

# 人工神经网络

机器学习研究室

计算机科学与技术学院

吉林大学

# 大纲

- 人工神经网络概述
- 人工神经元的形式化模型
- 感知器
- 多层前馈网络
- 作业
- 补充材料
  - 人工神经网络的发展和重要模型
  - 人工神经网络重要的会议和期刊

# 人工神经网络概述



# Training a Classifier



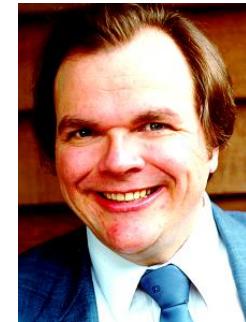
> classifier < Marks



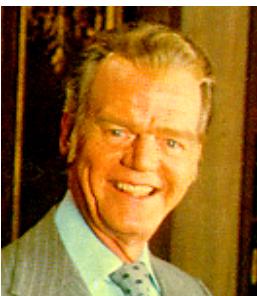
> classifier < not Marks



> classifier < not Marks



> classifier < Marks

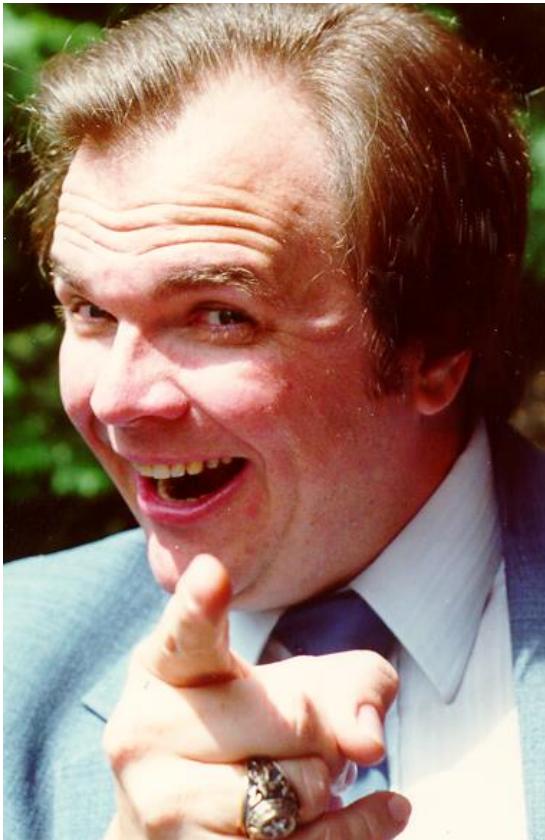


> classifier < not Marks



> classifier < not Marks

# Recall from a Trained Classifier



> Classifier > Marks

Note: The test image does not appear in the training data.

Learning ≠ Memorization

# Classifier In Feature Space, After Training

- ✕ = training data

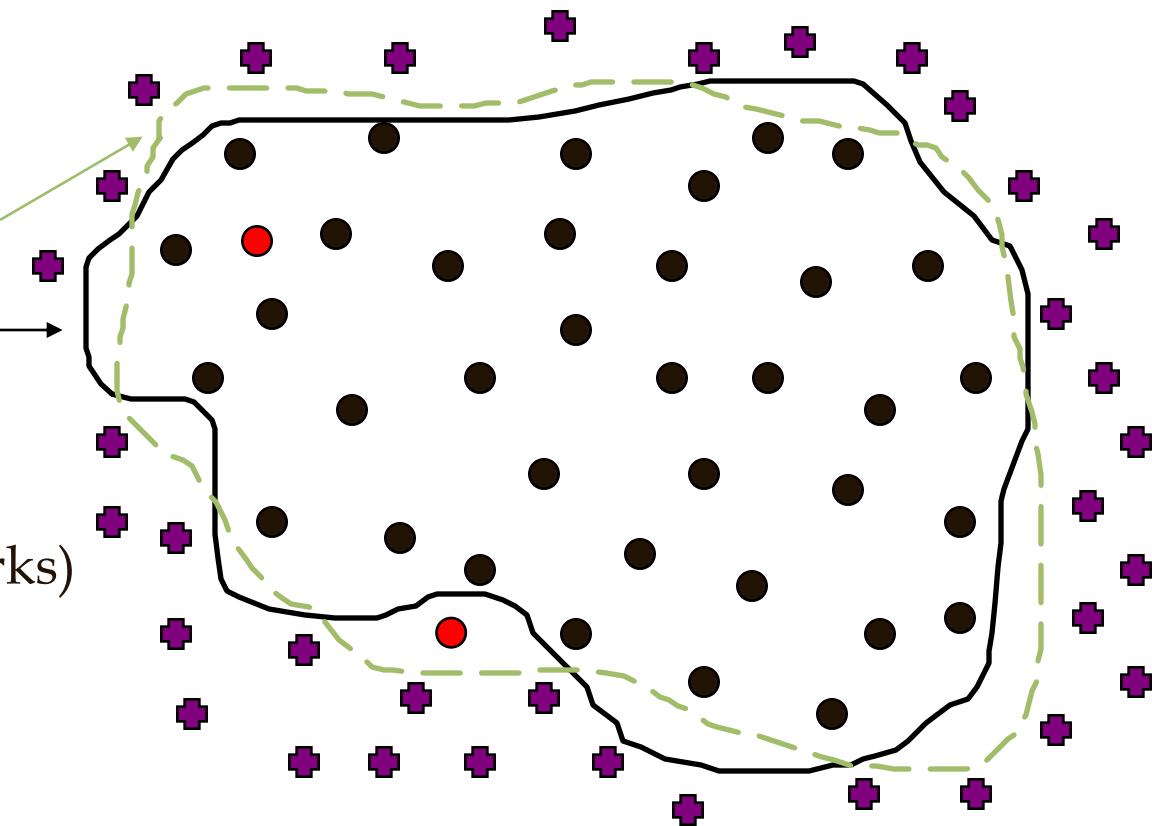
● ● = Marks

✕ = not Marks

representation

concept  
(truth)

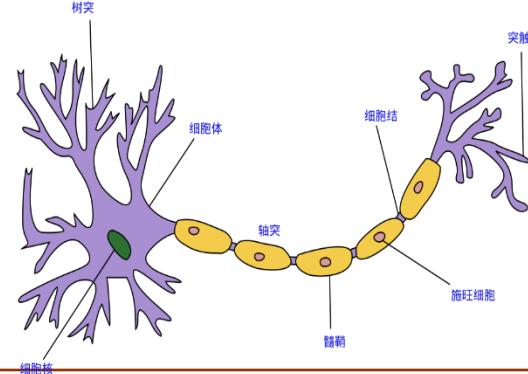
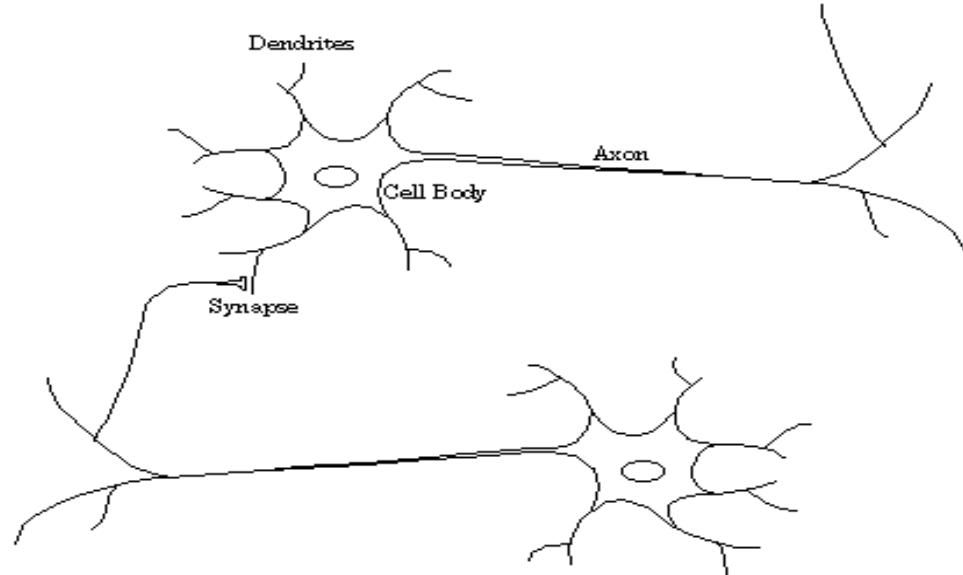
- = test data (Marks)



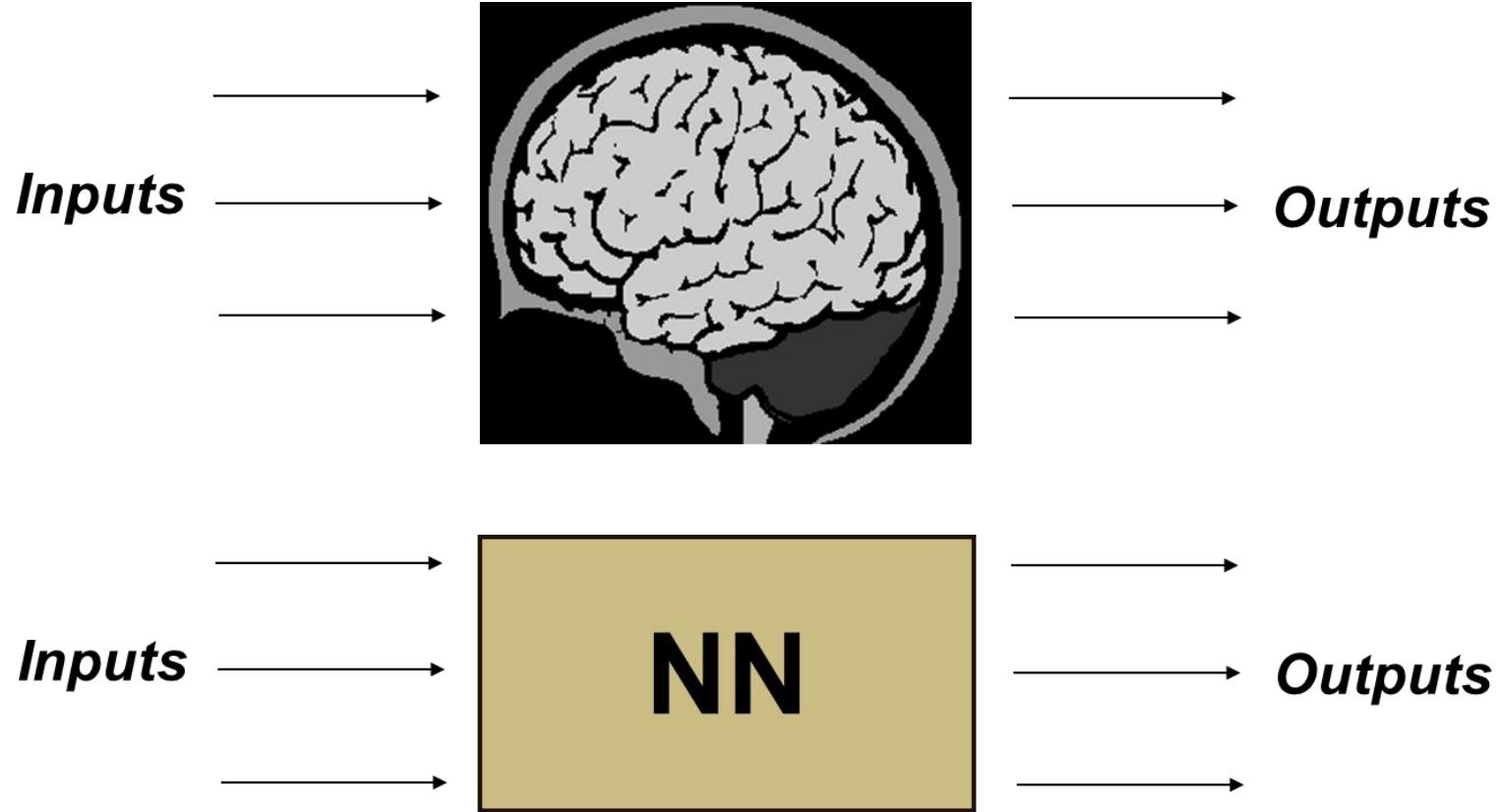
# 从大脑到神经网络

## 神经元

- **生物神经网络:**  
Biological Neural Network (BNN)
- **神经元:** neuron
  - 神经元经突触传递信号给其他神经元（胞体或树突）
  - $10^{11}$ 个神经元/人脑
  - $10^{14}$ 个连接/神经元
- **神经元基本工作机制:**
  - 状态: 兴奋与抑制
  - 互联, 激励, 处理, 阈值.



# 从大脑到神经网络



# 从大脑到神经网络

## 膜电位与神经元的兴奋

- 膜电位：细胞膜将细胞分为内外两部分，当外部电位为0时，称内部电位为膜电位。
- 静止膜电位：当没有输入信号时的膜电位称为静止膜电位,通常为-70mv左右。
- 兴奋状态：当外部有输入信号时，将使膜电位发生变化，倘若使膜电位升高，比静止膜电位高15mv以上，即超过-55mv（阈值），神经元被激活,内部电位急剧上升至100mv左右，并维持约1ms，然后急剧下降。相当输出一个100mv高1ms宽的脉冲，并沿轴突以100m/s的速度传至其它的神经元。

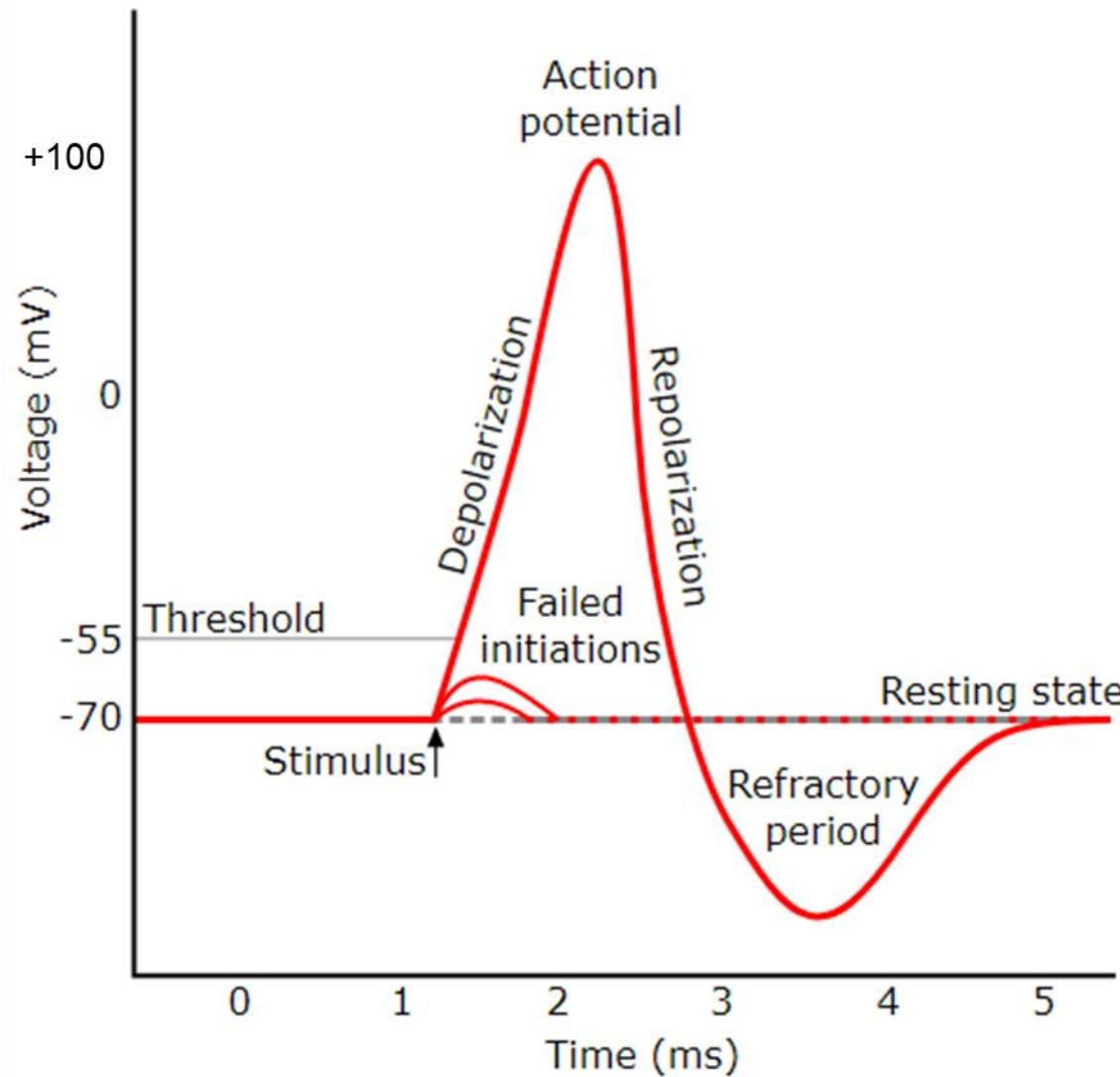
# 从大脑到神经网络

## 膜电位与神经元的兴奋

- **抑制状态**: 当外部输入信号使膜电位下降低于阈值电位时, 神经元处于抑制状态, 无脉冲输出。
- “兴奋—抑制” 状态满足 “0—1” 律
- A/D转换: 电脉冲到达各突触接口后, 放出某种化学物质, 该物质作用于各个和其相连的神经元的细胞膜, 并使其膜电位发生变化, 完成了将离散的脉冲信号转换为连续变化的电位信号。
- **不应期**: 神经元输出一个脉冲后, 一段时间内对激励不响应, 称之为不应期, 一般为几 ms。

# 生物神经元的工作机理

膜电位变化示意图



# 从大脑到神经网络

## 膜电位与神经元的兴奋

- **时间加算功能：**对于不同时间通过同一突触传入的信号具有时间的加算功能。
- **空间加算功能：**对于同一时间通过不同突触的输入信号具有空间加算功能。

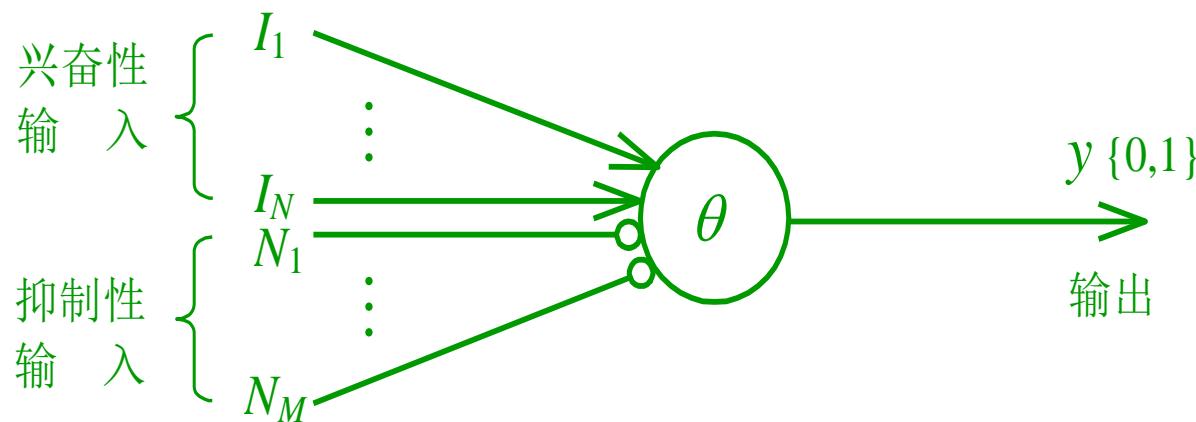
- 神经元网络中各神经元之间联结的强弱，按外部的激励信号做自适应变化，而每个神经元又随着所接收到的多个信号的综合大小而呈现兴奋或抑制状态
- 大脑的学习过程就是神经元之间连接强度随外部激励信息做自适应变化的过程，而大脑处理信息的结果则由神经元的状态表现出来

# 人工神经元的形式化模型

# 人工神经元的形式化模型

- McCulloch-Pitts Neurons(M-P 模型)
- Linear weighted model(线性加权模型)
- Logistic threshold model(阈值逻辑模型)

# M-P 模型



输入条件

输出

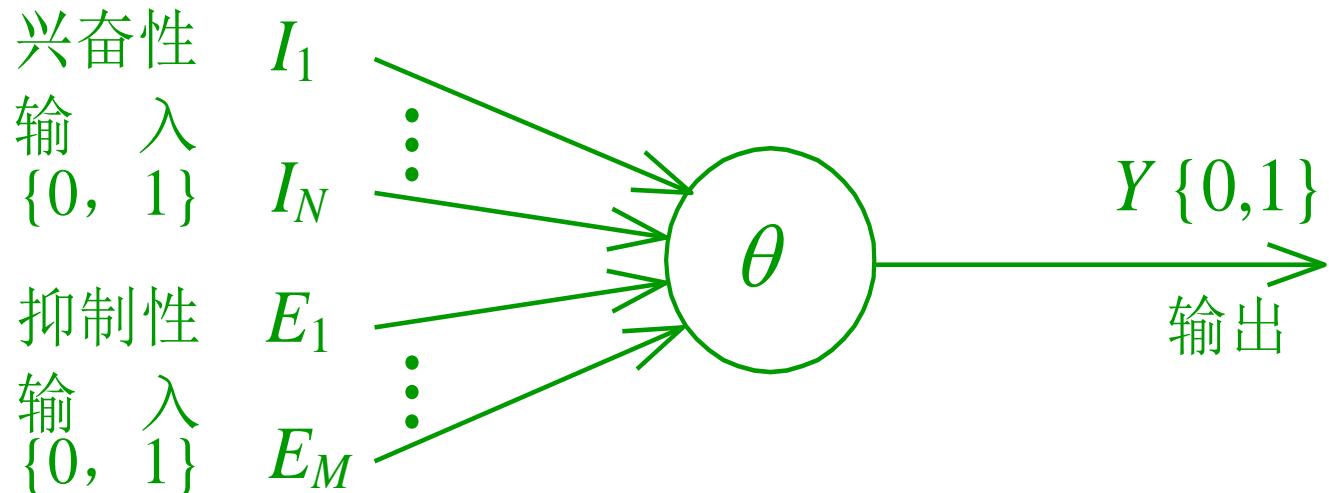
$$\sum_{i=1}^N I_i - \theta \geq 0 \quad \text{且} \quad \sum_{j=1}^M N_j = 0 \quad \text{时} \quad Y = 1$$

$$\sum_{i=1}^N I_i - \theta \geq 0 \quad \text{且} \quad \sum_{j=1}^M N_j > 0 \quad \text{时} \quad Y = 0$$

$$\sum_{i=1}^N I_i - \theta < 0 \quad \text{且} \quad \sum_{j=1}^M N_j = 0 \quad \text{时} \quad Y = 0$$

$$\sum_{i=1}^N I_i - \theta < 0 \quad \text{且} \quad \sum_{j=1}^M N_j > 0 \quad \text{时} \quad Y = 0$$

# 线性加权模型



输入条件

$$\sum_{i=1}^N I_i - \sum_{j=1}^M E_j - \theta \geq 0$$

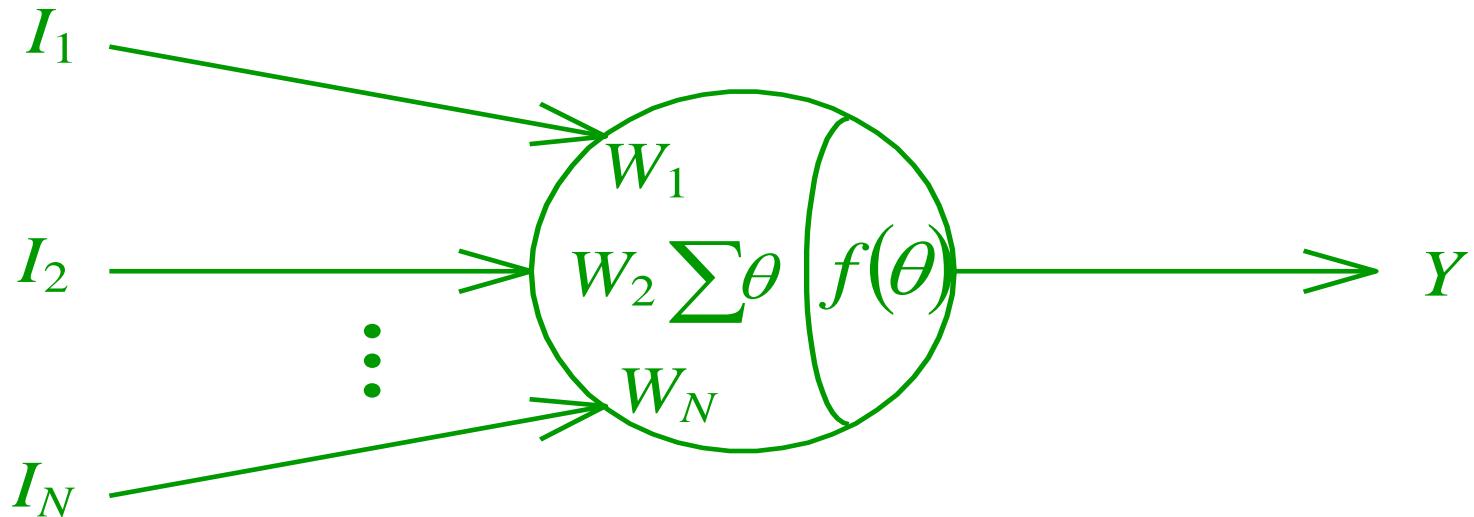
输出

$$Y = 1$$

$$\sum_{i=1}^N I_i - \sum_{j=1}^M E_j - \theta < 0$$

$$Y = 0$$

# 阈值逻辑模型



$$I_i \in \{-1, +1\}, \quad Y \in \{-1, +1\}, \quad W_i \in [-1, +1]$$

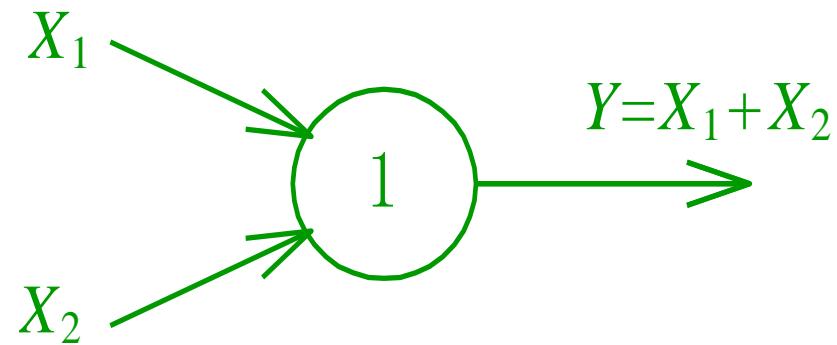
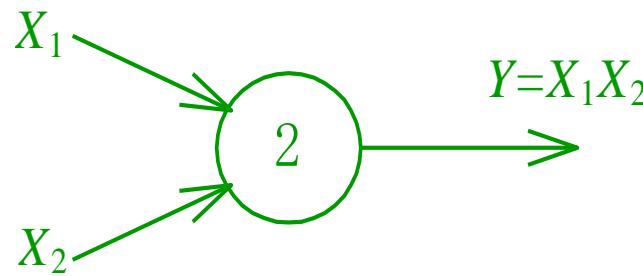
$$Y = \text{sgn}\left( \sum W_i I_i - \theta \right)$$

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

$$Y = \text{sgn}\left( \sum_{i=0}^N W_i I_i \right)$$

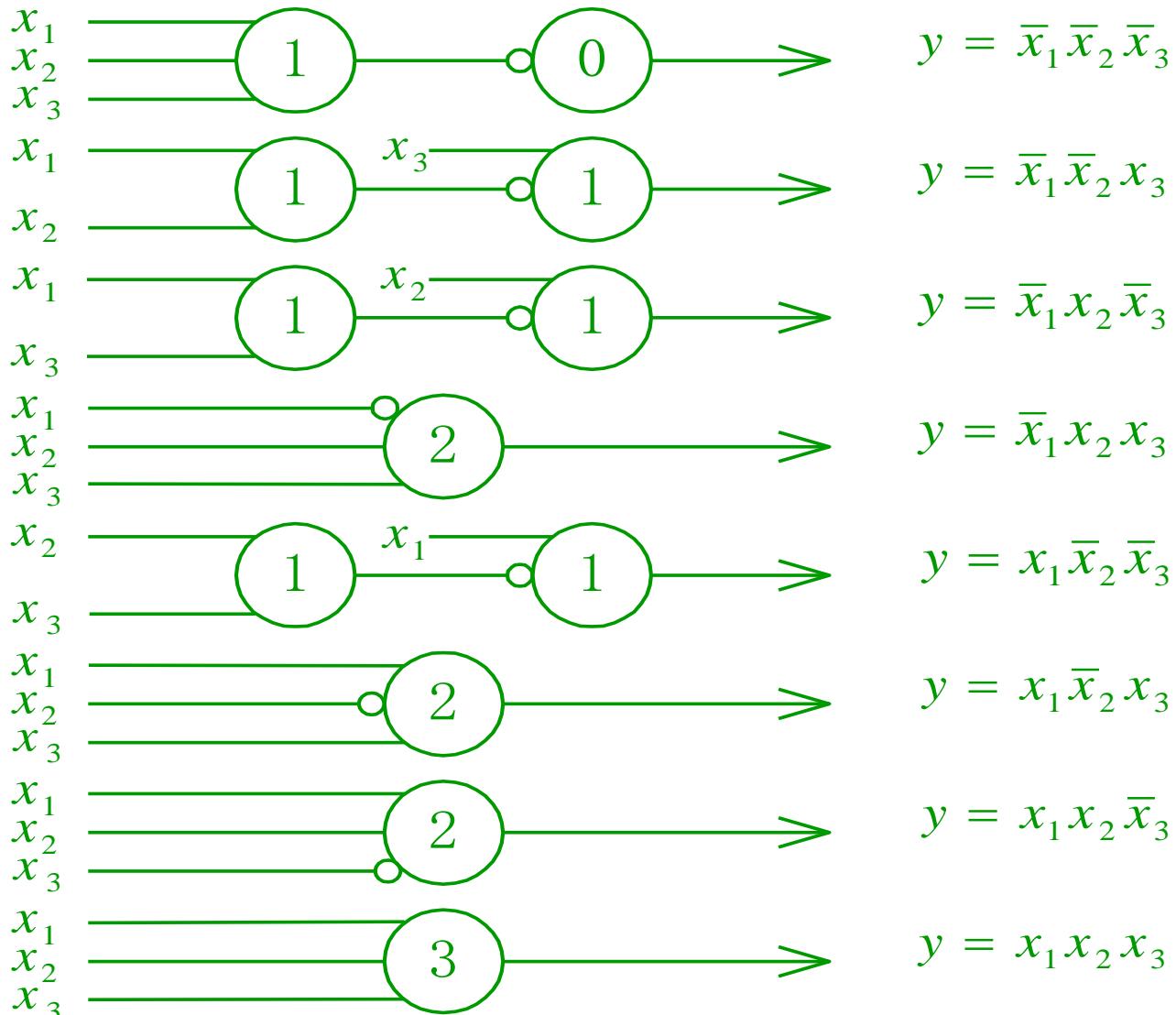
$$W_0 = -\theta, I_0 = 1$$

# 人工神经元应用举例

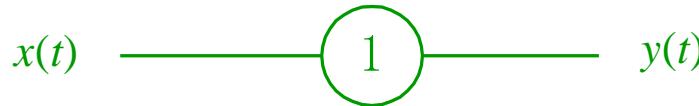


用M-P模型实现二元Boole逻辑

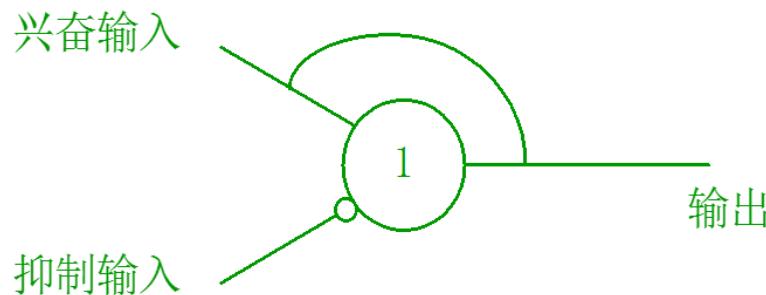
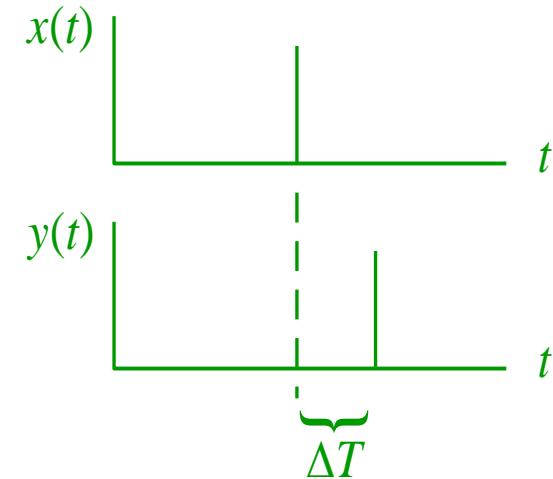
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1



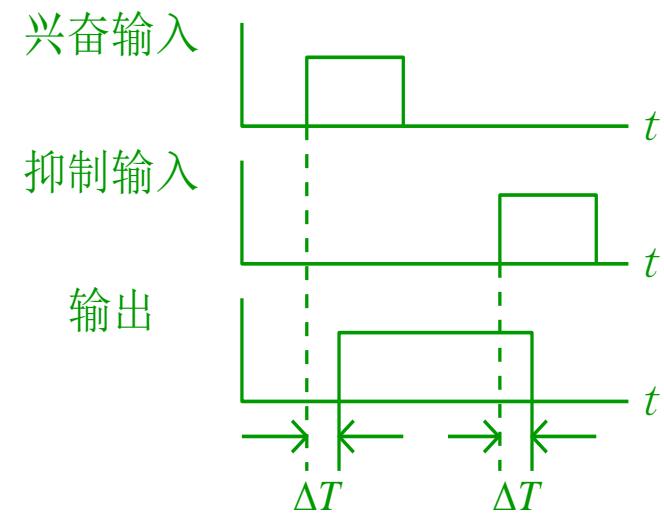
用M-P模型实现三元Boole小项逻辑



神经元的延时作用



利用神经元构造存储元件



X1代表无条件刺激  
(如巴甫洛夫试验  
中的“食物”)

$X_1$

2

N1和N2分别是限值为N1  
和N2的“达限归零”计  
数器, 即当输入N1(N2)  
个脉冲后, 计数器产生一  
个脉冲输出, 同时回零重  
新计数。

1

$Z_1$

$N_1$

$X_2$

1

$Z_2$

$N_2$

X2代表条件刺激或  
信号(如巴甫洛夫  
试验中的“铃声”)

Z1...Z5都是  
基于线性加权  
模型的神经元。

1

$Z_3$

2

$Z_4$

3

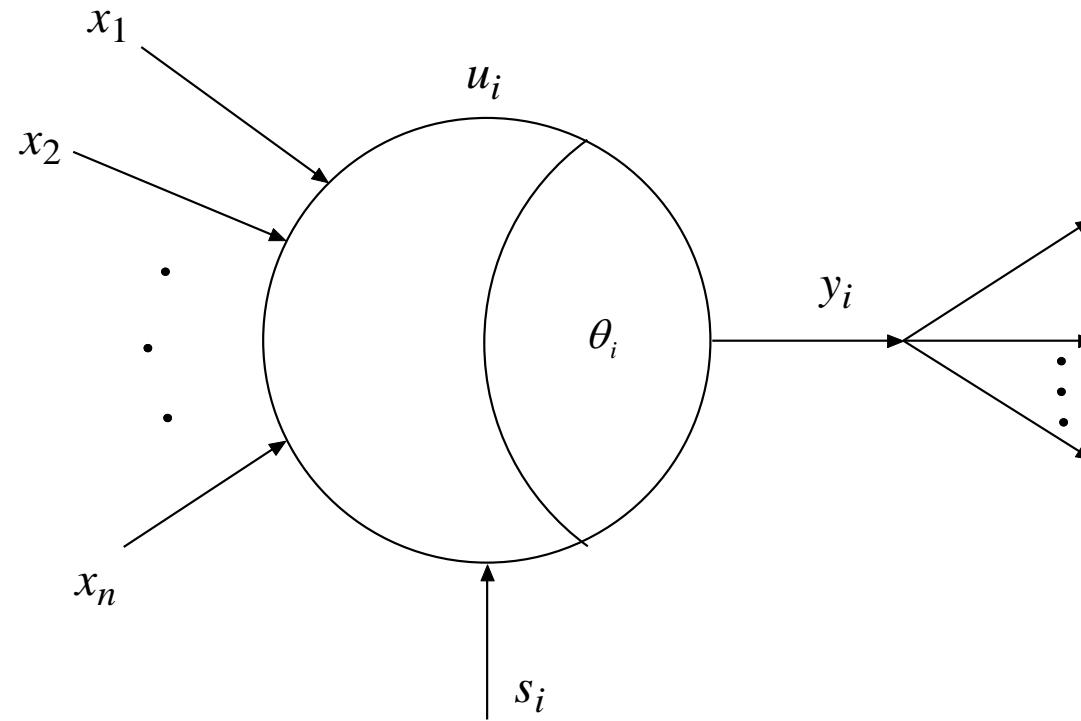
如果X1和X2同时  
输入N<N1次, 相  
当于巴甫洛夫试  
验中建立条件反  
射的训练过程。

$Z_5$

## 模拟条件反射的神经网络

- $X_1$ 代表无条件刺激(如巴甫洛夫试验中的“食物” )。
- $X_2$ 代表条件刺激或信号(如巴甫洛夫试验中的“铃声” )。
- $N_1$ 和 $N_2$ 分别是限值为 $N_1$ 和 $N_2$ 的“达限归零”计数器，即当输入 $N_1$  ( $N_2$ ) 个脉冲后，计数器产生一个脉冲输出，同时回零重新计数。
- $Z_1 \dots Z_5$ 都是基于线性加权模型的神经元，它们的阈值和联接方式都已在图中注明。
- 如果 $X_1$ 和 $X_2$ 同时输入 $N < N_1$ 次，相当于巴甫洛夫试验中建立条件反射的训练过程。

# 广义神经元模型

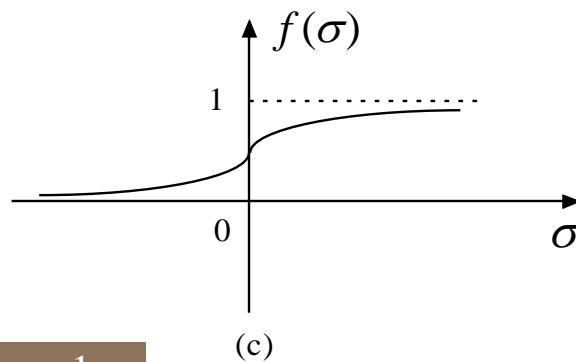
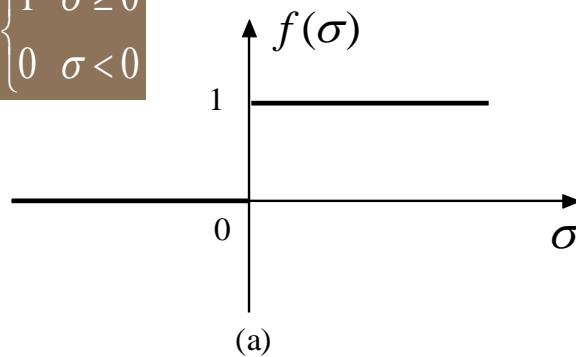


$$\sigma_i = \sum x_i w_{ij} - \theta_i$$

$$y_i = f(\sigma_i)$$

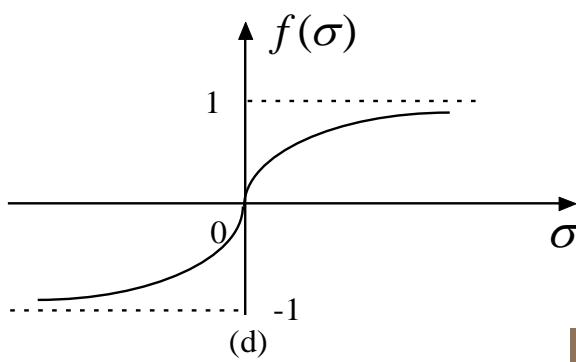
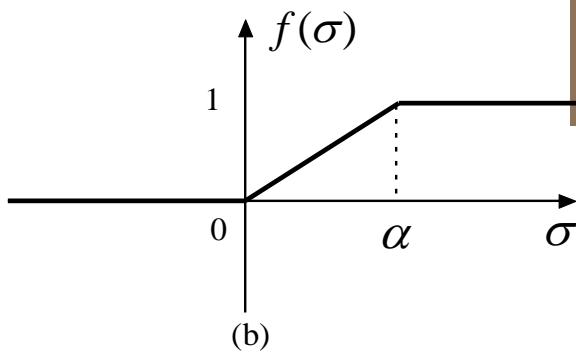
# 神经元状态转移函数的类型

$$y = f(\sigma) = \begin{cases} 1 & \sigma \geq 0 \\ 0 & \sigma < 0 \end{cases}$$



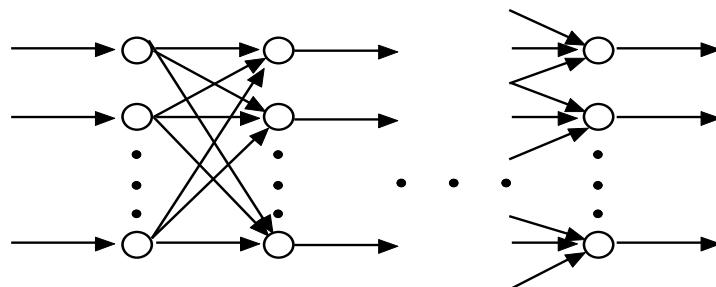
$$y = f(\sigma) = \frac{1}{1 + e^{-\sigma}}$$

$$y = f(\sigma) = \begin{cases} 1 & \sigma \geq \alpha \\ \sigma & 0 \leq \sigma < \alpha \\ 0 & \sigma < 0 \end{cases}$$

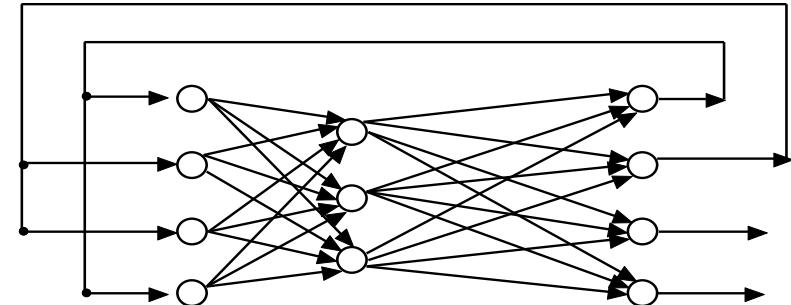


$$y = f(\sigma) = \frac{1 - e^{-\sigma}}{1 + e^{-\sigma}}$$

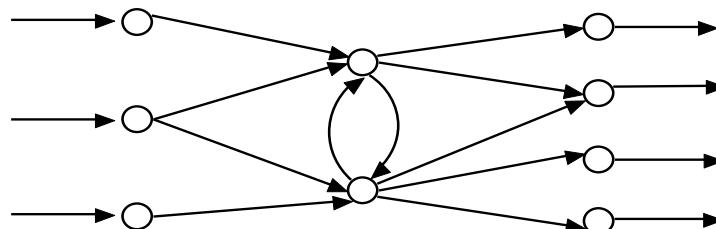
# 神经网络的拓扑结构



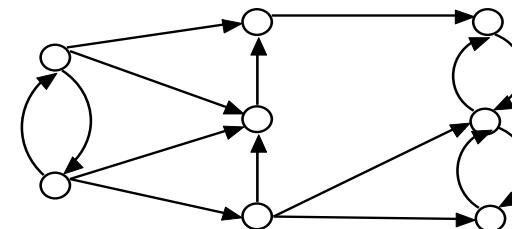
(a)



(b)



(c)



(d)

- (a) 模式分类、函数回归
- (b) 神经认知机，用来存储某种模式序列，系统辨识
- (c) 神经元可以分组进行激励响应
- (d) 网络最终进入一种动态平衡状态，可能是周期振荡或者混沌状态

# 人工神经网络的特点

■ 并行结构和并行处理

■ 知识的分布存储

- 在神经网络中，知识不是存储在特定的存储单元，而是分布在在整个系统中，要存储多个知识就需要很多连接。
- 要获得存储的知识则采用“联想”的办法，这类似于人类和动物的记忆。
- 联想记忆的两个主要特点：
  - ✓ 存储大量复杂数据的能力
  - ✓ 自适应的特征抽取能力
  - ✓ 快速的推理能力

■ 容错性

■ 自适应性

# 神经网络的特点

## 人工神经网络的局限性

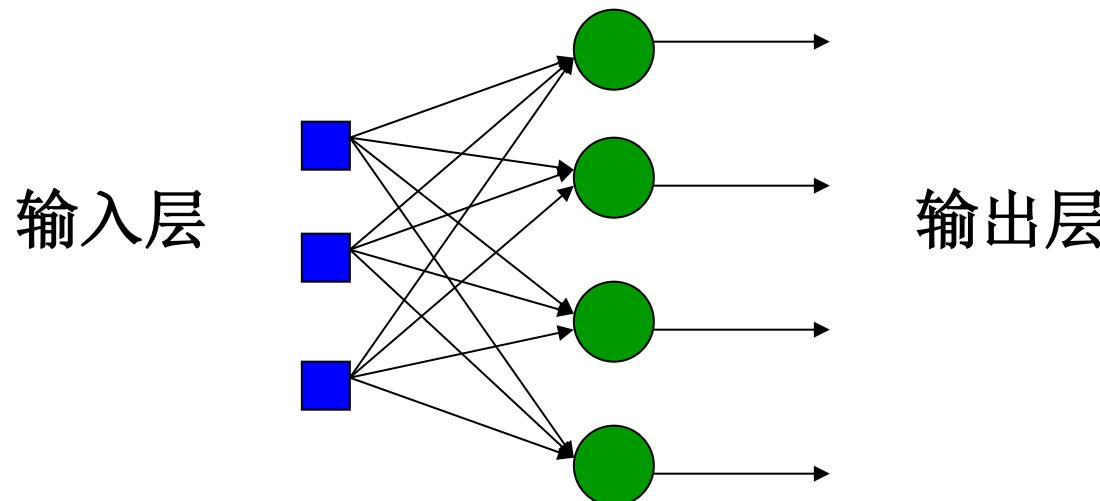
- 人工神经网络不适于高精度的计算。  
神经网络和其他计算智能方法一样，不是确定性算法。
- 人工神经网络不适于做类似顺序计数的工作。  
人工神经网络是以并行方式工作的。
- 人工神经网络的学习和训练往往是一个艰难的过程。  
网络的设计没有严格确定的方法（一般凭经验），所以选择训练方法和所需网络结构没有统一标准。  
脱机训练往往需要很长时间，为了获得最佳效果，常常要重复试验多次。  
网络收敛性的问题。

# 感知器模型和学习规则

# 感知器

## 感知器:单层前馈神经网络

感知器(Perception)是由美国学者Rosenblatt F于1957年提出的一个具有单层计算单元的神经网络。



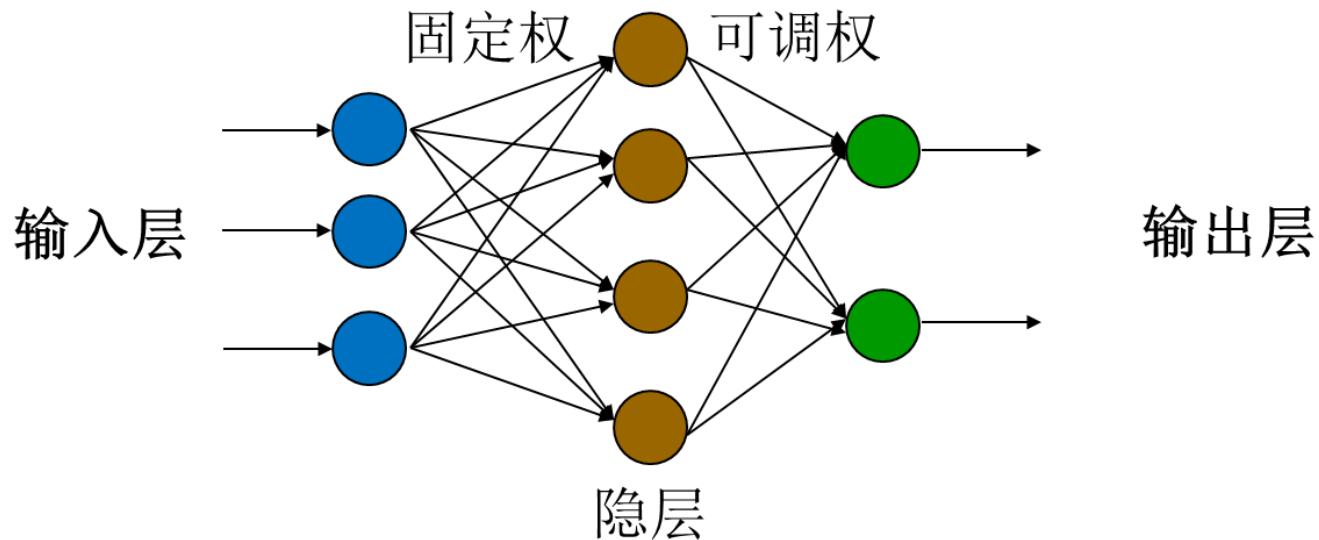
$$y = f(\sigma) = \begin{cases} 1 & \sigma \geq 0 \\ -1 & \sigma < 0 \end{cases}$$

# 感知器

## 感知器:多层前馈神经网络

如果在输入和输出层间加上一层或多层的神经元(隐层神经元),就可构成多层前向网络,这里称为多层感知器。

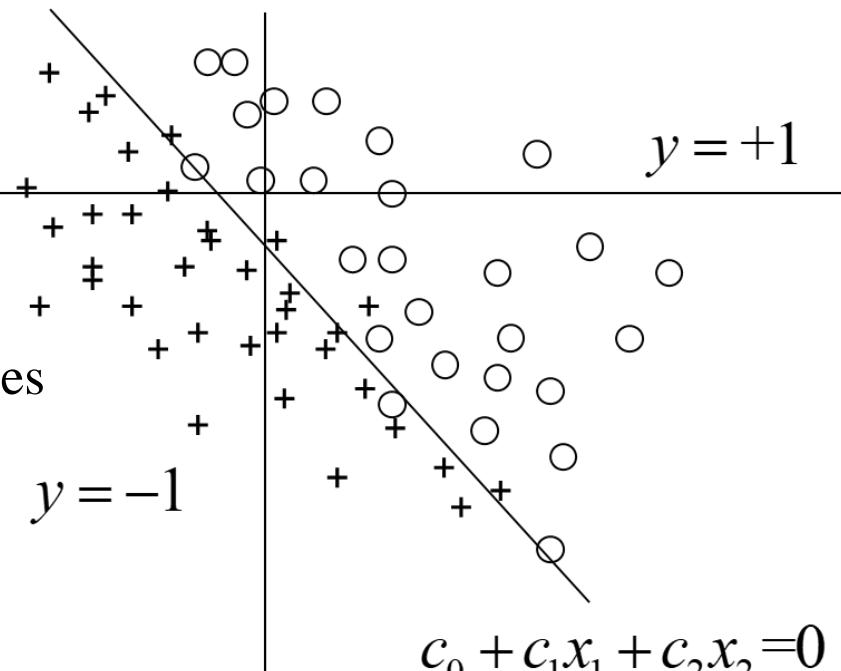
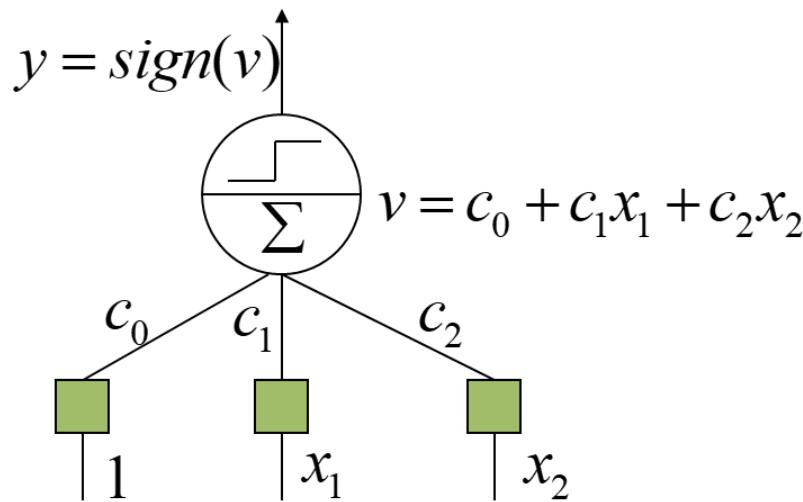
这里需指出的是:多层感知器只允许调节一层的连接权。这是因为按感知器的概念,无法给出一个有效的多层次感知器学习算法。



# 感知器

## 感知器的表征能力

- Rosenblatt (1962)
- Linear separation
- Inputs :Vector of real values
- Outputs :1 or -1



- $Err = T - O$

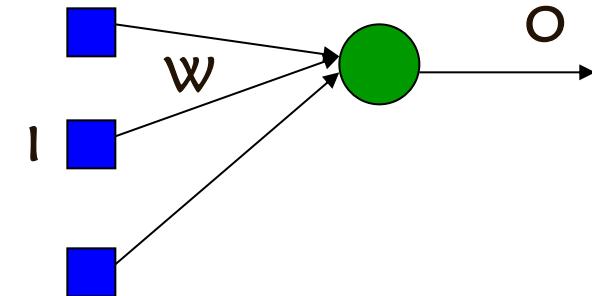
- $O$  是预测得到的输出

- $T$  是实际值,即教师信号

- $W_j \leftarrow W_j + \alpha \times I_j \times Err$

- $I_j$  是第j个输入节点的输入值

- $\alpha$  是一个常数,表示学习率



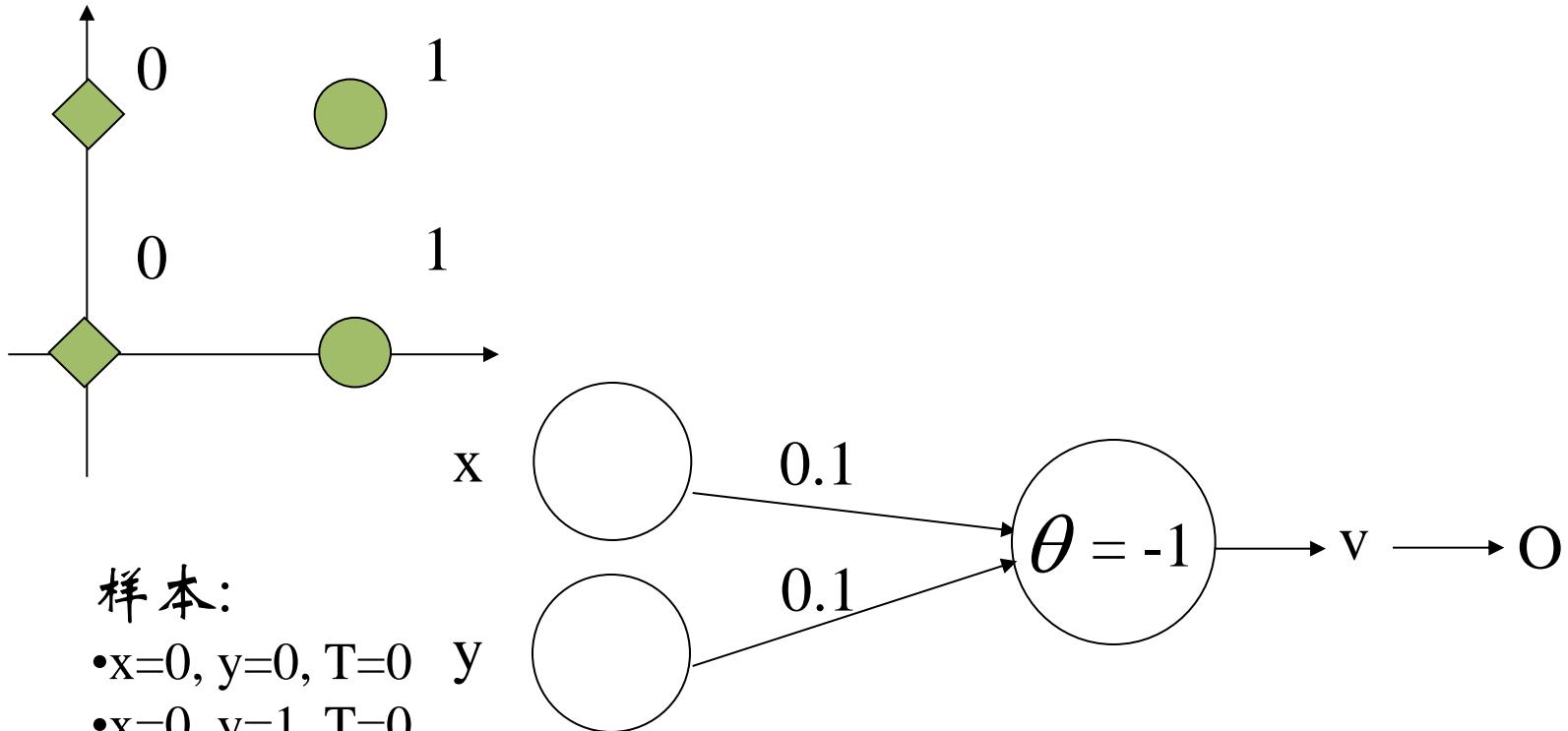
$$O = f(\sigma) = f\left(\sum_{i=1}^n w_i I_i - \theta\right) = f\left(\sum_{i=0}^n w_i I_i\right)$$

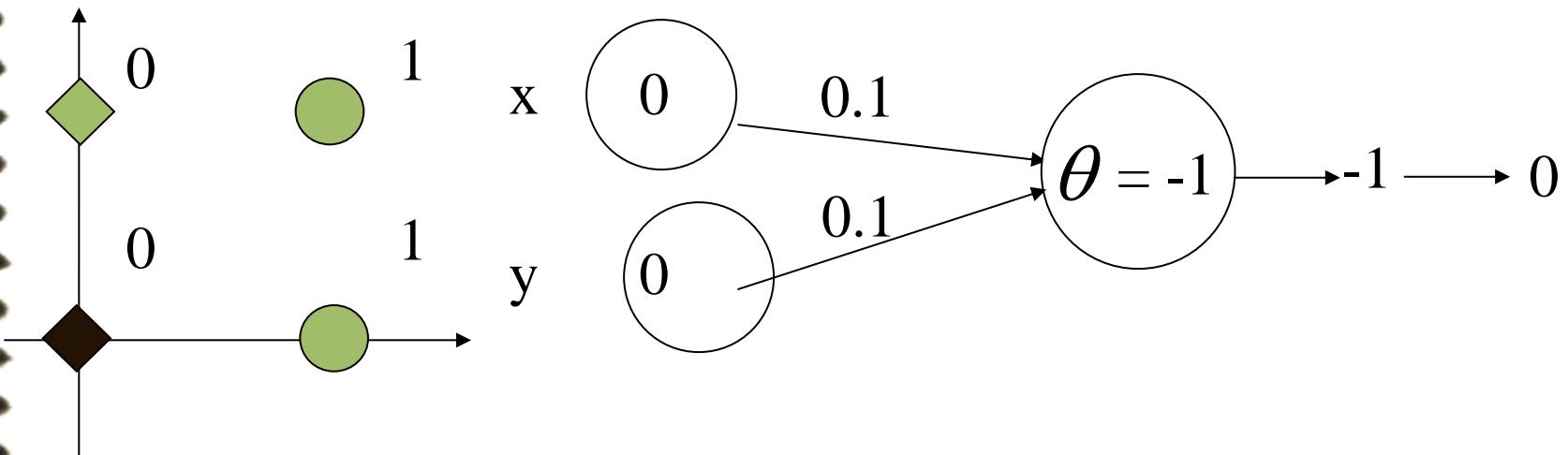
$$f(\sigma) = \begin{cases} 1 & \sigma \geq 0 \\ -1 & \sigma < 0 \end{cases}$$

# 感知器学习过程

- 随机选取权值W的初始值 (between 0-1)
- 将样本数据中的输入值输入到感知器的输入节点
- 得到网络的输出值 O，根据学习公式，由O与T的差，即误差信号来调整网络权值 W
- 如果误差小于给定阈值，或运行次数达到限定次数则停止；否则转2，反复运行。

# 一个简单例子



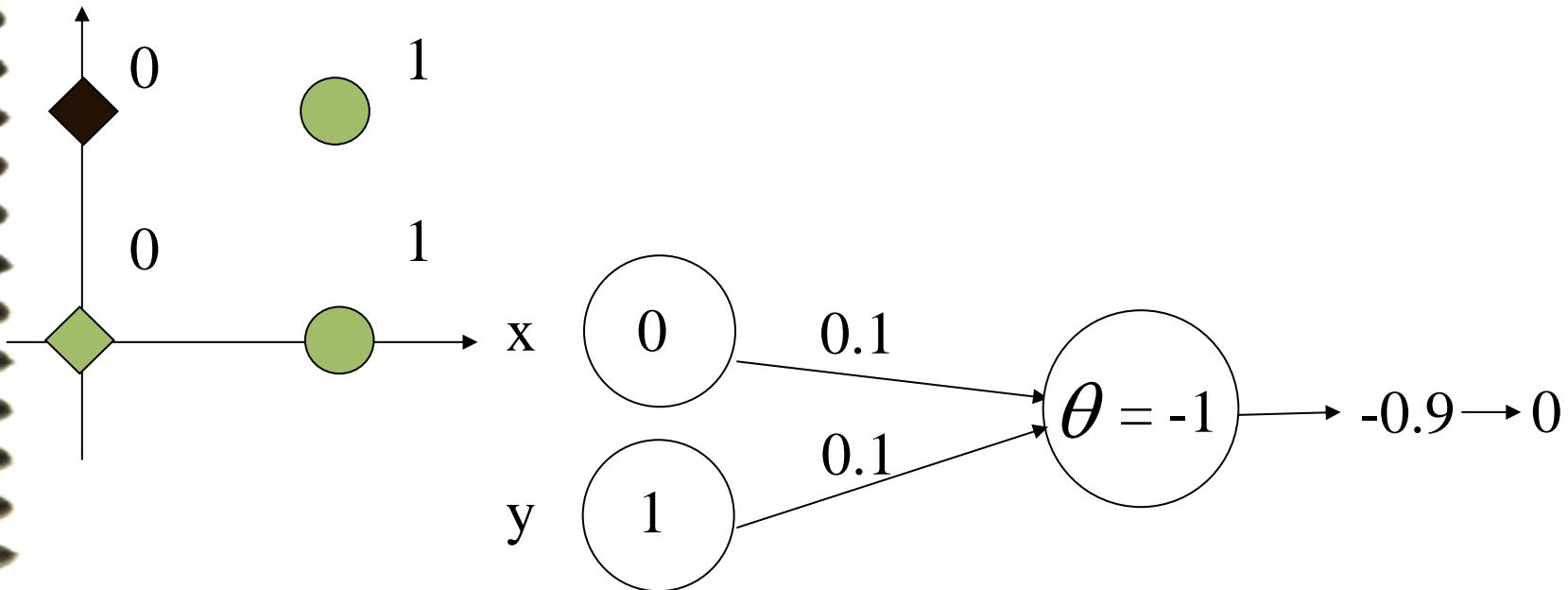


• $x=0, y=0$ 代入:  $v=0 \times 0.1+0 \times 0.1-1=-1; O=0; Err=T-O=0$

$$W1=W1+\alpha \times I1 \times Err=0.1;$$

$$W2=W2+\alpha \times I2 \times Err=0.1;$$

$$\underline{W0=W0+\alpha \times 1 \times Err=-1};$$

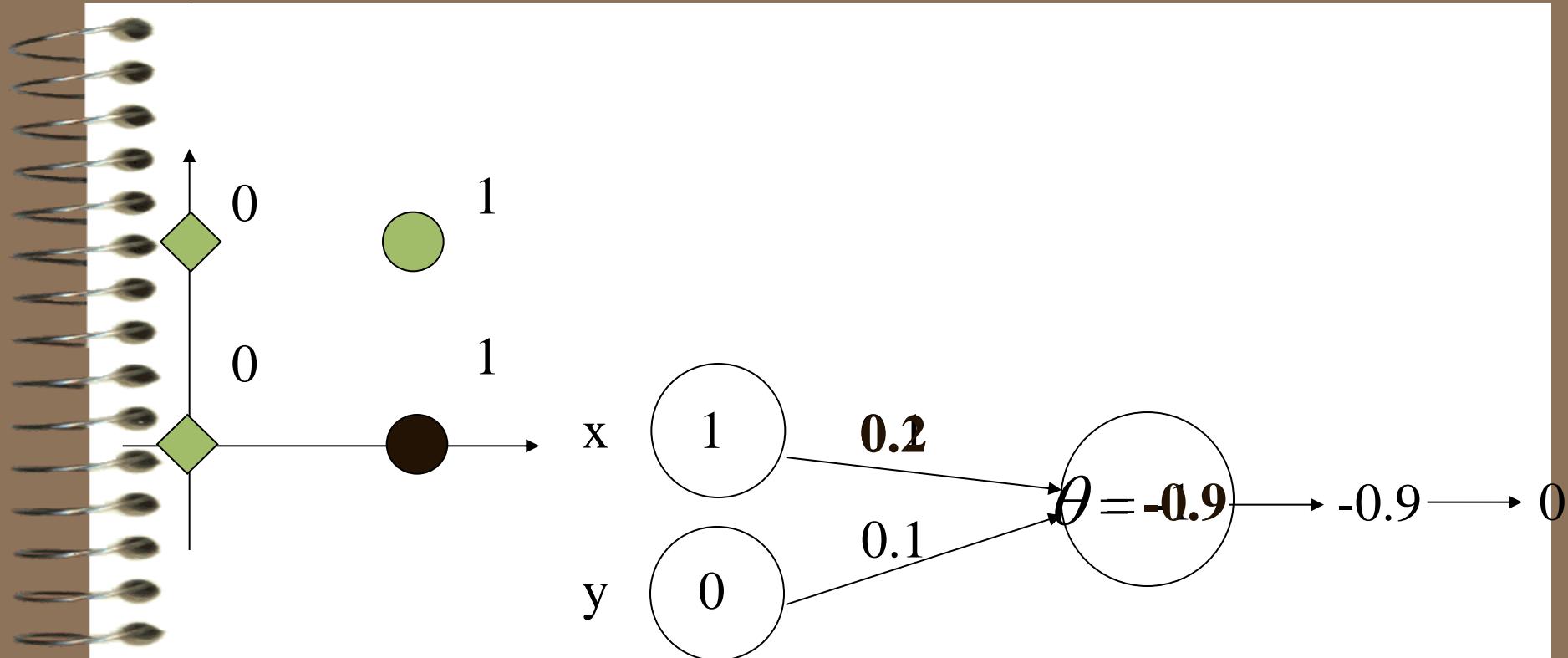


• $x=0, y=1$ 代入:  $v=0*0.1+1*0.1-1=-0.9; O=0; Err=T-O=0$

$$W1=W1+\alpha * I1*Err=0.1;$$

$$W2=W2+\alpha * I2*Err=0.1;$$

$$\underline{W0=W0+\alpha * 1*Err=-1;}$$

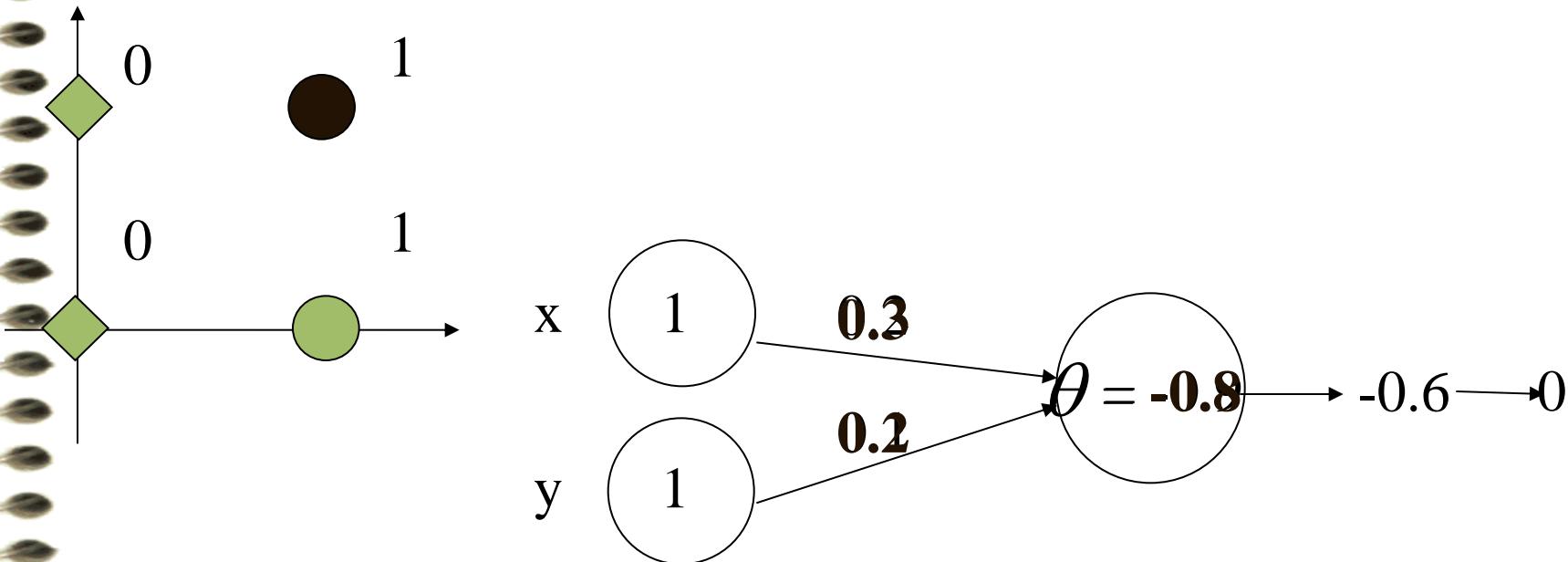


• $x=1, y=0$ 代入:  $v=1*0.1+0*0.1-1=-0.9; O=0; Err=T-O=1$

$$W1=W1 + \alpha * I1 * Err = 0.1 + 0.1 * 1 * 1 = 0.2;$$

$$W2=W2 + \alpha * I2 * Err = 0.1 + 0.1 * 0 * 1 = 0.1;$$

~~$$W0=W0 + \alpha * 1 * Err = 1 + 0.1 * 1 * 1 = 0.9;$$~~

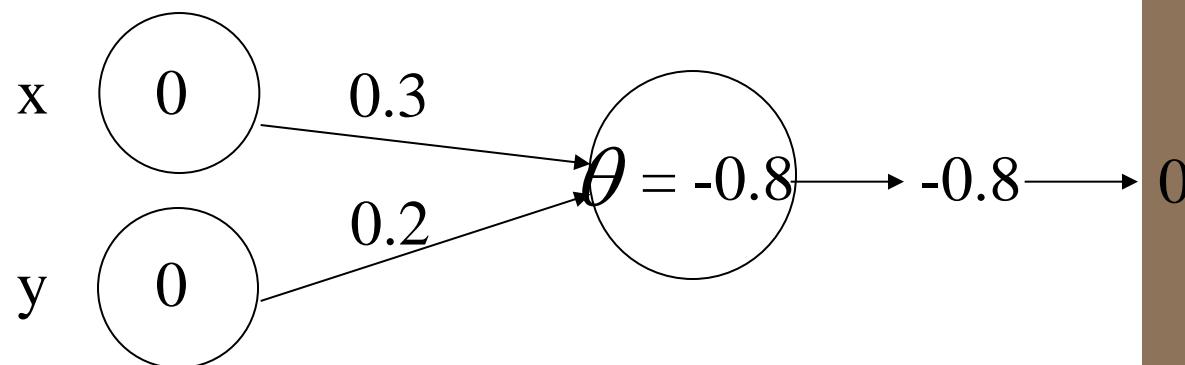
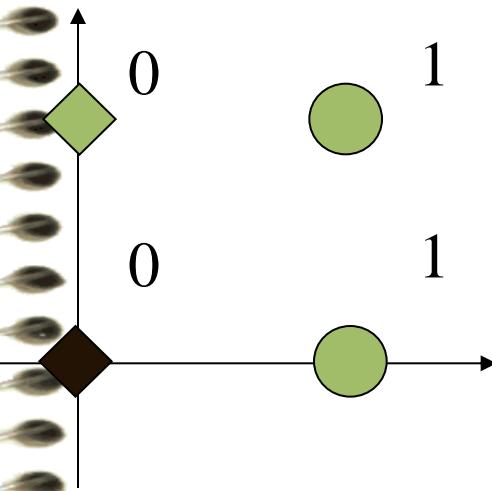


• $x=1, y=1$ 代入:  $v=1*0.2+1*0.1-0.9=-0.6$ ;  $O=0$ ;  $Err=T-O=1$

$$W1=W1 + \alpha * I1 * Err = 0.2 + 0.1 * 1 * 1 = 0.3;$$

$$W2=W2 + \alpha * I2 * Err = 0.1 + 0.1 * 1 * 1 = 0.2;$$

$$\underline{W0=W0 + \alpha * 1 * Err = -0.9 + 0.1 * 1 * 1 = -0.8;}$$

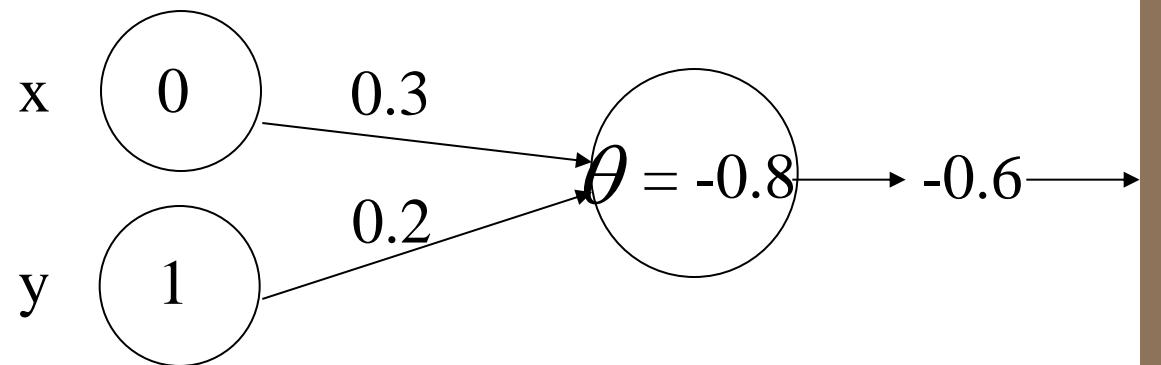
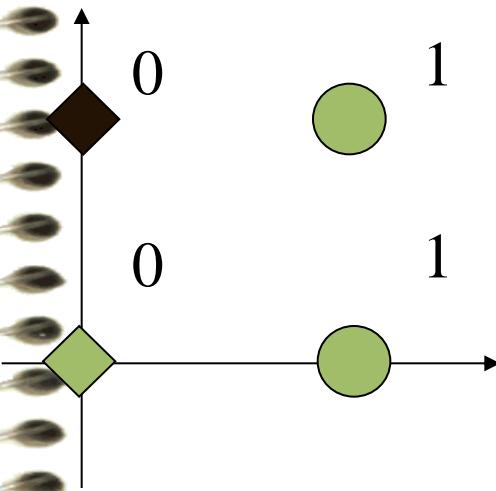


• $x=0, y=0$ 代入:  $v=0*0.3+0*0.2-0.8=-0.8; O=0; Err=T-O=0$

$$W1=W1 + \alpha * I1 * Err = 0.3 + 0.1 * 0 * 0 = 0.3;$$

$$W2=W2 + \alpha * I2 * Err = 0.2 + 0.1 * 0 * 0 = 0.2;$$

~~$$W0=W0 + \alpha * 1 * Err = -0.8 + 0.1 * 1 * 0 = -0.8;$$~~

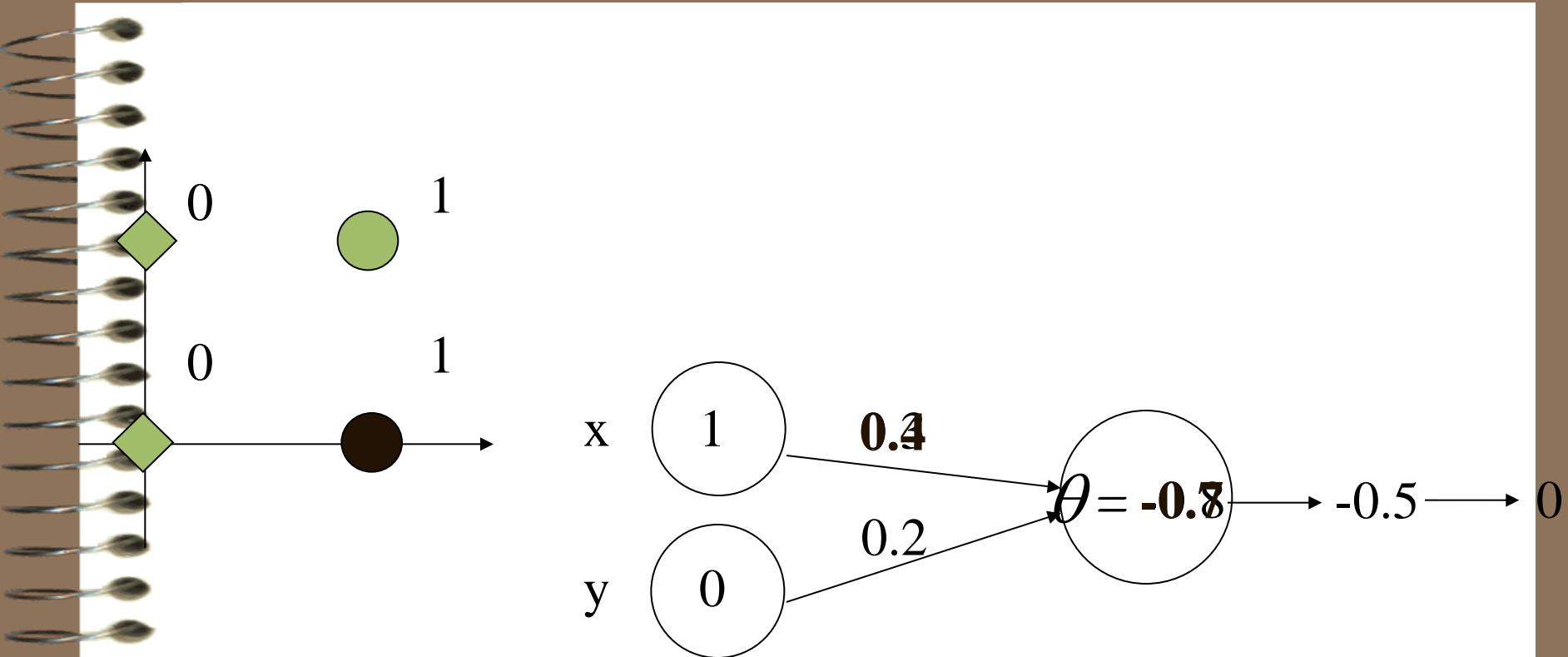


$x=0, y=1$ 代入:  $v=0*0.3+1*0.2-0.8=-0.6; O=0; Err=T-O=0$

$$W1=W1 + \alpha * I1 * Err = 0.3 + 0.1 * 0 * 0 = 0.3;$$

$$W2=W2 + \alpha * I2 * Err = 0.2 + 0.1 * 1 * 0 = 0.2;$$

~~$$W0=W0 + \alpha * 1 * Err = -0.8 + 0.1 * 1 * 0 = -0.8;$$~~

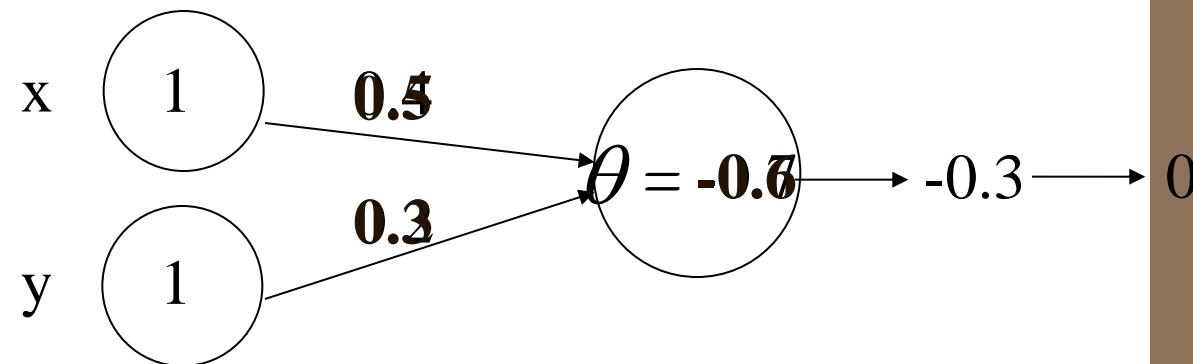
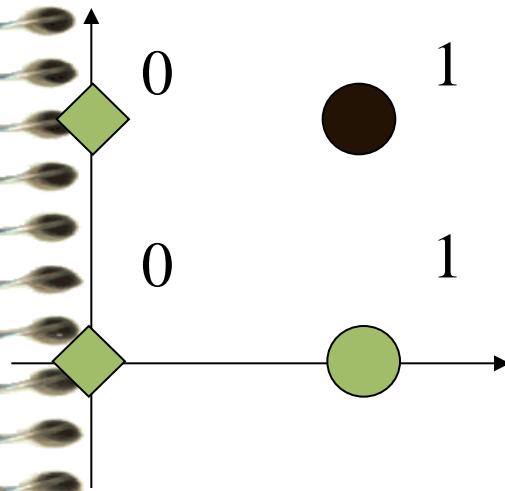


$x=1, y=0$ 代入:  $v=1*0.3+0*0.1-0.8=-0.5; O=0; Err=T-O=1$

$$W1=W1+\alpha * I1*Err=0.3+0.1*1*1=0.4;$$

$$W2=W2+\alpha * I2*Err=0.2+0.1*1*0=0.2;$$

~~$$W0=W0+\alpha * 1*Err=-0.8+0.1*1*1=-0.7;$$~~

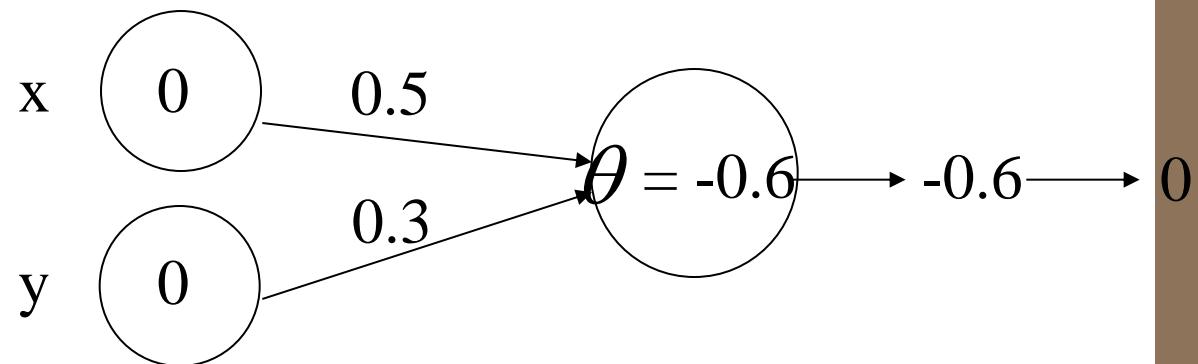
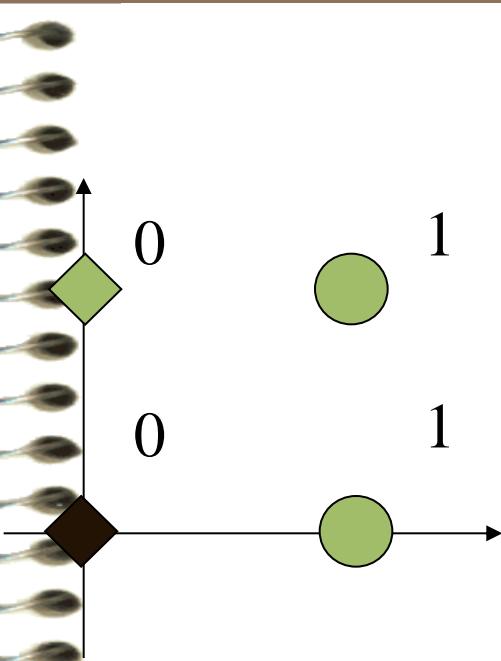


$x=1, y=1$ 代入:  $v=1*0.4+1*0.2-0.9=-0.3; O=0; Err=T-O=1$

$$W1=W1 + \alpha * I1 * Err = 0.4 + 0.1 * 1 * 1 = 0.5;$$

$$W2=W2 + \alpha * I2 * Err = 0.2 + 0.1 * 1 * 1 = 0.3;$$

~~$$W0=W0 + \alpha * 1 * Err = -0.7 + 0.1 * 1 * 1 = -0.6;$$~~

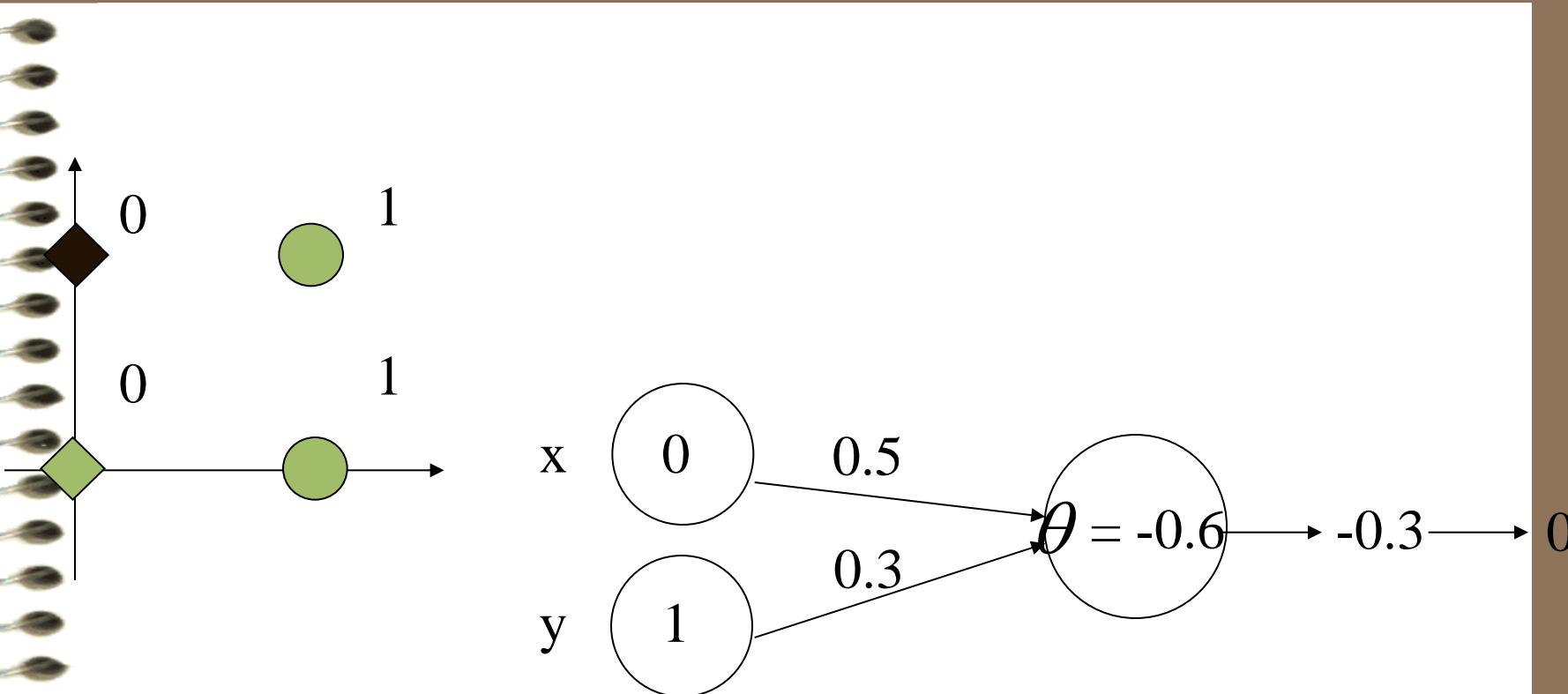


$x=0, y=0$ 代入:  $v=0*0.5+0*0.3-0.6=-0.6$ ;  $O=0$ ;  $Err=T-O=0$

$W1=W1+\alpha * I1*Err=0.5;$

$W2=W2+\alpha * I2*Err=0.3;$

~~$W0=W0+\alpha * 1*Err=-0.6;$~~

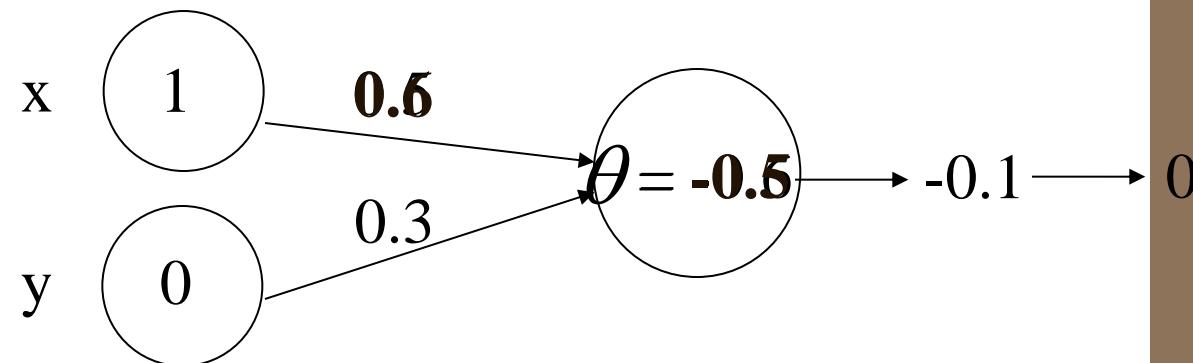
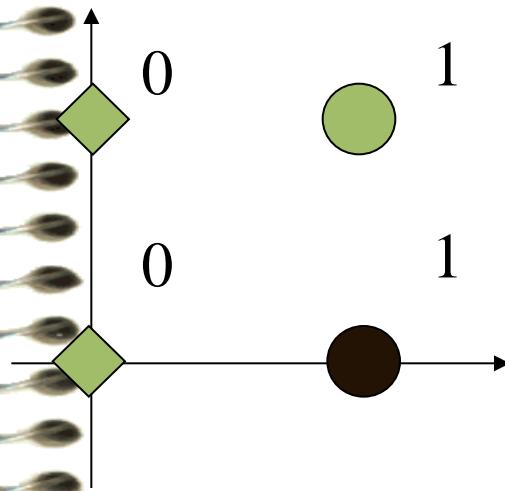


$x=0, y=1$ 代入:  $v=0*0.5+1*0.3-0.6=-0.3; O=0; Err=T-O=0$

$W1=W1+\alpha * I1*Err=0.5;$

$W2=W2+\alpha * I2*Err=0.3;$

~~$W0=W0+\alpha * 1*Err=-0.6;$~~

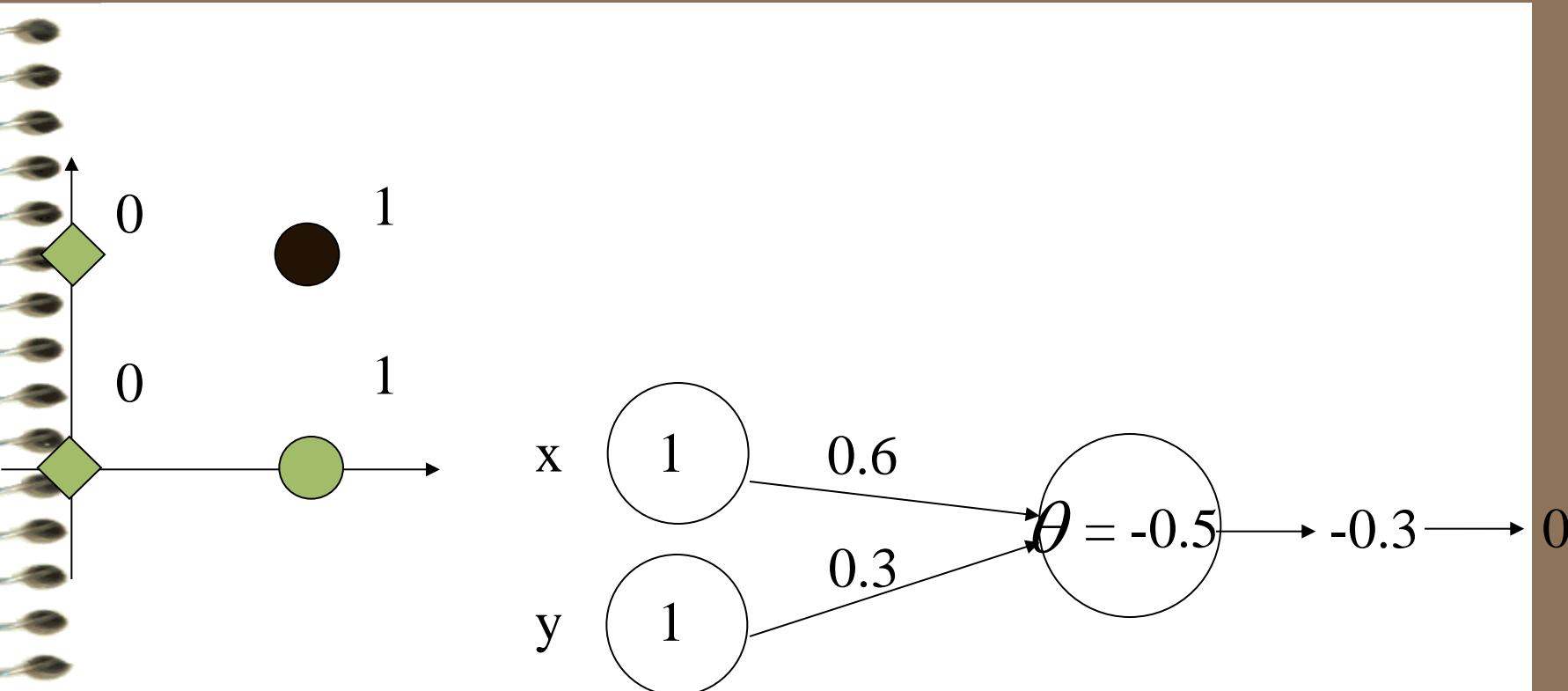


$x=1, y=0$ 代入:  $v=1*0.5+0*0.3-0.6=-0.1; O=0; Err=T-O=1$

$$W1=W1 + \alpha * I1 * Err = 0.5 + 0.1 * 1 * 1 = 0.6;$$

$$W2=W2 + \alpha * I2 * Err = 0.3 + 0.1 * 1 * 0 = 0.3;$$

~~$$W0=W0 + \alpha * 1 * Err = -0.6 + 0.1 * 1 * 1 = -0.5;$$~~

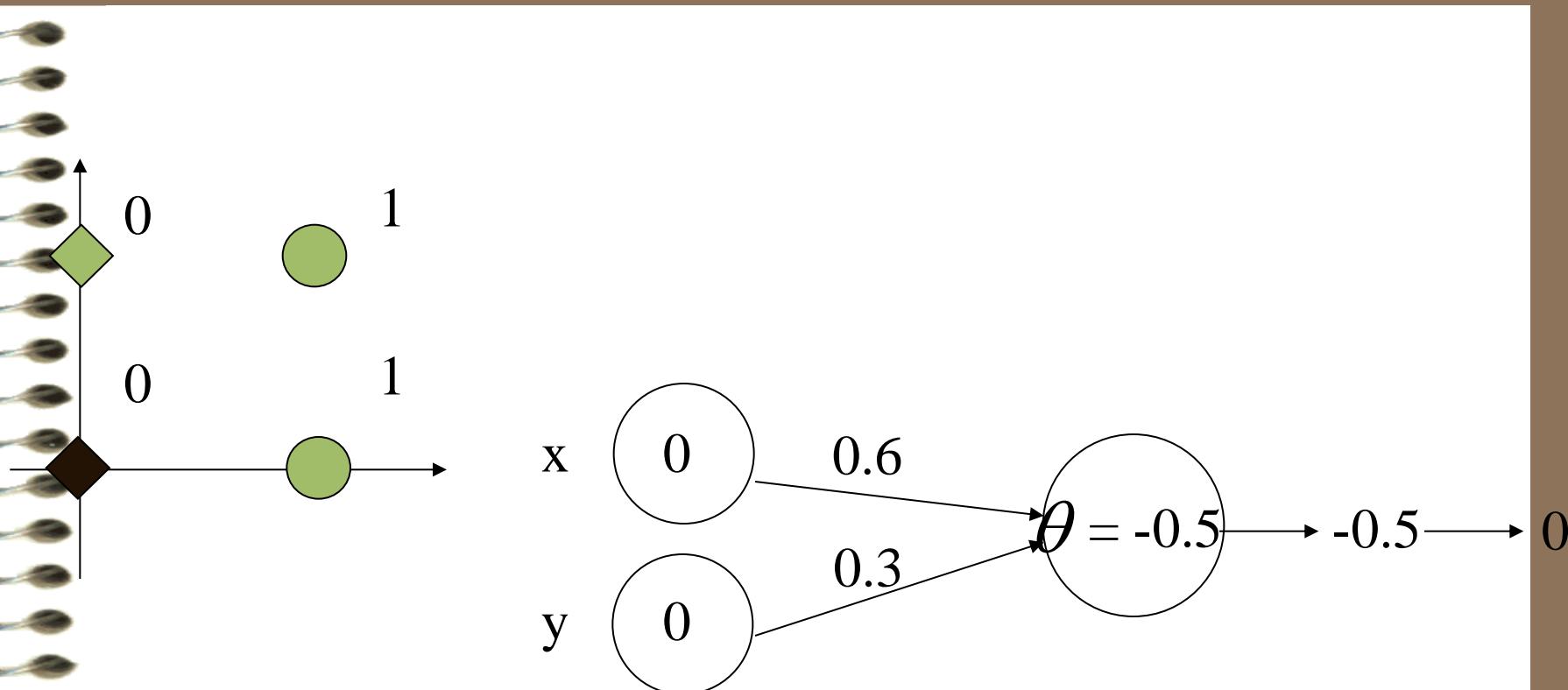


$x=1, y=1$ 代入:  $v=1*0.6+1*0.3-0.5=0.4; O=1; Err=T-O=0$

$$W1=W1 + \alpha * I1 * Err = 0.6;$$

$$W2=W2 + \alpha * I2 * Err = 0.3;$$

$$W0=W0 + \alpha * 1 * Err = -0.5;$$

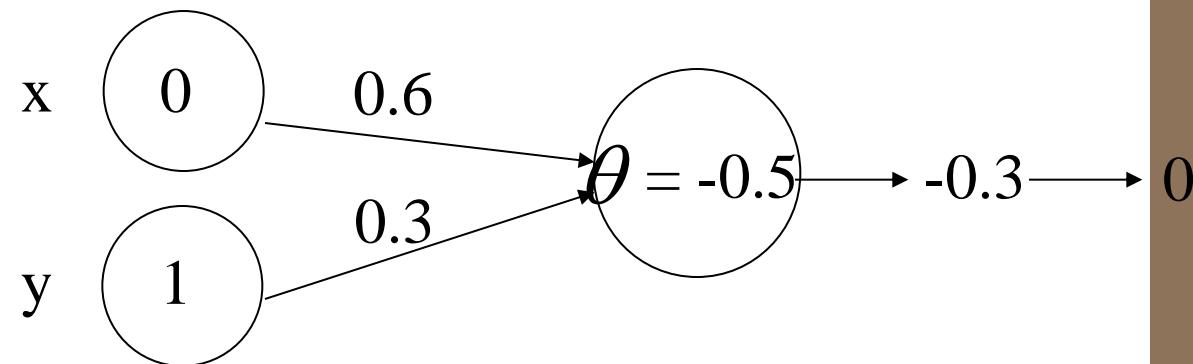
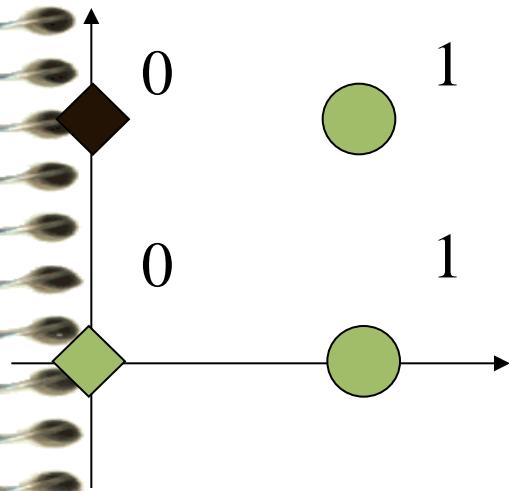


$x=0, y=0$ 代入:  $v=0*0.6+0*0.3-0.5=-0.5; O=0; Err=T-O=0$

$W1=W1+\alpha * I1*Err=0.6;$

$W2=W2+\alpha * I2*Err=0.3;$

$W0=W0+\alpha * 1*Err=-0.5;$

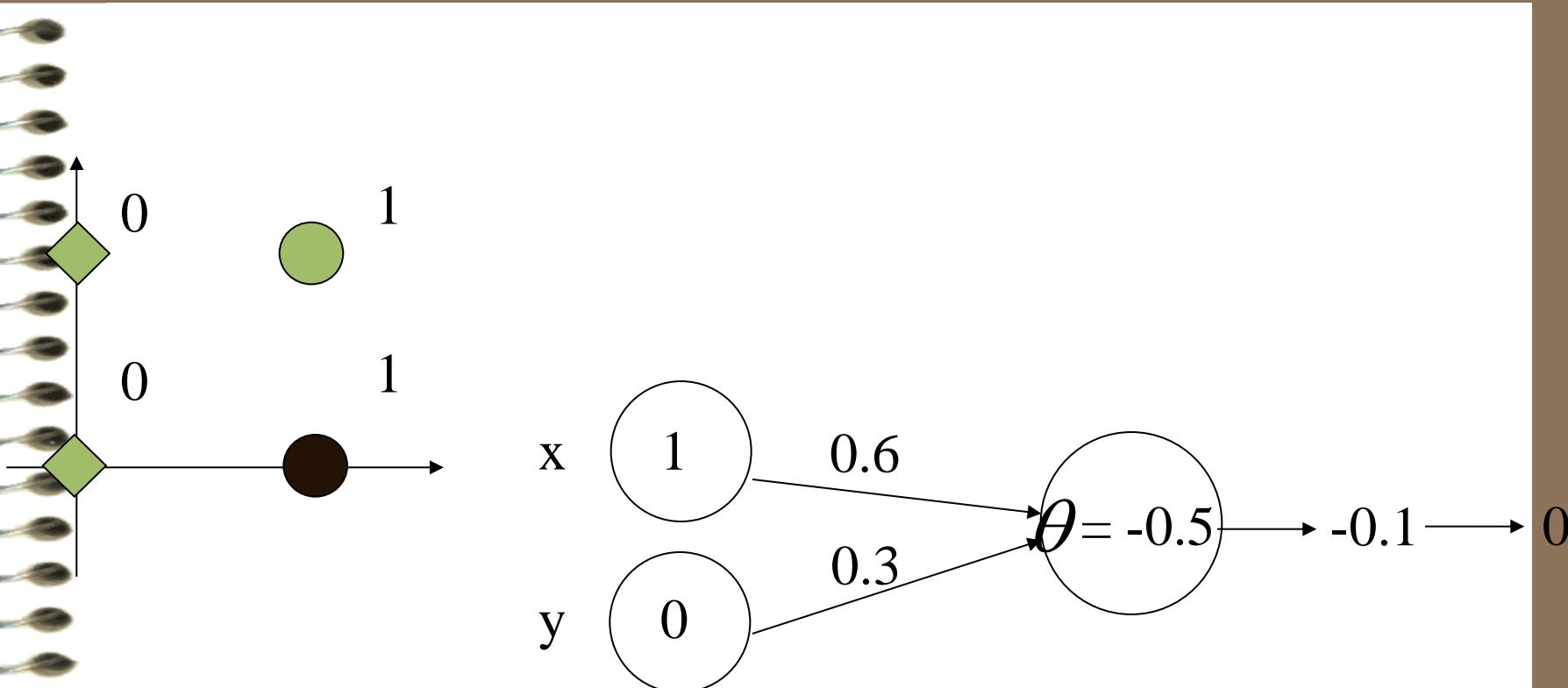


$x=0, y=1$ 代入:  $v=0*0.6+1*0.3-0.5=-0.2; O=0; Err=T-O=0$

$W1=W1+\alpha * I1*Err=0.6;$

$W2=W2+\alpha * I2*Err=0.3;$

$W0=W0+\alpha * 1*Err=-0.5;$

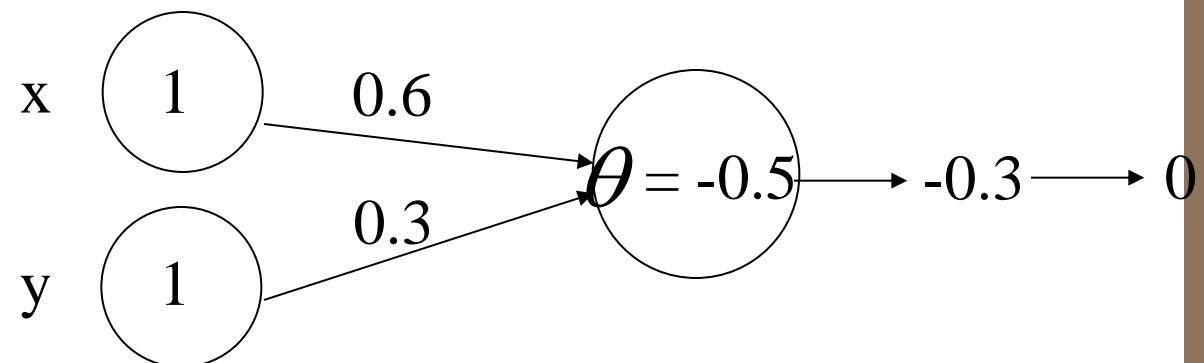
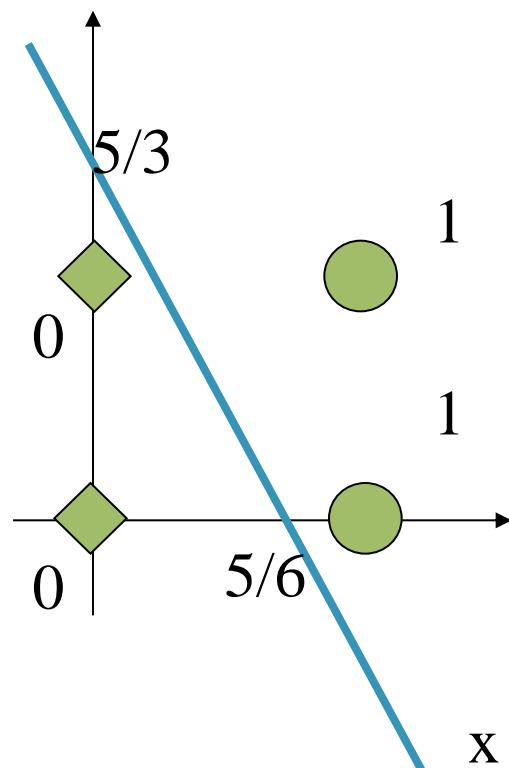


$x=1, y=0$ 代入:  $v=1*0.6+0*0.3-0.5=0.1; O=1; Err=T-O=0$

$W1=W1+\alpha * I1*Err=0.6;$

$W2=W2+\alpha * I2*Err=0.3;$

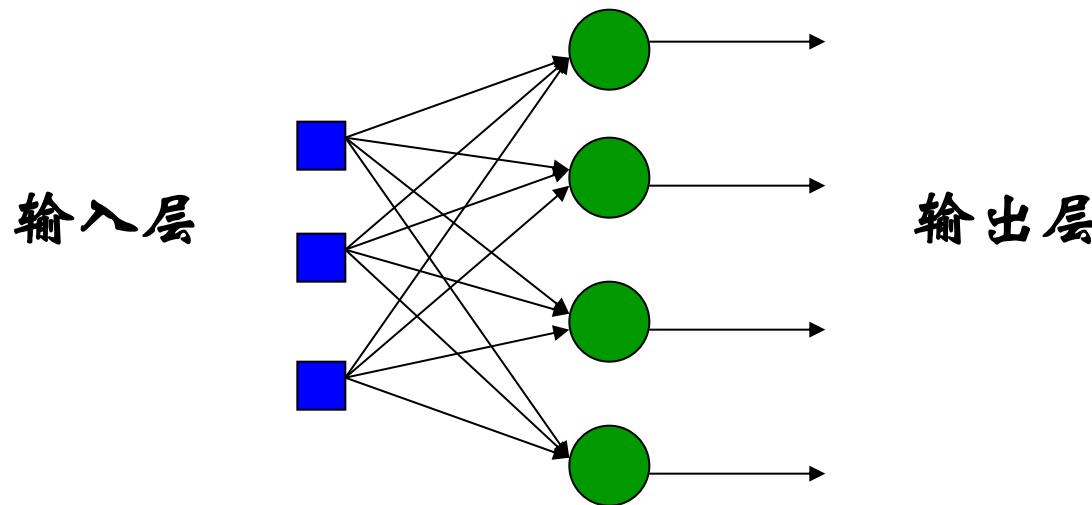
$W0=W0+\alpha * 1*Err=-0.5;$



$$V = 0.6*x + 0.3*y - 0.5$$

$$O = \text{sign}(v) = \text{sign}(0.6*x + 0.3*y - 0.5)$$

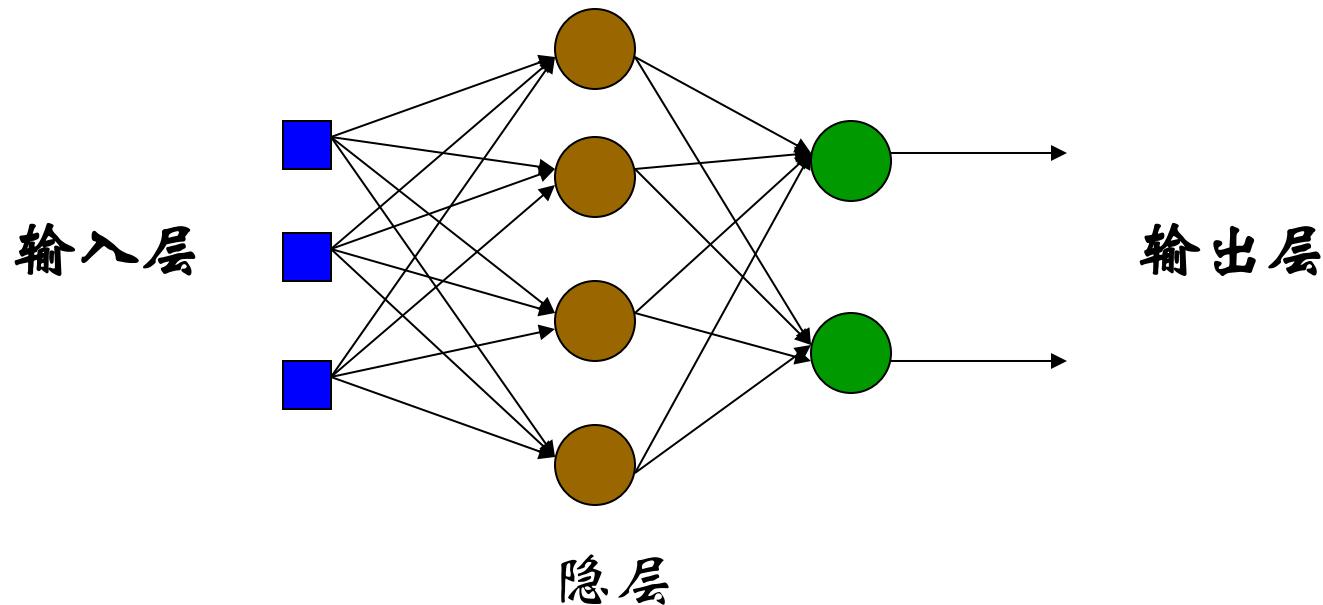
# 感知器: 单层前馈神经网络



$$y = f(\sigma) = \begin{cases} 1 & \sigma \geq 0 \\ -1 & \sigma < 0 \end{cases}$$

# 感知器: 多层前馈神经网络

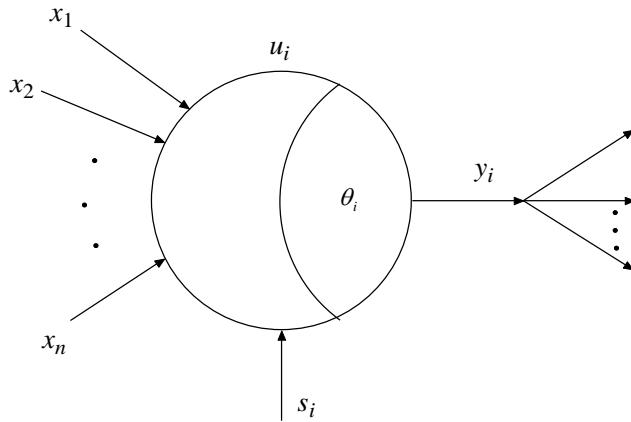
3-4-2 Network



# 逻辑回归与感知器的关系

# 逻辑回归与感知器

- 感知器模型



$$\sigma_i = \sum x_i w_{ij} - \theta_i$$

$$y_i = f(\sigma_i)$$

$$f(\sigma) = \begin{cases} 1 & \sigma \geq 0 \\ 0 & \sigma < 0 \end{cases}$$

# 逻辑回归与感知器的关系

- Digression: The perceptron learning algorithm
  - Consider modifying the logistic regression method to “**force**” it to output values that are either 0 or 1
$$g(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{if } z < 0 \end{cases}$$
  - If we then let  $h_{\theta}(x) = g(\theta^T x)$  as before but using this modified definition of  $g$ , and if we use the update rule:
$$\theta_j := \theta_j + \alpha(y^{(i)} - h_{\theta}(x^{(i)}))x_j^{(i)}$$
then we have the **perceptron learning algorithm**.

- In the 1960s, this “perceptron” was argued to be a rough model for how **individual neurons** in the brain work.

# 感知器的表征能力

# 感知器的表征能力

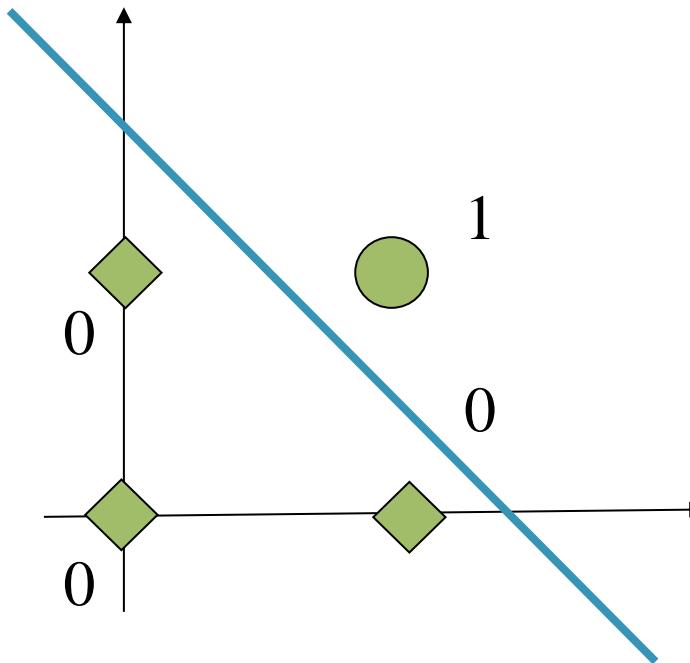
- 感知器看作是n维实例空间（即点空间）中的超平面决策面
- 对于超平面一侧的实例，感知器输出1，对于另一侧的实例，输出-1
- 决策超平面方程是  $\vec{w} \cdot \vec{x} = 0$
- 可以被某个超平面分割的样例集合，称为线性可分样例集合

# 感知器的表征能力

- 为什么前述更新法则会成功收敛到正确的权值呢?
  - 可以证明 (Minsky & Papert 1969)
    - ✓ 如果训练样例线性可分，并且使用了充分小的 $\eta$
    - ✓ 否则，不能保证

# 感知器的表征能力

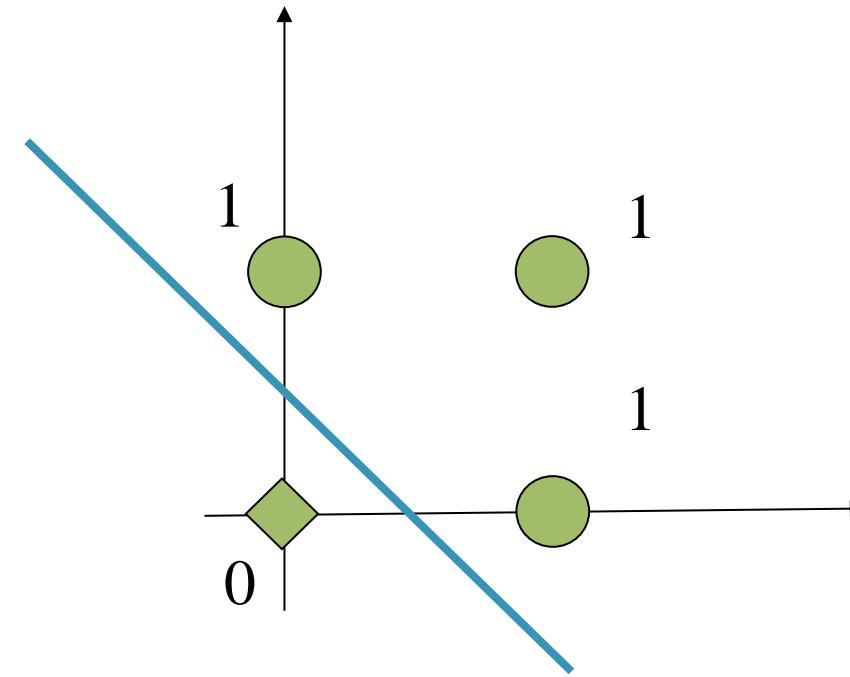
AND



$$O = \text{sign}(v) = \text{sign}(0.5*x + 0.5*y - 0.8)$$

# 感知器的表征能力

OR



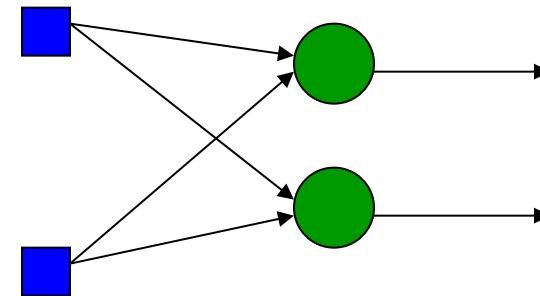
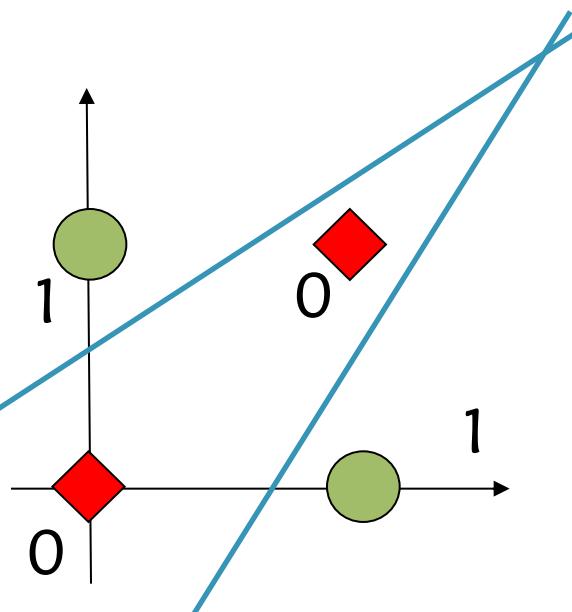
$$O = \text{sign}(v) = \text{sign}(0.5*x + 0.5*y - 0.3)$$

# 感知器的表征能力

- 线性不可分情况

XOR

$$\begin{aligned} A \oplus B &= (A \wedge \neg B) \vee (\neg A \wedge B) = AB + \bar{A}B \\ &= (A \vee B) \wedge (\neg A \vee \neg B) = (A + B)(\bar{A} + \bar{B}) \\ &= (A \vee B) \wedge \neg(A \wedge B) = (A + B)(\bar{A}\bar{B}) \end{aligned}$$



# 感知器的表征能力

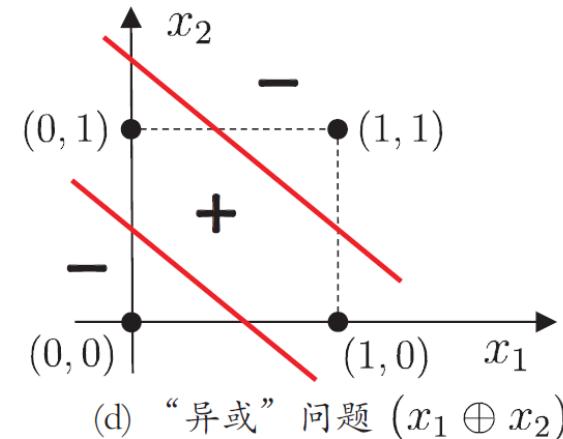
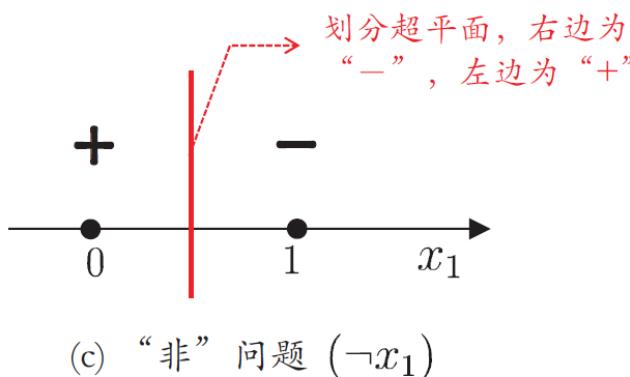
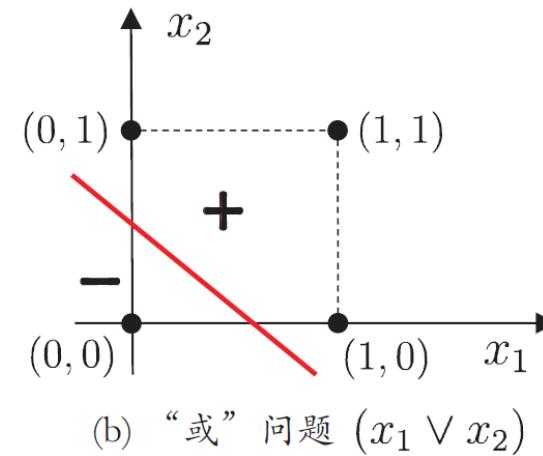
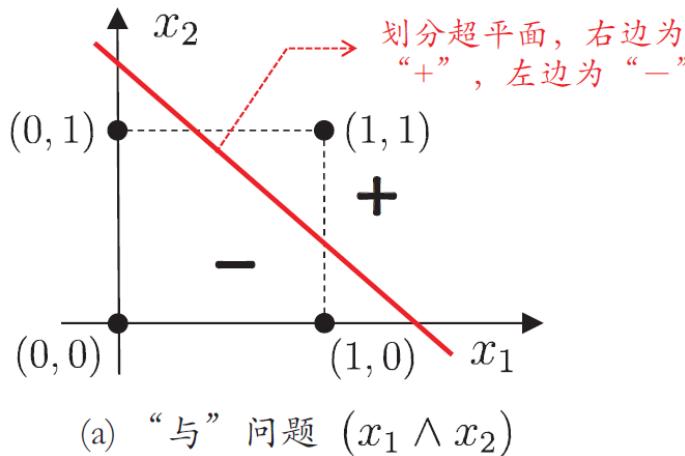
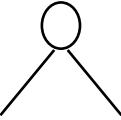
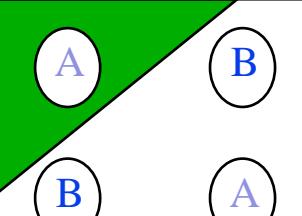
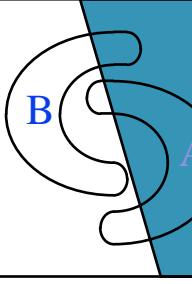
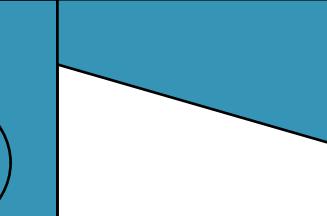
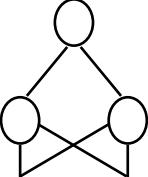
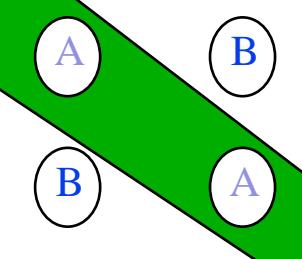
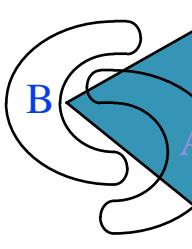
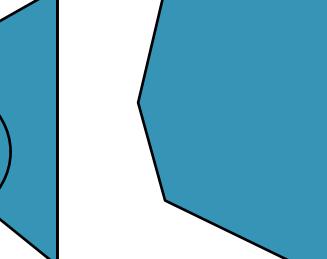
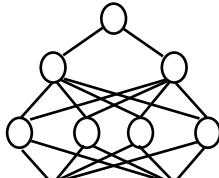
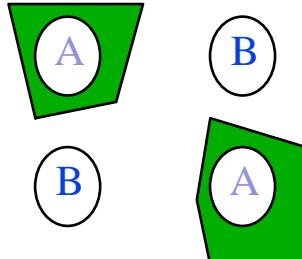
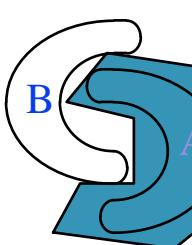
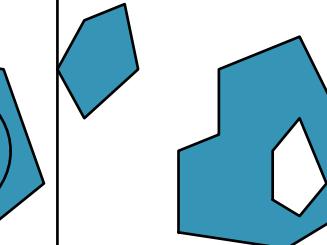
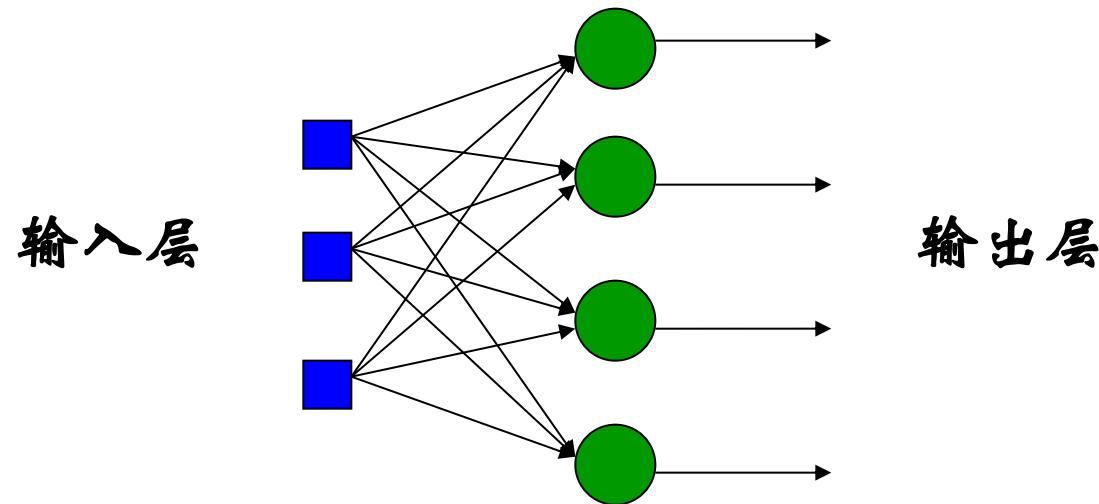


图 5.4 线性可分的“与”“或”“非”问题与非线性可分的“异或”问题

# 感知器的表征能力

<i>Structure</i>	<i>Types of Decision Regions</i>	<i>Exclusive-OR Problem</i>	<i>Classes with Meshed Regions</i>	<i>Most General Region Shapes</i>
<i>Single-Layer</i> 	<i>Half Plane Bounded by Hyperplane</i>			
<i>Two-Layer</i> 	<i>Convex Open or Closed Regions</i>			
<i>Three-Layer</i> 	Arbitrary (Complexity Limited by No. of Nodes)			

## 感知器的学习规则(2)



当取 $f()$ 为线性连续函数时

$$O_i = f(v_i) = \sum_j I_j W_{ij}$$

定义误差函数为：

$$E = \frac{1}{2} \sum_i (T_i - O_i)^2 = \frac{1}{2} \sum_i (T_i - \sum_j I_j W_{ij})^2$$

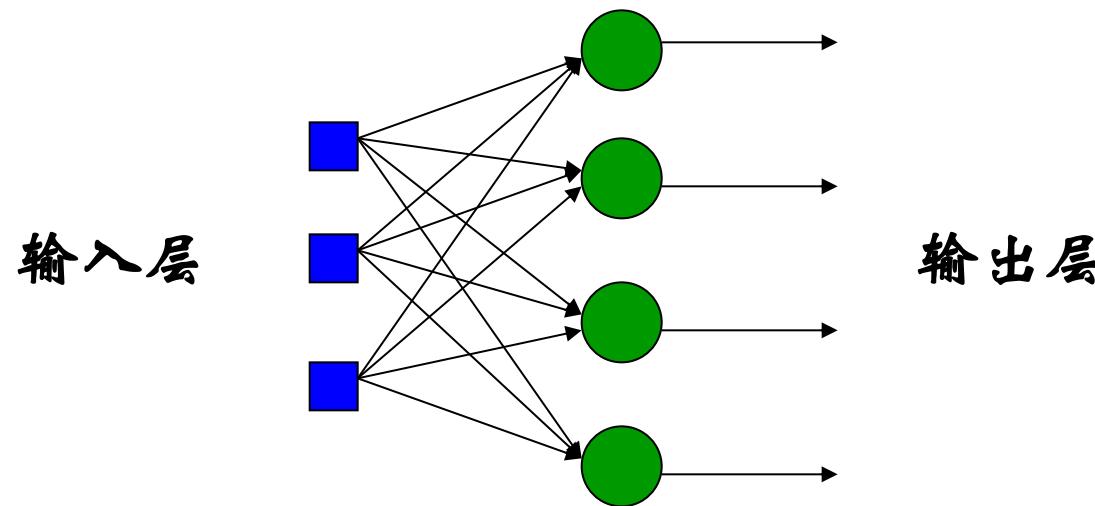
则

$$\frac{\partial E}{\partial w_{ij}} = -(T_j - O_j) \frac{\partial O_j}{\partial w_{ij}} = -(T_j - O_j) I_i$$

则权值更新公式为：

$$w_{ij} = w_{ij} + \eta \cdot (T_j - O_j) \cdot I_i$$

## 感知器的学习规则(3)



当取 $f$ 为sigmoid函数时

$$O_i = f(v_i) = \frac{1}{1 + e^{-v_i}} = \frac{1}{1 + e^{-\sum_j I_j W_{ij}}}$$

则

$$\begin{aligned}\frac{\partial E}{\partial w_{ij}} &= -(T_j - O_j) \frac{\partial O_j}{\partial v_j} \frac{\partial v_j}{\partial w_{ij}} \\&= -(T_j - O_j) \frac{-e^{-v_j}}{(1 + e^{-v_j})^2} \frac{\partial v_j}{\partial w_{ij}} \\&= (T_j - O_j) O_j (1 - O_j) I_i\end{aligned}$$

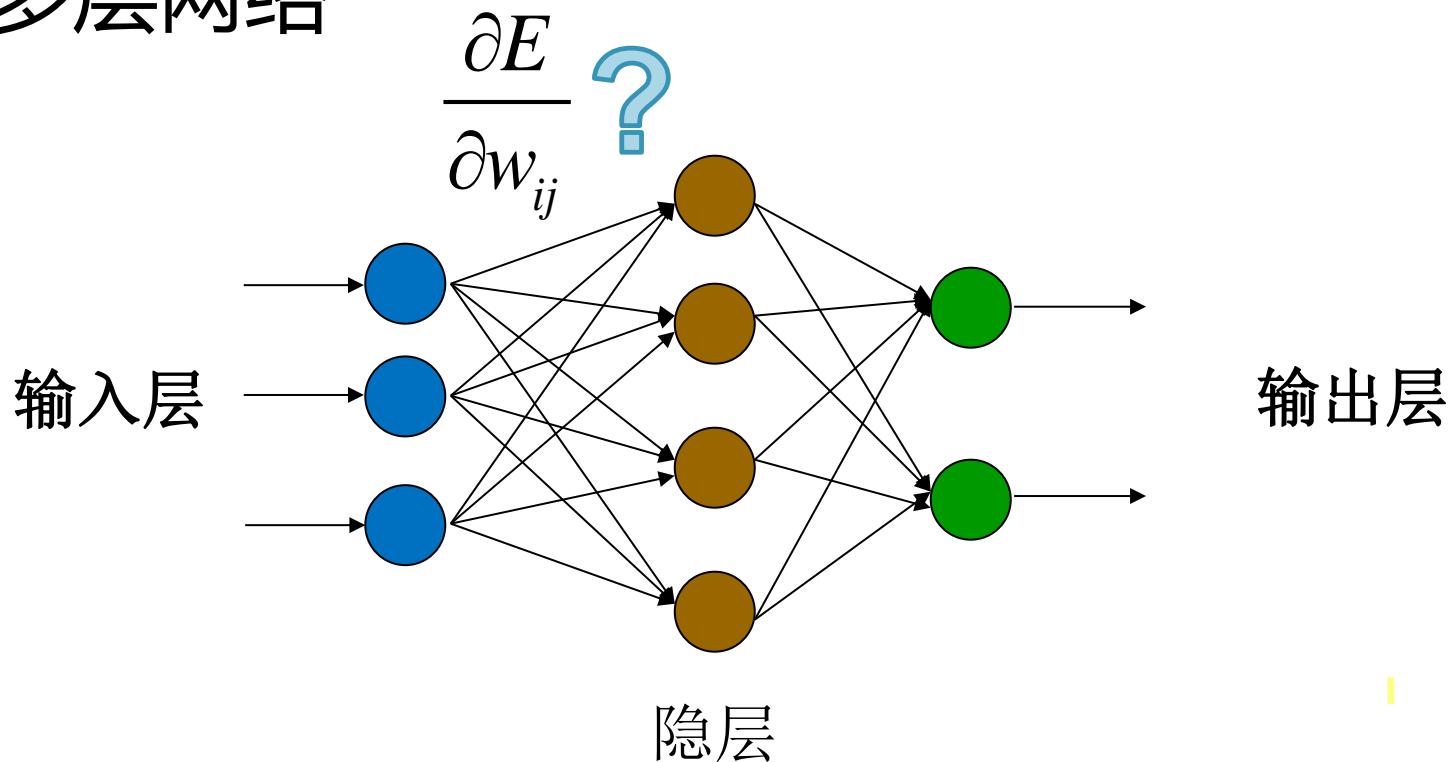
则权值更新公式为：

$$w_{ij} = w_{ij} - \eta \cdot I_i \cdot O_j \cdot (1 - O_j) \cdot (T_j - O_j)$$

# 多层前馈网络BP学习算法

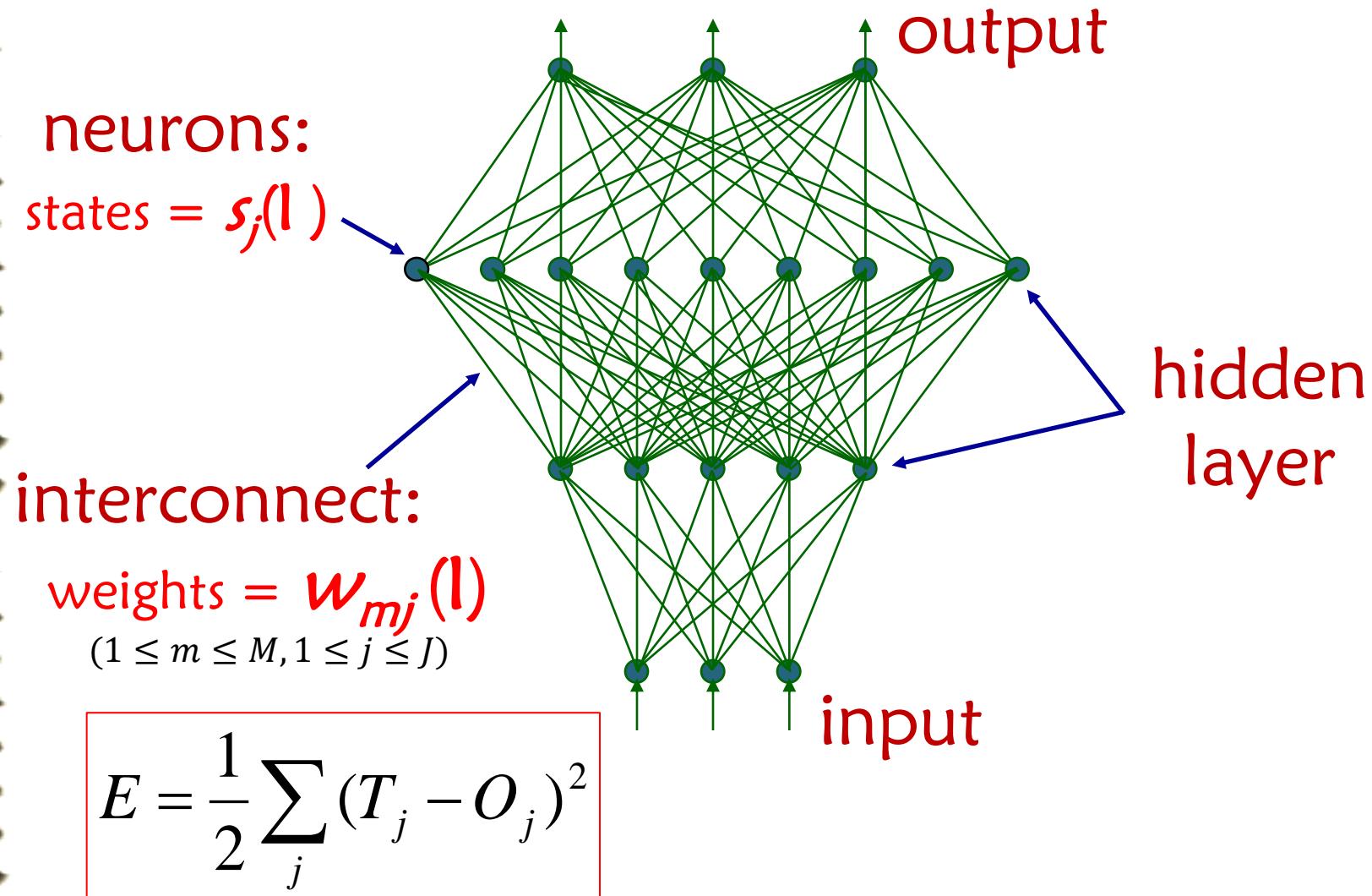
# BP学习算法—多层前馈网络

- 多层网络



$$E = \frac{1}{2} \sum_i (T_i - O_i)^2$$

# BP学习算法—多层前馈网络



# 误差反传过程

Problem: 对于任意权值,  $w_{mj}(l)$ , 更新

$$w_{mj}(I) \leftarrow w_{mj}(I) - \eta \frac{dE}{dw_{mj}(I)}$$

A Solution:

误差反传  
偏微分链式法则

$$\frac{dE}{dw_{mj}(I)} = \frac{dE}{ds_j(I)} \frac{ds_j(I)}{d\vartheta_j(I)} \frac{d\vartheta_j(I)}{dw_{mj}(I)}$$

# 偏导数的计算

(完美的数学!!!)

$$\begin{aligned}\frac{ds_j(I)}{d\vartheta_j(I)} &= \frac{d}{d\vartheta_j(I)} \left( \frac{1}{1 + \exp(-\vartheta_j(I))} \right) \\ &= s_j(I)(1 - s_j(I))\end{aligned}$$

$$\frac{d\vartheta_j(I)}{dw_{mj}(I)} = s_m(I - 1)$$

$$\begin{aligned}\frac{dE}{ds_j(I)} &= \delta_j(I) \\ &= \sum_n \delta_n(I + 1) s_n(I + 1) [1 - s_n(I + 1)] w_{jn}(I + 1)\end{aligned}$$

## 权值更新

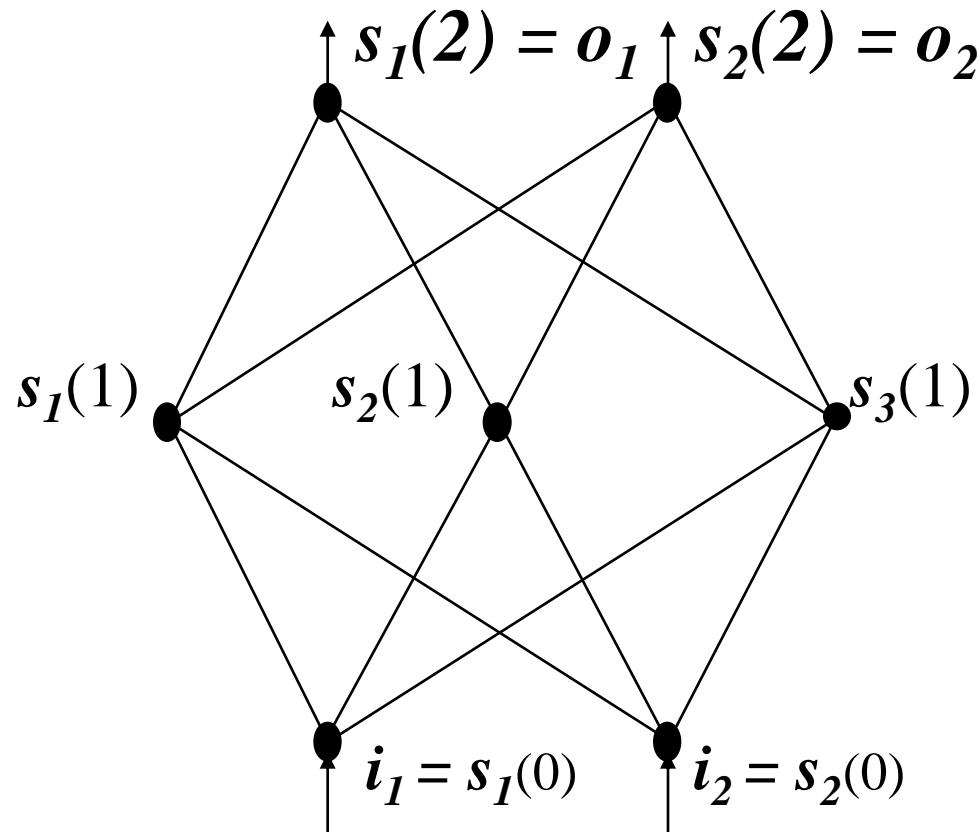
$$w_{mj}(I) \leftarrow w_{mj}(I) - \eta \frac{dE}{dw_{mj}(I)}$$

$$\begin{aligned} \frac{dE}{dw_{mj}(I)} &= \frac{dE}{ds_j(I)} \frac{ds_j(I)}{d\vartheta_j(I)} \frac{d\vartheta_j(I)}{dw_{mj}(I)} \\ &= \delta_j(I) s_j(I)(1-s_j(I)) s_m(I-1) \end{aligned}$$

# BP学习算法—多层前馈网络

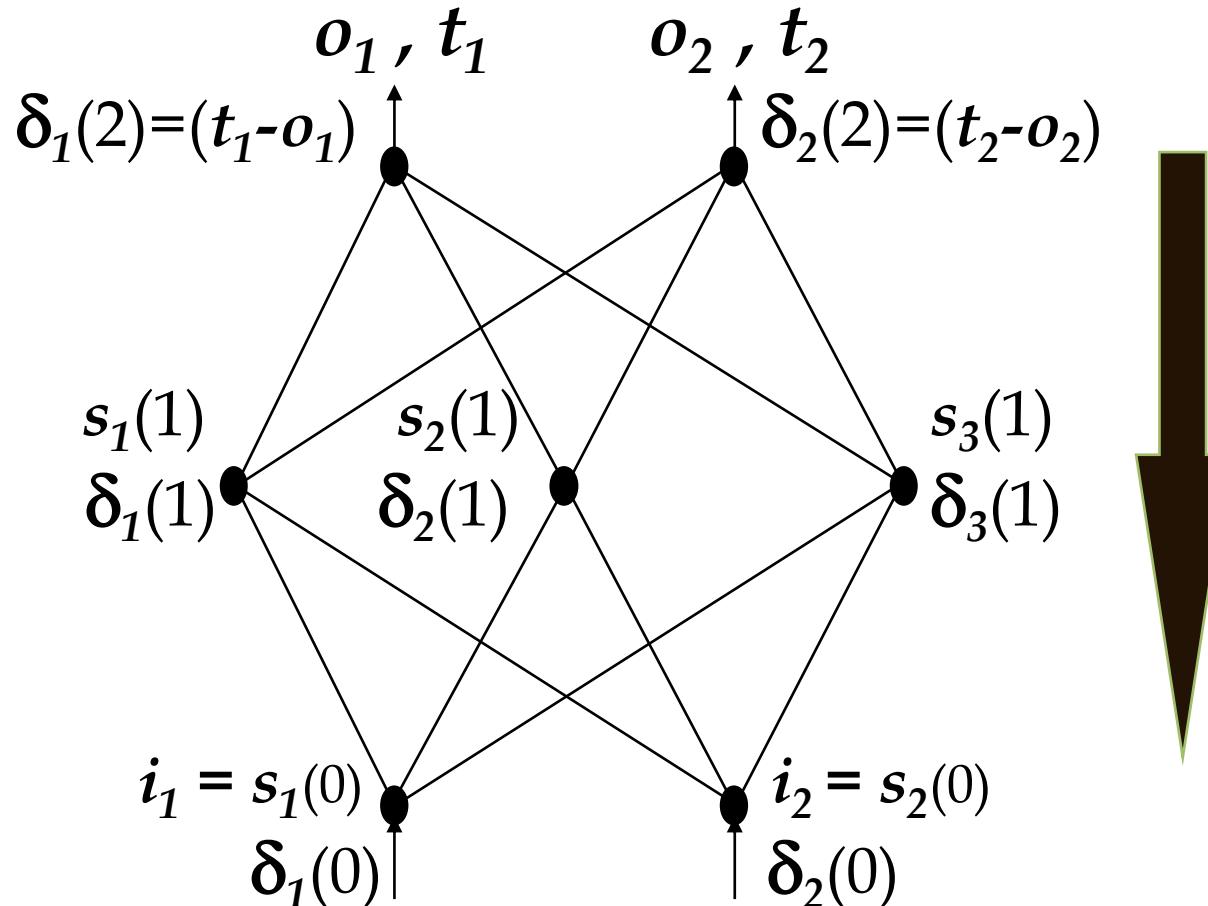
- 总体思想：正向传播 + 反向传播
  - 正向传播时，输入信息从输入层开始，经过各层神经元的处理后产生一个输出。然后，将实际输出和所需输出进行比较，得到一个误差矢量。
  - 反向传播过程，从输出层至输入层，利用这个误差矢量对权值进行逐层修正。
  - 正向传播和反向传播交替迭代进行。
  - 通过不断迭代优化，使得网络的输出逐渐接近目标值，从而实现对输入数据的准确识别或预测。

# 步骤1：数据输入与前向传播



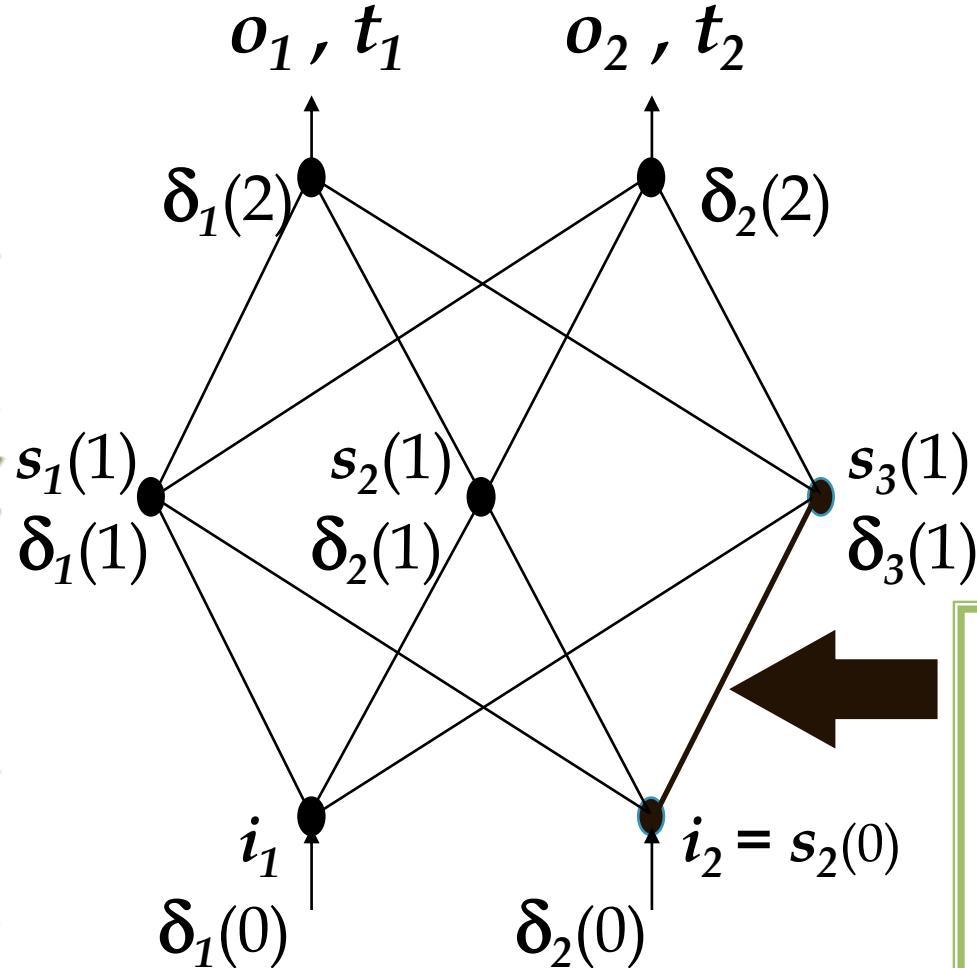
所有神经元  
的状态由其经  
过前一层和接权  
值确定

## 步骤2: 计算输出误差, 反向确定各神经元修正量



各权值修正量由反向传播的输出误差确定

## 步骤3: 更新权值



$$w_{32}(1) \leftarrow w_{32}(1) - \eta \delta_3(1) s_3(1) [1 - s_3(1)] s_2(0)$$

# 动量项

- Steepest descent

$$w_{jk}^{m+1}(I) = w_{jk}^m(I) + \eta \Delta w_{jk}^{m+1}(I)$$

- With momentum,  $\alpha$

$$w_{jk}^{m+1}(I) = w_{jk}^m(I) + \eta \Delta w_{jk}^{m+1}(I) + \alpha \Delta w_{jk}^m(I)$$

- New step effected by previous step
- $m$  is the iteration number
- Convergence is improved

# 批处理BP算法

- 在每次更新权值之前，计算所有样本的误差
  - 真正的最速梯度下降方法
  - 每代更新一次
- 每计算一个样本点的误差就更新权值一次
  - 需将数据每个循环随机打乱

# 学习 vs. 记忆

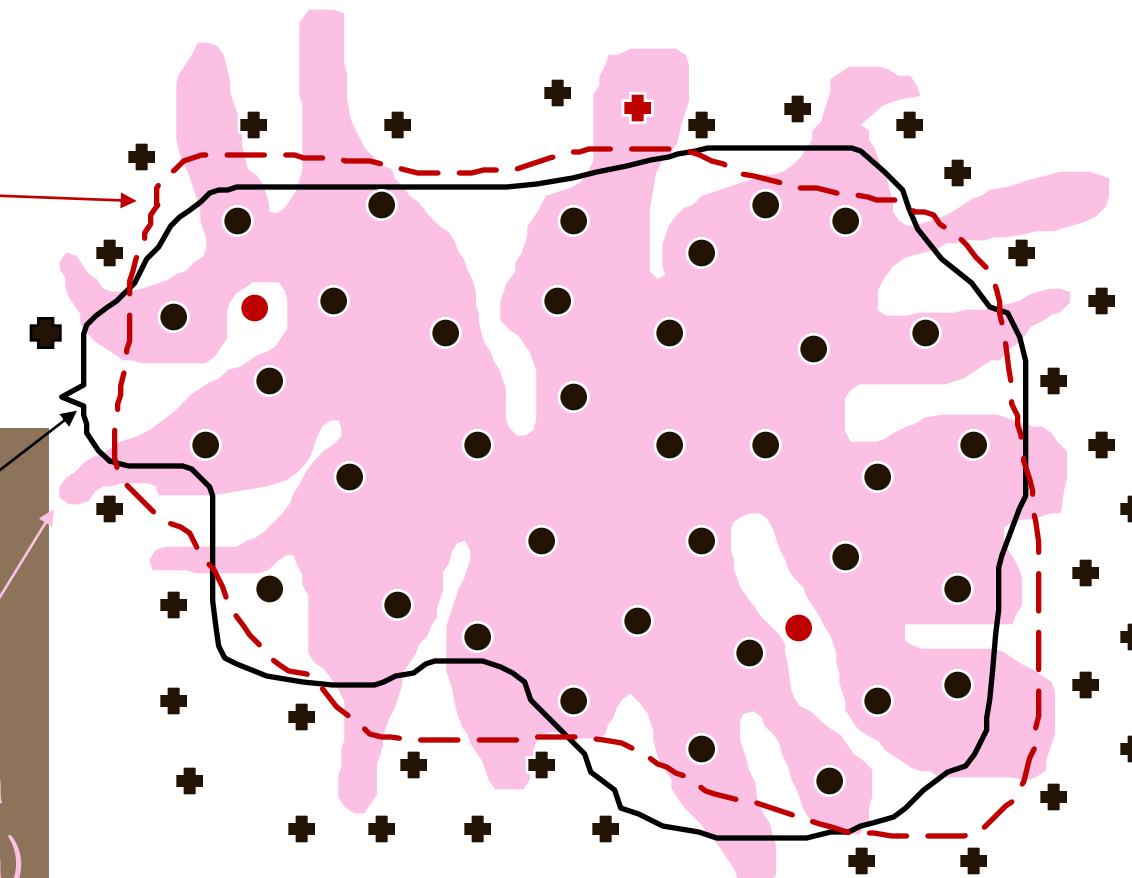
- + = training data
- + = test data

训练误差都为0

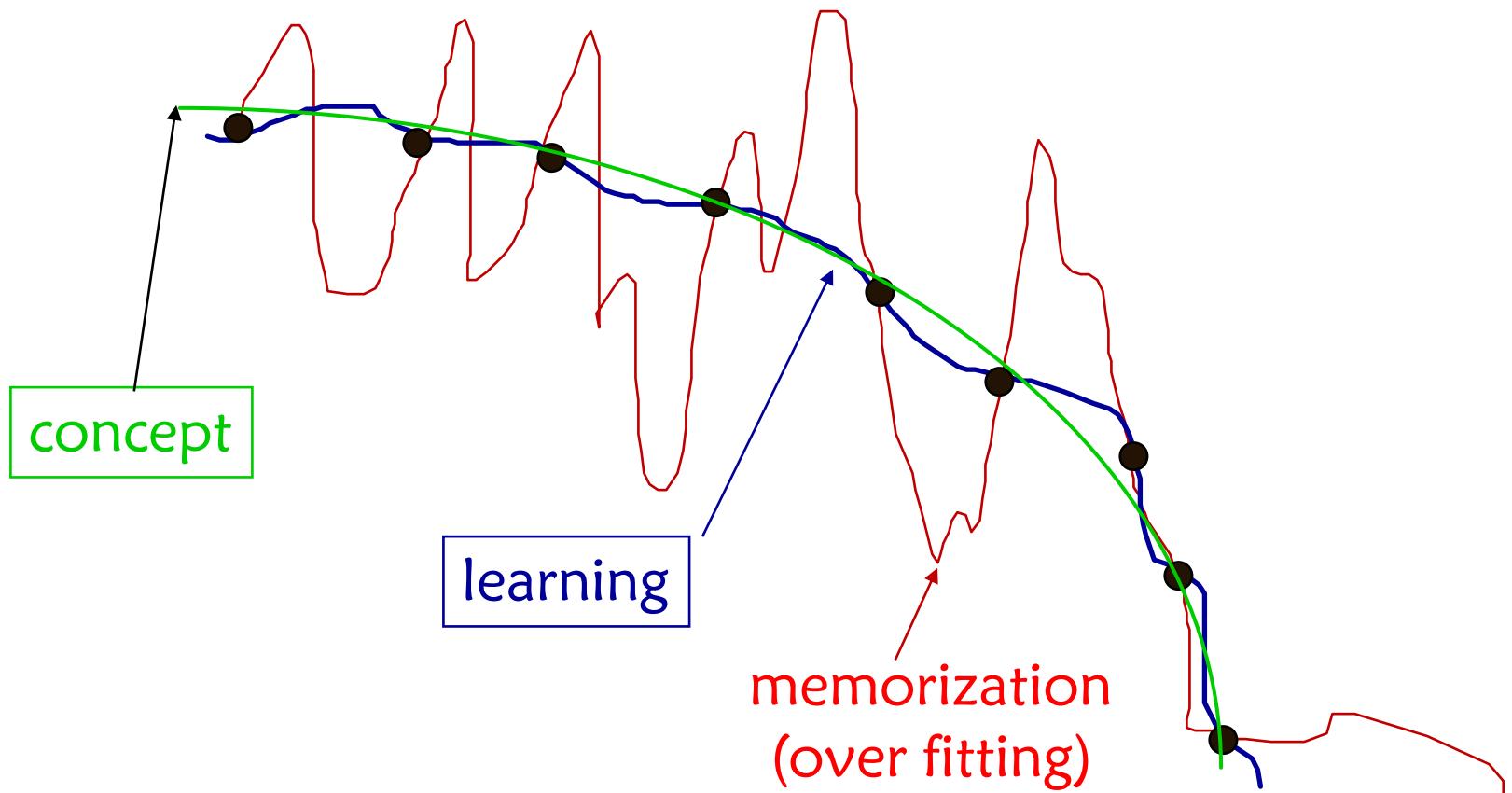
good  
generalization  
(learning)

concept  
(truth)

bad  
generalization  
(memorization)



# 学习 vs. 记忆 (续)



# 学习 vs. 记忆（续）

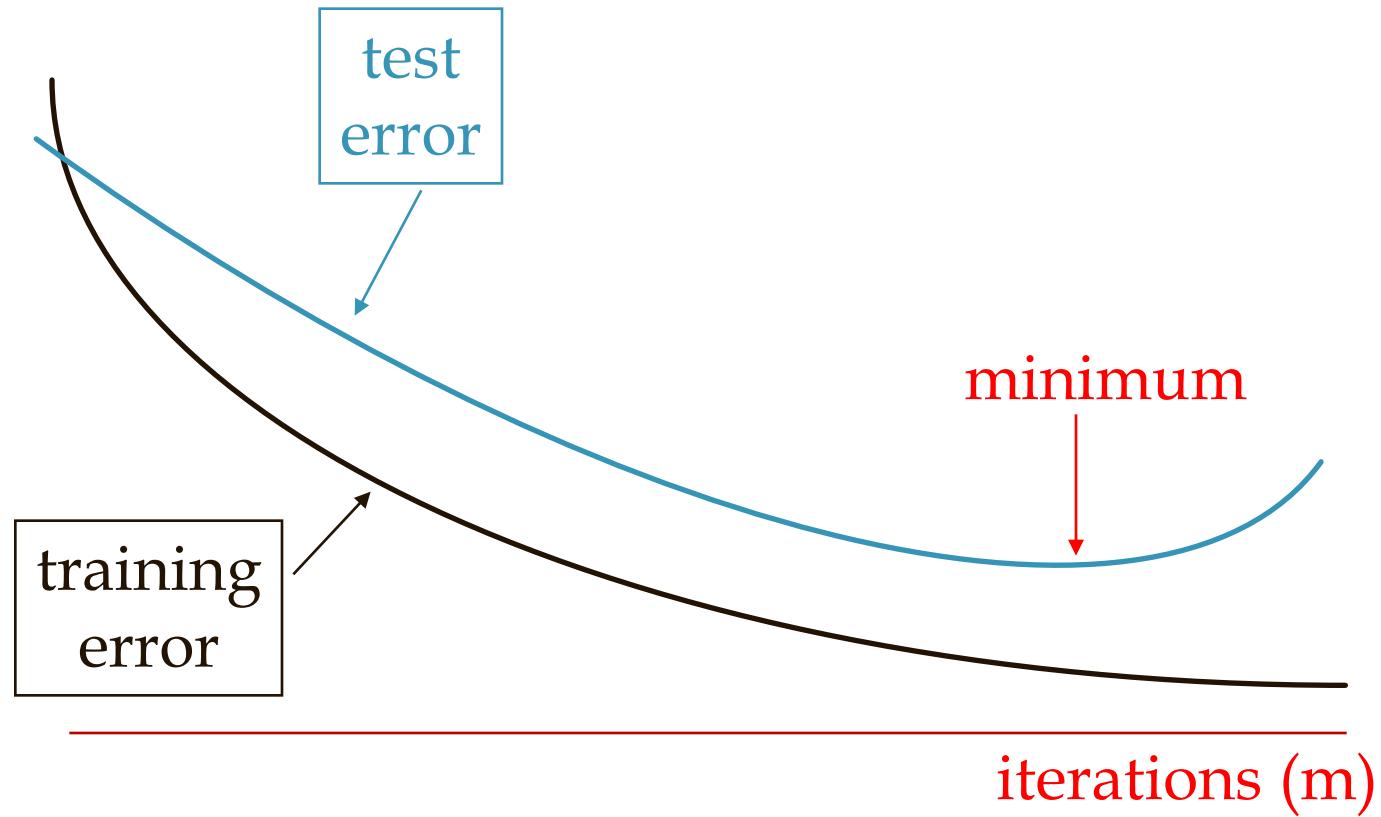
## ■ 好的学习：

- 能够识别出训练集之外的数据，如测试集
- 训练好的神经网络必须能成功地对未见到过的数据进行分类。

## ■ 如何保证我们的训练结果？

- 交叉验证
- 适当地选择网络结构
  - ✓ 尽可能地缩减网络规模——剪枝
  - ✓ 遗传算法训练网络结构

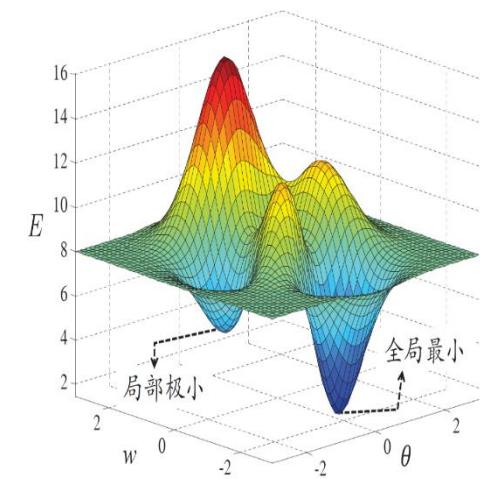
# 交叉验证



- 可能存在多个局部极小值, 但却只会有一个全局极最小值

## “跳出”局部最小的策略

- 多组不同的初始参数优化神经网络, 选取误差最小的解作为最终参数.
- 模拟退火技术 [Aarts and Korst, 1989]. 每一步都以一定的概率接受比当前解更差的结果, 从而有助于跳出局部极小.
- 随机梯度下降. 与标准梯度下降法精确计算梯度不同, 随机梯度下降法在计算梯度时加入了随机因素.
- 遗传算法 [Goldberg, 1989]. 遗传算法也常用来训练神经网络以更好地逼近全局极小.



## □ 多层前馈网络表示能力

- 只需要一个包含足够多神经元的隐层，多层前馈神经网络就能以任意精度逼近任意复杂度的连续函数

[Hornik et al., 1989]

## □ 多层前馈网络局限

- 神经网络由于强大的表示能力，经常遭遇过拟合。表现为：训练误差持续降低，但测试误差却可能上升
- 如何设置隐层神经元的个数仍然是个未决问题。实际应用中通常使用“试错法”调整

## □ 缓解过拟合的策略

- 早停：在训练过程中，若训练误差降低，但验证误差升高，则停止训练
- 正则化：在误差目标函数中增加一项描述网络复杂程度的部分，例如连接权值与阈值的平方和

# 作业

## Iris Data Set

[Download](#) [Data Folder](#) [Data Set Description](#)

Abstract: Famous database; from Fisher, 1936



Data Set Characteristics:	Multivariate	Number of Instances:	150	Area:	Life
Attribute Characteristics:	Real	Number of Attributes:	4	Date Donated	1988-07-01
Associated Tasks:	Classification	Missing Values?	No	Number of Web Hits:	720483

### Source:

Creator:

R.A. Fisher

Donor:

Michael Marshall ([MARSHALL%PLU@io.arc.nasa.gov](mailto:MARSHALL%PLU@io.arc.nasa.gov))

- 150 个样本
- 属于鸢尾属下的三个亚属：山鸢尾、变色鸢尾和维吉尼亚鸢尾
- 四个特征：花萼和花瓣的长度和宽度

# 作业

- 下载iris数据集<https://archive.ics.uci.edu/dataset/53/iris>
- 用Matlab或Python完成如下工作：
  - ① 对于三层和四层的网络结构，不同的隐层节点数目，对模型的精度进行比较，用图或表的形式给出对比结果。
  - ② 在上一步确定的网络结构基础上，选择不同的激活函数进行比较，用图或表的形式给出对比结果。

## 补充材料

- 人工神经网络的发展和重要模型
- 人工神经网络重要的会议和期刊

# 人工神经网络的发展和重要模型

- 初始（萌发）期——人工神经网络的兴起
  - 1943年，美国神经生理学家Warren Mcculloch 和数学家Walter Pitts合写了一篇关于神经元如何工作的开拓性文章：“A Logical Calculus of Ideas Immanent in Nervous Acitvity”。该文指出，脑细胞的活动像断/通开关，这些细胞可以按各种方式相互结合，进行各种逻辑运算。
  - 1949年，心理学家Donala Hebb写了一本书：“The Organization of Behavior”。在该书中，他强调了心理学和生理学间的联系和沟通，指出脑细胞间的思路每当通过参与某种活动时将被加强，这就是后来的Hebb学习规则。

- 到了二十世纪50年代，随着计算机的发展和软硬件的进步，有些神经系统功能的理论开始在计算机上进行模拟，拓宽了研究的路子。
- IBM的研究室在Hebb工作的基础上，对神经网络的模型进行了软件模拟，虽然开始时失败了，但在使得模型像人那样适应环境的实验上取得了一定程度的成功。
- 1956年，一个人工智能研究项目（Dartmouth Summer）给人工智能领域，同时也给神经计算领域以巨大推动。

- 人们提出两条研究思路
  - 采用高级人工智能方法，试图建立描述智能机功能的计算机程序；
  - 根据低水平的大脑处理方式构成结构模型，以实现智能化。
  - 这宣告了人工神经网络的诞生。

- 第一次高潮期 — 感知器模型和人工神经网络
  - 1957年，计算机专家Frank Rosenblatt开始从事感知器的研究，并制成硬件，通常被认为是最早的神经网络模型。
  - 1959年，两位电机工程师Bernard Widrow和Marcian Haff开发出一种叫作自适应线性单元（ADALINE）的网络模型，并在他们的论文“Adaptive Switching Circuits”中描述了该模型和它的学习算法（Widrow- Haff算法）。该网络通过训练，可以成功用于抵消通信中的回波和噪声，也可用于天气预报，成为第一个用于实际问题的神经网络。

- 1962年，Rosenblatt出版了一本书“*The Principles of Neurodynamics*”，详述了他的感知器模型。该感知器具有输入层、输出层和中间层，通过实验可以模仿人的某些特性，并断言它可以学会任何它可以表示的功能。
- 1967年，Stephen Grossberg通过对生理学的研究，开发了一种称作雪崩网的神经网络模型，可以控制机器人手臂的运动。
- 在这一时期，由于感知器的某些进展和对神经网络的宣传，人们乐观地认为几乎已经找到了实现智能的关键。人们夸大了神经网络的潜力（有人甚至担心制造机器人的人类会很快受到机器人的攻击）。

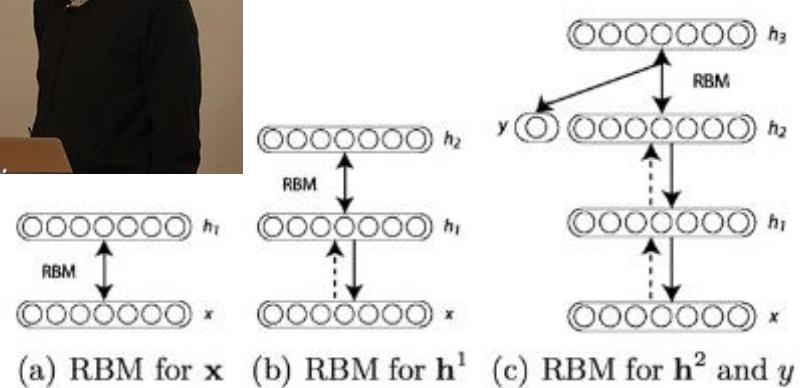
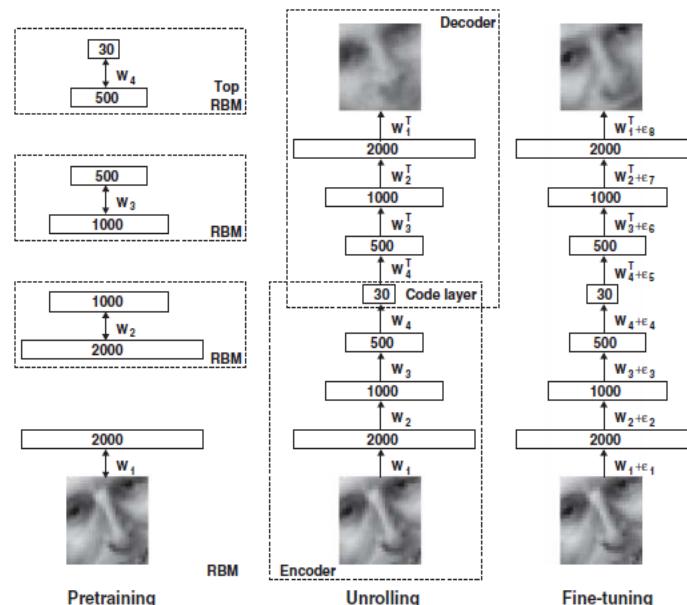
- 反思期 —— 神经网络的低潮
  - 1969年， Marvin Minsky和Seymour Papert合著了一本书 “Perception”，分析了当时的简单感知器，指出它有非常严重的局限性，甚至不能解决简单的“异或”问题，为 Rosenblatt的感知器判了“死刑”。
  - 此时，批评的声音高涨，导致了停止对人工神经网络研究所需的大量投资。
  - 不少研究人员把注意力转向了人工智能，导致对人工神经网络的研究陷入低潮。

- 第二次高潮期 —— Hopfield网络模型的出现和人工神经网络的复苏
  - 1982年，John Hopfield向美国科学院递交了有关神经网络的报告，主要内容就是建议收集和重视以前对神经网络的工作，其中特别强调了每种模型的实用性。
  - Hopfield揭示了以往的网络是如何工作的，可以做些什么，并提出了他自己的模型，能从失真的或不完善的数据图像中获得完整的数据图像，引起了美国军方的兴趣。
  - 当时，人工智能对自动制导车的研究失败，而利用神经网络有可能解决这个问题，从而使人们的注意力重新投向人工神经网络，导致了人工神经网络的第二次高潮。

- 1984年， Hopfield设计研制了后来被人们称为 Hopfield网的电路，较好地解决了TSP问题，找到了最佳解的近似解，引起了较大轰动。
- 1985年， Hinton、 Sejnowsky、 Rumelhart等研究者在Hopfield网络中引入随机机制，提出了所谓的Bolzman机。
- 1986年， Rumelhart等研究者重新独立地提出多层网络的学习算法—BP算法，较好地解决了多层网络的学习问题。
- 1990年12月，国内首届神经网络大会在北京举行。

- 再认识与应用研究期
  - 二十世纪90年代后，研究趋于平缓，主要问题：
    - ✓ 应用面还不够宽
    - ✓ 结果不够精确
    - ✓ 存在可信度的问题
  - ✓ 统计学习理论和支持向量机兴起

- 2006年，首次提出深度学习  
加拿大多伦多大学Geoffrey Hinton在《科学》上发表  
了一篇文章：提出了深度信念网络(DBN)，通过“预训练  
+微调”使得深度模型的最优化变得相对容易
- 开启了深度学习在学术界和工业界的浪潮。



Reducing the Dimensionality of Data with Neural Networks G. E. Hinton\* and R. R. Salakhutdinov , 28 JULY 2006 VOL 313 SCIENCE www.sciencemag.org

# 2010年，在‘深度学习’的大旗下，神经网络再度崛起---第三次高潮

- 2010年，美国国防部DARPA计划首次资助深度学习项目
- Deep learning 在ImageNet上，语音识别上的巨大成功
  - 2011年以来，微软研究院和Google的语音识别研究人员先后采用DNN技术降低语音识别错误率20%~30%，是语音识别领域十多年来最大的突破性进展。
  - 2012年，DNN技术在图像识别领域取得惊人的效果，在ImageNet评测上将错误率从26%降低到15%。
- AlphaGo
- 自然语言理解 Google, Baidu, Facebook
- ....

- 深度学习模型在计算机视觉 (image&video)、自然语言处理 (Text&language)、语音识别 (Speech&Audio) 等众多领域都取得了较大的成功。

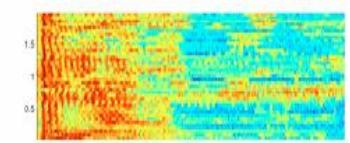
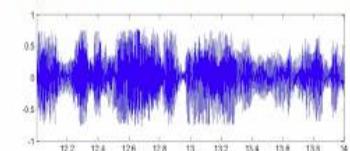
Images & Video



Text & Language



Speech & Audio



# Hall of Fames in Deep Learning



Geoffery Hinton  
(RBM, DNN, capsule)



Michael Jordan  
(RNN)



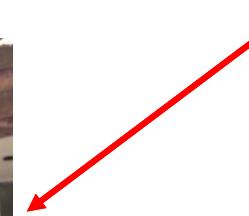
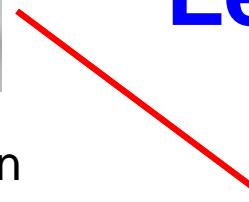
Yann LeCun  
(CNN)



Ian Goodfellow  
(GAN)



Yoshua Bengio  
(auto-encoder, GAN)



- LeNet, 1998
- AlexNet, 2012
- ZF-net, 2013
- GoogleNet, 2014
- VGG, 2014
- ResNet, 2015

## ResNets @ ILSVRC & COCO 2015 Competitions

- **1st places in all five main tracks**
  - ImageNet Classification: “Ultra-deep” **152-layer** nets
  - ImageNet Detection: **16%** better than 2nd
  - ImageNet Localization: **27%** better than 2nd
  - COCO Detection: **11%** better than 2nd
  - COCO Segmentation: **12%** better than 2nd

# Revolution of Depth

152 layers

3.57

ILSVRC'15  
ResNet

22 layers

6.7

ILSVRC'14  
GoogleNet

19 layers

7.3

ILSVRC'14  
VGG

11.7

8 layers

ILSVRC'13

16.4

8 layers

ILSVRC'12  
AlexNet

25.8

shallow

ILSVRC'11

28.2

ILSVRC'10

AlexNet → ZFNet → VGGNet → Google → ResNet

- **图像分类**: MNIST, CIFAR, ImageNet
- **物体定位**: ImageNet
- **物体识别**: PASCAL, COCO
- **语义分割**: PASCAL, COCO

# ImageNet Classification



mite

container ship

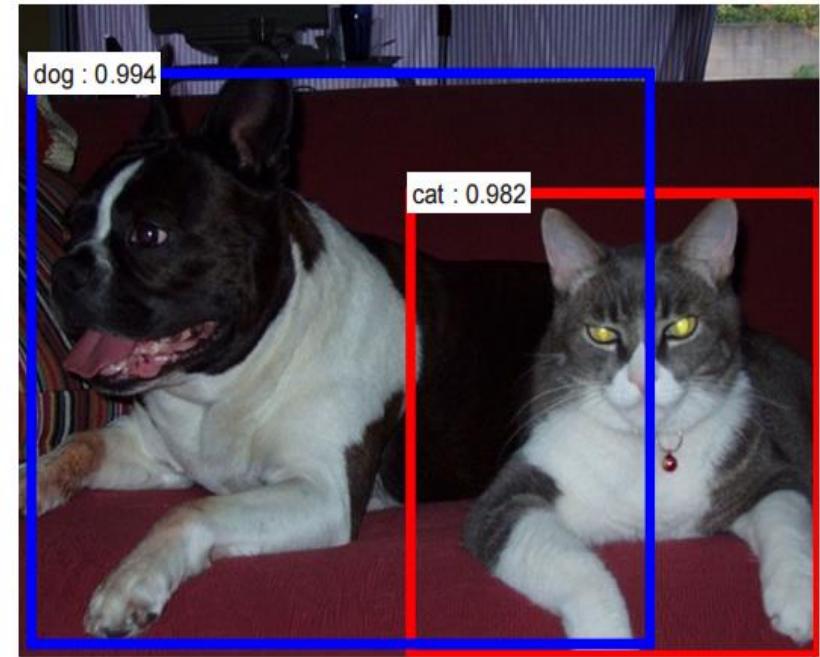
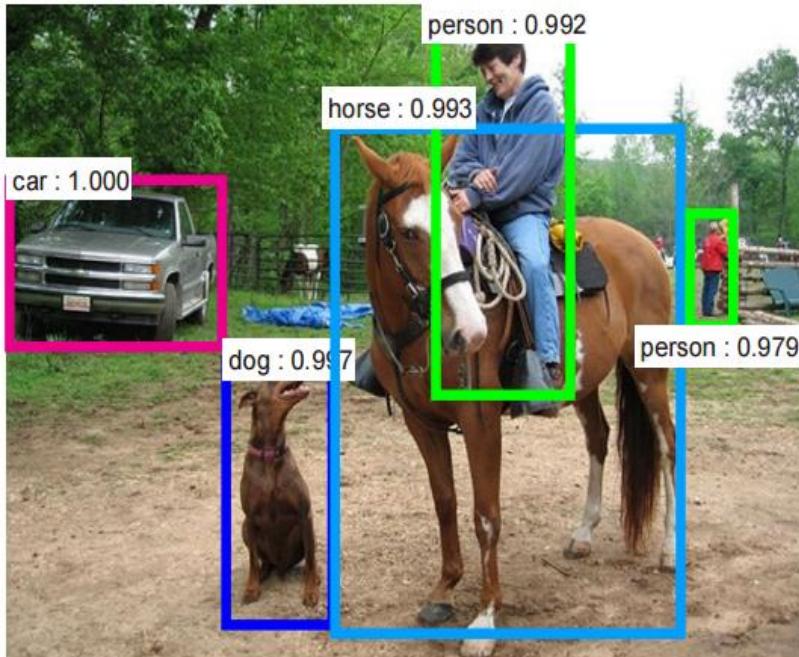
motor scooter

leopard

mite	container ship	motor scooter	leopard
black widow	lifeboat	go-kart	jaguar
cockroach	amphibian	moped	cheetah
tick	fireboat	bumper car	snow leopard
starfish	drilling platform	golfcart	Egyptian cat

<http://www.image-net.org/>

# Object Detection



# Image Captioning



man in black shirt is playing guitar.



construction worker in orange safety vest is working on road.



two young girls are playing with lego toy.



boy is doing backflip on wakeboard.

## 人工神经网络重要的会议和期刊

## □ 主流学术期刊

- Neural Computation
- Neural Networks
- IEEE Transactions on Neural Networks and Learning Systems

## □ 国际会议

- 国际神经信息处理系统会议 (NeurIPS)
- 国际神经网络联合会议 (IJCNN)

## □ 区域性国际会议

- 欧洲神经网络会议 (ICANN)
- 亚太神经网络会议 (ICONIP)