

神经网络模型

- 学习算法的目标是确定一组最小化错误的权重.
- 学习算法应该被设计成能够找到表面上最快速减少错误的方向.
- 这可以通过在每个表面点处的梯度矢量的相反方向上移动来实现（即，通过采用梯度下降学习方法）。

神经网络模型

- **缺点**
  - 训练时间长
  - 通常需要经验最佳确定的多个参数，例如网络拓扑或“结构”.
  - 可解释性差
    - 很难解释网络中学习权重和“隐藏单位”背后的象征意义

神经网络模型

- 优点
  - 高鲁棒性
  - 非常适合于连续值输入和输出
  - 在广泛的现实数据上取得成功
  - 最近已经开发了用于从训练的神经网络提取规则的技术

神经网络模型

- **从网络中提取规则**：网络修剪(network pruning)
  - 通过去除对训练网络影响最小的加权链接来简化网络结构
  - 研究输入和激活值的集合以导出描述输入和隐藏单元层之间关系的规则

- **敏感性(Sensitivity)分析**：评估给定输入变量对网络输出的影响。从这个分析中获得的知识可以用规则来表示