机器学习

计算机学院

杨晓春

xcyang@bit.edu.cn

1

第五章：神经网络

- Tom M. Mitchell, McGraw Hill, 2003
- http://www.cs.cmu.edu/~awm/tutorials
- 周志华，机器学习，2016

目标

- 了解神经网络NN的工作原理与适用范围
- 理解线性规划与NN的关系

3

提纲

- 神经网络发展史
  - 发展史
  - 受生物神经元网络的启发
- 神经元网络
  - 神经元模型
  - 感知机与多层网络
  - 误差逆传播算法
  - 全局最小与局部最小
- 其他神经元网络

4

神经网络发展史

第一阶段

- 1943年, McCulloch和Pitts 提出第一个神经元数学模型, 即M-P模型, 并从原理上证明了人工神经网络能够计算任何算数和逻辑函数
- 1949年, Hebb 发表《The Organization of Behavior》一书, 提出生物神经元学习的机理, 即Hebb学习规则
- 1958年, Rosenblatt 提出感知机网络（Perceptron）模型和其学习规则
- 1960年, Widrow和Hoff提出自适应线性神经元（Adaline）模型和最小均方学习算法
- 1969年, Minsky和Papert 发表《Perceptrons》一书, 指出单层神经网路不能解决非线性问题, 多层网络的训练算法尚无希望. 这个论断导致神经网络进入低谷

神经网络发展史

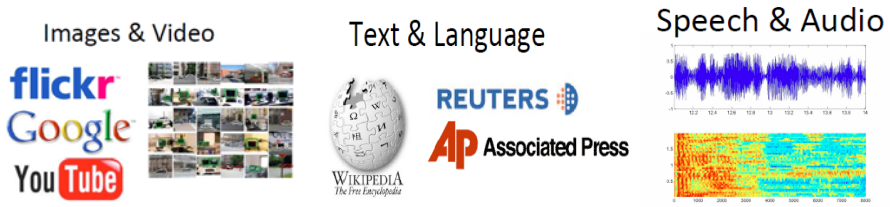
第二阶段

- 1982年, 物理学家Hopfield提出了一种具有联想记忆、优化计算能力的递归网络模型, 即Hopfield 网络
- 1986年, Rumelhart 等编辑的著作《Parallel Distributed Proceesing:Explorations in the Microstructures of Cognition》报告了反向传播算法
- 1987年, IEEE 在美国加州圣地亚哥召开第一届神经网络国际会议（ICNN）
- 90年代初, 伴随统计学习理论和SVM的兴起, 神经网络由于理论不够清楚, 试错性强, 难以训练, 再次进入低谷

神经网络发展史

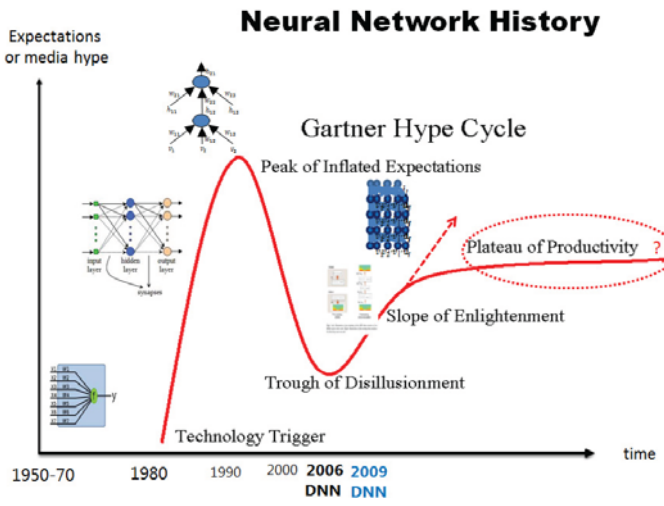
第三阶段

- 2006年, Hinton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
- 2012年, Hinton 组参加ImageNet 竞赛, 使用 CNN 模型以超过第二名10个百分点的成绩夺得当年竞赛的冠军
- 伴随云计算、大数据时代的到来，计算能力的大幅提升，使得深度学习模型在计算机视觉、自然语言处理、语音识别等众多领域都取得了较大的成功



本页课件来源：周志华《机器学习》及其课件，致谢： 高斌斌

神经网络发展史

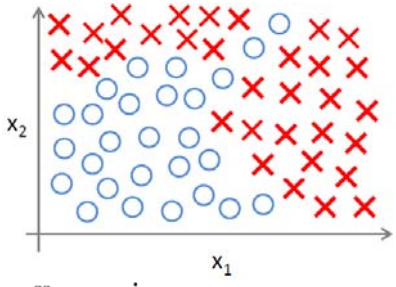


本页课件来源：周志华《机器学习》及其课件，致谢： 高斌斌

最新一届图灵奖得主

最新一届图灵奖被授予了“深度学习三巨头”：约书亚·本希奥（Yoshua Bengio）、杰弗里·欣顿（Geoffrey Hinton）和杨立昆（Yann LeCun）| 图片来源：Botler AI

非线性分类问题

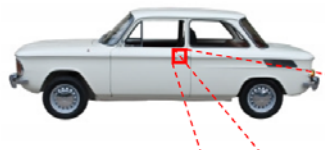


$$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 x_2 + \theta_5 x_1^3 x_2 + \theta_6 x_1 x_2^2 + \dots)$$

x_1 = size
x_2 = # bedrooms
x_3 = # floors
x_4 = age
$\dots$
x_{100}

What is this?

You see this:



But the camera sees this:

194	210	201	212	199	213	215	195	178	158	182	209
180	189	190	221	209	205	191	167	147	115	129	163
114	126	140	188	176	165	152	140	170	106	78	88
87	103	115	154	143	142	149	153	173	101	57	57
102	112	106	131	122	138	152	147	128	84	58	66
94	95	79	104	105	124	129	113	107	87	69	67
68	71	69	98	89	92	98	95	89	88	76	67
41	56	68	99	63	45	60	82	58	76	75	65
20	43	69	75	56	41	51	73	55	70	63	44
50	50	57	69	75	75	73	74	53	68	59	37
72	59	53	66	84	92	84	74	57	72	63	42
67	61	58	65	75	78	76	73	59	75	69	50

11

Computer Vision: Car detection

Cars

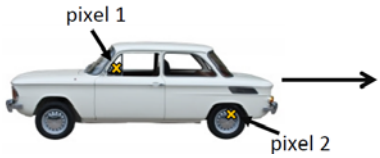
Not a car

Testing:

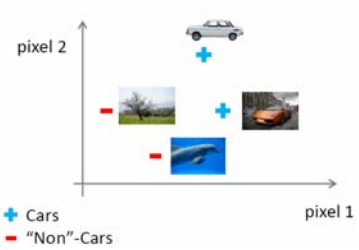
What is this?

12

Non-linear Classification



Learning Algorithm



50 x 50 pixel images→ 2500 pixels

$n = 2500$ (7500 if RGB)

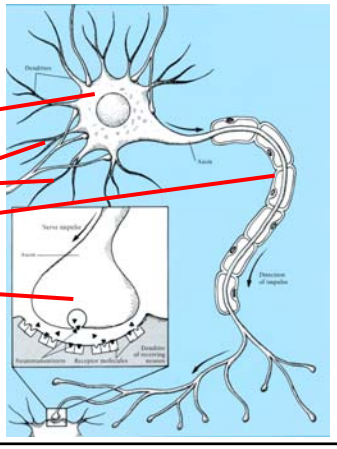
$$x = \begin{bmatrix} \text{pixel 1 intensity} \\ \text{pixel 2 intensity} \\ \vdots \\ \text{pixel 2500 intensity} \end{bmatrix}$$

Quadratic features ($x_i \times x_j$): ≈3 million features

13

神经元网络

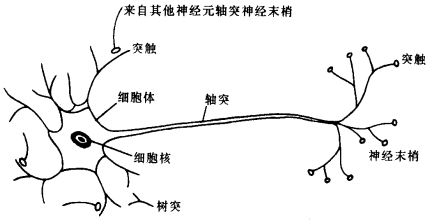
- 最初的想法：算法试图模拟人类的大脑
  - 50年代：Perceptron(感知器) —单层神经元网络
  - 80年代：Backpropagation(反向传播)—多层神经元网络
  - 和90年代初：广泛使用、90年代末：popularity diminished
  - 近10年：蓬勃发展
- 神经元结构
  - 神经元
  - 树突
  - 轴突
  - 神经末梢



14

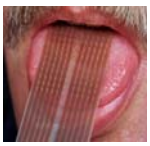
生物神经网络

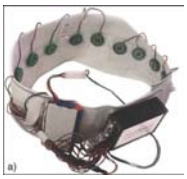
- 生物神经网络
  - 每个神经元与其他神经元相连, 当它“兴奋”时, 就会向相连的神经云发送化学物质, 从而改变这些神经元内的电位；如果某神经元的电位超过一个“阈值”, 那么它就会被激活, 即“兴奋”起来, 向其它神经元发送化学物质
- 神经末梢根据经验改变粗细和信号强度
- 连在一起的神经元被同时触发
- 赫比学习法(Hebbian learning): 当两个连接的神经元被同时触发，它们之间的神经末梢的强度增强



15

感知头脑内部的传感器



16

神经元模型

神经网络的定义

“神经网络是由具有适应性的简单单元组成的广泛并行互联的网络, 它的组织能够模拟生物神经系统对真实世界物体所作出的反应”[Kohonen, 1988]

- 机器学习中的神经网络通常是指“神经网络学习” 或者机器学习与神经网络两个学科的交叉部分
- 神经元模型即上述定义中的“简单单元”是神经网络的基本成分

生物神经网络与人工神经网络的关系

- 人类
  - 神经元switching time ~.001秒
  - 神经元个数~10^{11}
  - 每个神经元有 ~10^{4-5}个连接
  - 场景识别时间 ~.1秒
  - 100推导步骤
- 人工神经元网络（ANN: Artificial Neural Network）的特点
  - Switching time: 计算机硬件nanoseconds(十亿分之一秒)
  - 很多类似神经元的阈值交换单元(switching unit)
  - 单元间有很多加权的内部连接
  - 高并行计算、分布式处理
  - 自动调节权重

何时考虑使用ANN

- 高维输入：离散或实数 (例如：原始感知的传感器输入)
- 输出：离散或实数
- 输出是一组值的向量
- 可能保护噪声数
- 目标函数的形式不清楚
- 结果的可解释性不重要
- 例子
  - 语音音素识别
  - 图像分类
  - 财务预测

19

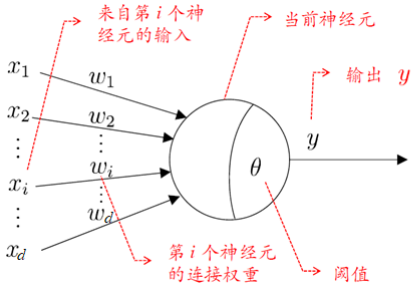
提纲

- 神经网络发展史
  - 发展史
  - 受生物神经元网络的启发
- 神经元网络
  - 神经元模型
  - 感知机与多层网络
  - 误差逆传播算法
  - 全局最小与局部最小
- 其他神经元网络

20

神经元模型

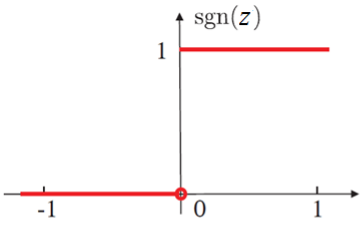
- M-P 神经元模型 [McCulloch and Pitts, 1943]
  - 输入：来自其他 n 个神经元传递过来的输入信号
  - 处理：输入信号通过带权重的连接进行传递, 神经元接受到总输入值将与神经元的阈值进行比较
  - 输出：通过激活函数的处理以得到输出



$$X = \begin{bmatrix} X_1 \\ X_2 \\ \cdots \\ X_d \end{bmatrix} \qquad \omega = \begin{bmatrix} \omega_1 \\ \omega_2 \\ \cdots \\ \omega_d \end{bmatrix}$$

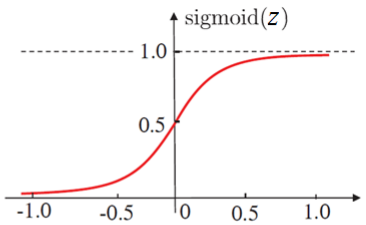
神经元模型

激活函数 $g(\cdot)$



$$\mathrm{sgn}(z) = \begin{cases} 1, & \text{if } z \geq 0; \\ 0, & \text{if } z < 0. \end{cases}$$

(a) 阶跃函数



$$\mathrm{sigmoid}(z) = \frac{1}{1+e^{-z}}$$

(b) Sigmoid 函数

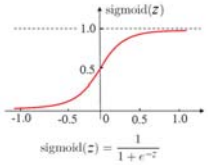
- 理想激活函数是阶跃函数, 0表示抑制神经元而1表示激活神经元
- 阶跃函数具有不连续、不光滑等不好的性质, 常用的是 Sigmoid 函数

提纲

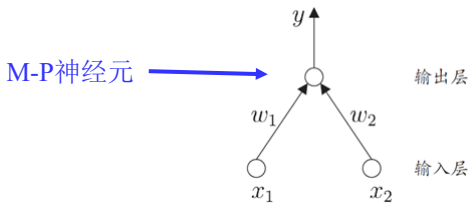
- 神经网络发展史
  - 发展史
  - 受生物神经元网络的启发
- 神经元网络
  - 神经元模型
  - 感知机与网络模型
  - 误差逆传播算法
  - 全局最小与局部最小
- 其他神经元网络

23

感知机

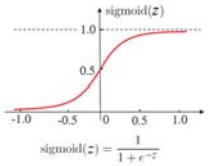


- 感知机
  - 感知机由两层神经元组成, 输入层接受外界输入信号传递给输出层, 输出层是M-P神经元（阈值逻辑单元）
  - 只有输出层进行激活函数处理，即只有一层功能神经元



24

感知机



感知机

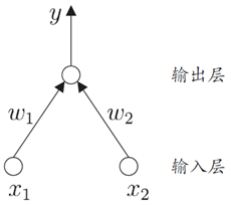
- 感知机由两层神经元组成, 输入层接受外界输入信号传递给输出层, 输出层是M-P神经元（阈值逻辑单元）
- 感知机能够容易地实现逻辑与、或、非运算

“与”操作：$x_1 \wedge x_2$

令 $w_1 = w_2 = 1, \theta = 2$

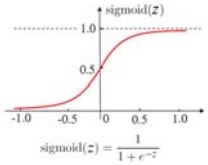
则 $y = g(1 \cdot x_1 + 1 \cdot x_2 - 2)$

仅在 $x_1 = x_2 = 1$时, $y = 1$



25

感知机



感知机

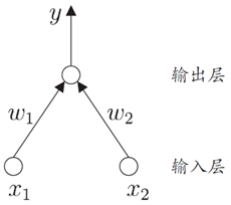
- 感知机由两层神经元组成, 输入层接受外界输入信号传递给输出层, 输出层是M-P神经元（阈值逻辑单元）
- 感知机能够容易地实现逻辑与、或、非运算

“或”操作：$x_1 \vee x_2$

令 $w_1 = w_2 = 1, \theta = 0.5$

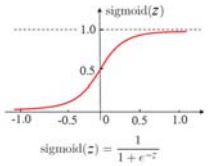
则 $y = g(1 \cdot x_1 + 1 \cdot x_2 - 0.5)$

仅在 $x_1 = 1$ 或 $x_2 = 1$时, $y = 1$



26

感知机



- 感知机
  - 感知机由两层神经元组成, 输入层接受外界输入信号传递给输出层, 输出层是M-P神经元（阈值逻辑单元）
  - 感知机能够容易地实现逻辑与、或、非运算

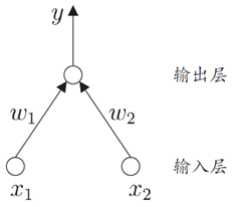
“非”操作：$\neg x_1$

令 $w_1 = -0.6, w_2 = 0, \theta = -0.5$

则 $y = g(-0.6 \cdot x_1 + 0 \cdot x_2 - 0.5)$

当 x_1=1时，$y = 0$；

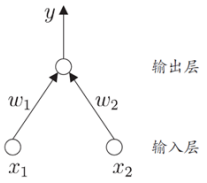
当 x_1=0时，$y = 1$。

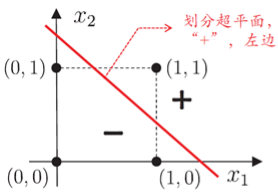


27

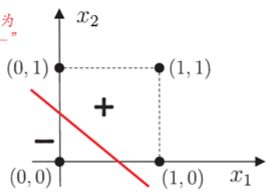
感知机

- 感知机求解与、或、非问题
  - 线性可分的与、或、非问题
  - 保证可以收敛 (converge)

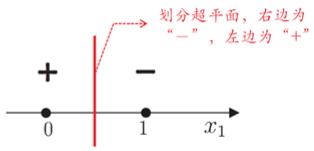




(a) “与”问题 $(x_1 \wedge x_2)$



(b) “或”问题 $(x_1 \vee x_2)$



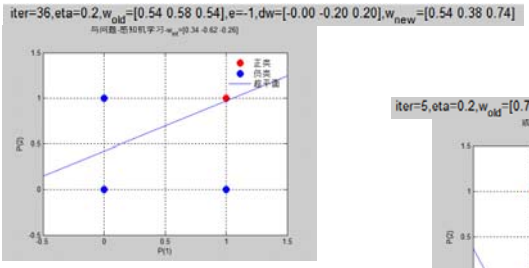
(c) “非”问题 $(\neg x_1)$

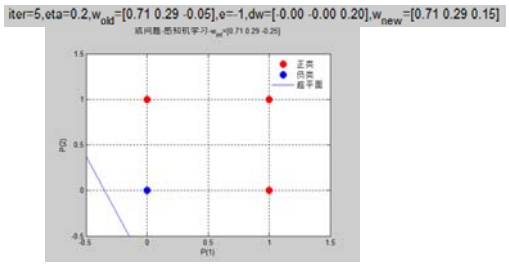
本页课件来源：周志华《机器学习》及其课件，致谢： 高斌斌

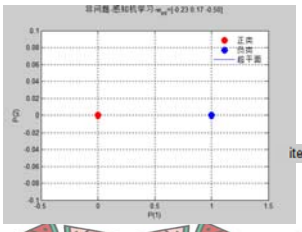
28

感知机

- 感知机求解与、或、非问题

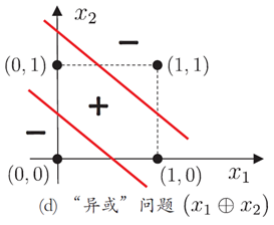




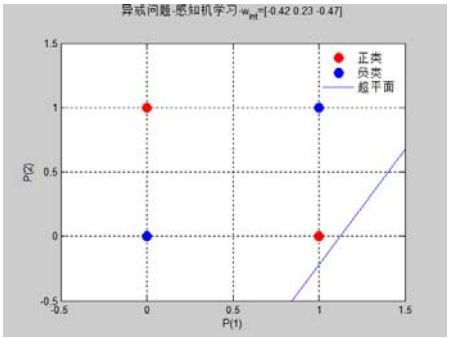


线性不可分的问题

- 感知机求解异或问题
  - *w*发生震荡，无法收敛



(d) “异或”问题 $(x_1 \oplus x_2)$

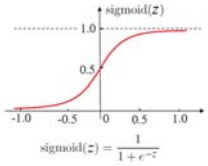


感知机与多层网络

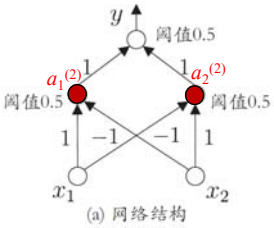
- 感知机学习能力
  - 若两类模式线性可分, 则感知机的学习过程一定会收敛; 否感知机的学习过程将会发生震荡[Minsky and Papert, 1969]
  - 单层感知机的学习能力非常有限, 只能解决线性可分问题
  - 事实上, 与、或、非问题是线性可分的, 因此感知机学习过程能够求得适当的权值向量. 而异或问题不是线性可分的, 感知机学习不能求得合适解
  - 对于非线性可分问题, 如何求解?

    多层感知机（多个线性模型的叠加）

感知机与多层网络



- 解决异或问题的两层感知机

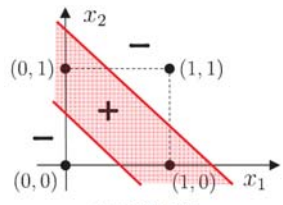


(a) 网络结构

$$a_1^{(2)} = g(1 \cdot x_1 - 1 \cdot x_2 - 0.5)$$
$$a_2^{(2)} = g(-1 \cdot x_1 + 1 \cdot x_2 - 0.5)$$
$$y = g(1 \cdot x_1 + 1 \cdot x_2 - 0.5)$$

x_1, x_2	$a_1^{(2)}, a_2^{(2)}$	y
0, 0	0, 0	0
0, 1	0, 1	1
1, 0	1, 0	1
1, 1	0, 0	0

通过每层的激活函数，
将点映射到新的一层所代表的特征空间里

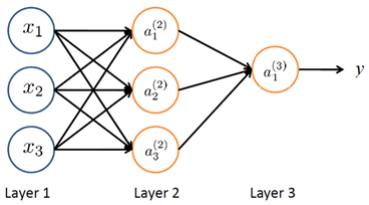


(b) 分类区域

- 输出层与输入层之间的一层神经元, 被称之为隐层或隐含层, 隐含层和输出层神经元都是具有激活函数的功能神经元

感知机与多层网络

- 通过控制矩阵w，完成从第j层映射到第$j+1$层的函数映射



$a_i^{(j)}$：在第j层激活单元i

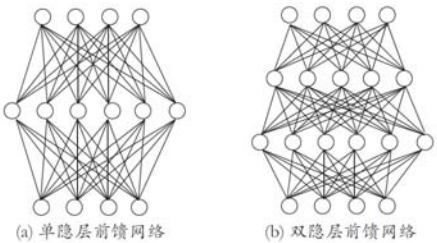
$$a_1^{(2)} = g(\omega_{10}^{(1)}x_0 + \omega_{11}^{(1)}x_1 + \omega_{12}^{(1)}x_2 + \omega_{13}^{(1)}x_3)$$
$$a_2^{(2)} = g(\omega_{20}^{(1)}x_0 + \omega_{21}^{(1)}x_1 + \omega_{22}^{(1)}x_2 + \omega_{23}^{(1)}x_3)$$
$$a_3^{(2)} = g(\omega_{30}^{(1)}x_0 + \omega_{31}^{(1)}x_1 + \omega_{32}^{(1)}x_2 + \omega_{33}^{(1)}x_3)$$
$$y = a_1^{(3)} = g(\omega_{10}^{(1)}a_0^{(2)} + \omega_{11}^{(2)}a_1^{(2)} + \omega_{12}^{(2)}a_1^{(2)} + \omega_{13}^{(2)}a_1^{(2)})$$
$$x_0 = -1 \leftarrow \text{哑节点(dummy node)}$$

- 如果网络第i层有s_i个单元，第$i+1$层有s_{i+1}个单元，则 $w^{(i)}$将有$s_{i+1}\times(s_i+1)$个维

33

感知机与多层网络

- 多层前馈神经网络
  - 定义：每层神经元与下一层神经元全互联, 神经元之间不存在同层连接也不存在跨层连接
  - 前馈：输入层接受外界输入, 隐含层与输出层神经元对信号进行加工, 最终结果由输出层神经元输出
  - 学习：根据训练数据来调整神经元之间的“连接权”以及每个功能神经元的“阈值”
  - 多层网络：包含隐层的网络

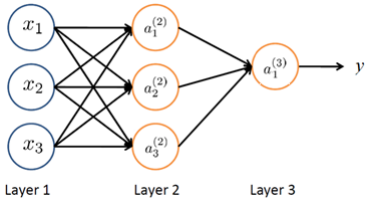


本页课件来源：周志华《机器学习》及其课件，致谢！

34

前馈传播 (Forward propagation)

向量实现



$$a_1^{(2)} = g(\omega_{10}^{(1)}x_0 + \omega_{11}^{(1)}x_1 + \omega_{12}^{(1)}x_2 + \omega_{13}^{(1)}x_3)$$
$$a_2^{(2)} = g(\omega_{20}^{(1)}x_0 + \omega_{21}^{(1)}x_1 + \omega_{22}^{(1)}x_2 + \omega_{23}^{(1)}x_3)$$
$$a_3^{(2)} = g(\omega_{30}^{(1)}x_0 + \omega_{31}^{(1)}x_1 + \omega_{32}^{(1)}x_2 + \omega_{33}^{(1)}x_3)$$
$$y = a_1^{(3)} = g(\omega_{10}^{(1)}a_0^{(2)} + \omega_{11}^{(2)}a_1^{(2)} + \omega_{12}^{(2)}a_1^{(2)} + \omega_{13}^{(2)}a_1^{(2)})$$
$$x_0 = -1$$

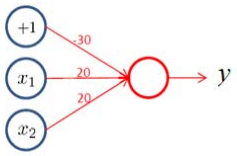
$$X = \begin{bmatrix} -1 \\ X_1 \\ X_2 \\ X_3 \end{bmatrix} \qquad Z^{(j)} = \begin{bmatrix} Z_1^{(j)} \\ Z_2^{(j)} \\ Z_3^{(j)} \end{bmatrix}$$

$$z^{(2)} = \omega^{(1)}X$$
$$a^{(2)} = g(z^{(2)})$$
And $\quad a_0^{(2)} = 1$
$$z^{(3)} = \omega^{(2)}a^{(2)}$$
$$y = a^{(3)} = g(z^{(3)})$$

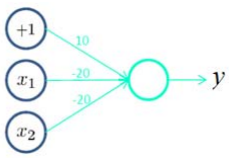
关于异或的另一个例子

$x_1 \mathrm{XOR} x_2$

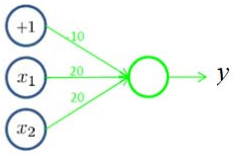
x_1	x_2	$a_1^{(2)}$	$a_2^{(2)}$	y
0	0	0	1	1
0	1	0	0	0
1	0	0	0	0
1	1	1	0	1



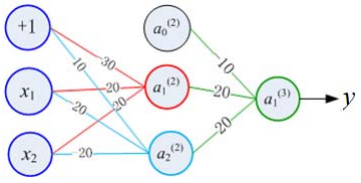
x_1 AND x_2



(NOT x_1) AND (NOT x_2)



x_1 OR x_2



$x_1 \mathrm{XOR} x_2$

同一个问题，可以训练出不同的网络

例子

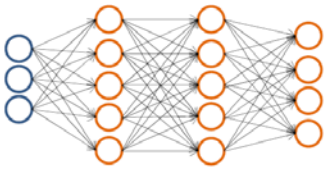
- 多个输出单元


Pedestrian


Car


Motorcycle


Truck



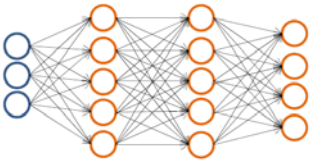
$h_\Theta(x) \in \mathbb{R}^4$

Want $y \approx \begin{bmatrix}1\\0\\0\\0\end{bmatrix}$, $y \approx \begin{bmatrix}0\\1\\0\\0\end{bmatrix}$, $y \approx \begin{bmatrix}0\\0\\1\\0\end{bmatrix}$, etc.

when pedestrian when car when motorcycle

37

例子



$h_\Theta(x) \in \mathbb{R}^4$

Want $y \approx \begin{bmatrix}1\\0\\0\\0\end{bmatrix}$, $y \approx \begin{bmatrix}0\\1\\0\\0\end{bmatrix}$, $y \approx \begin{bmatrix}0\\0\\1\\0\end{bmatrix}$, etc.

when pedestrian when car when motorcycle

Training set: $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})$

$y^{(i)}$ one of $\begin{bmatrix}1\\0\\0\\0\end{bmatrix}$, $\begin{bmatrix}0\\1\\0\\0\end{bmatrix}$, $\begin{bmatrix}0\\0\\1\\0\end{bmatrix}$, $\begin{bmatrix}0\\0\\0\\1\end{bmatrix}$

pedestrian car motorcycle truck

38

提纲

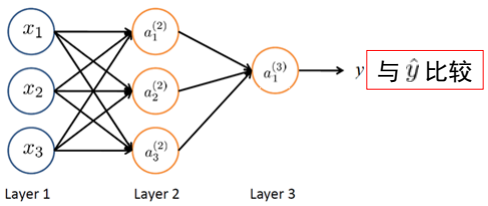
- 神经网络发展史
  - 发展史
  - 受生物神经元网络的启发
- 神经元网络
  - 神经元模型
  - 感知机与网络模型
  - 误差逆传播算法
  - 全局最小与局部最小
- 其他神经元网络

网络模型学习

- 给定训练数据集，则权重w_i (i=1,2,…,n)、阈值$\theta = w_0$可通过学习得到
- 感知机学习规则
  - 对训练样例($\boldsymbol{x}$,y)，若当前感知机的输出为$\hat{y}$，则感知机权重调整规则为：

$$w_i \leftarrow w_i + \triangle w_i$$
$$\triangle w_i = \eta(y - \hat{y})x_i$$

  - 其中$\eta \in (0, 1)$称为学习率



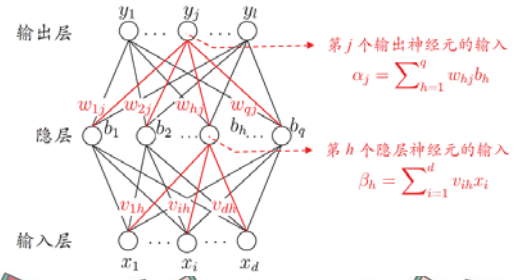
若感知机对训练样例($\boldsymbol{x}$,y)预测正确，则感知机不发生变化；
否则根据错误程度进行权重的调整.

误差逆传播算法（BP算法）

- 误差逆传播算法（Error Back Propagation, 简称BP）
  - 最成功的训练多层前馈神经网络的学习算法
- 训练集：输入样例由d个属性描述, 输出l 维实值向量

$$D=\{(\boldsymbol{x}_i,y_i)\},\boldsymbol{x}_i\in R^d,y_i\in R^l,(i=1,2,\cdots,m)$$

- 构成一个多层前向前馈网络结构
  - d个输入神经元, l个输出神经元, q个隐层神经元的



本页课件来源：周志华《机器学习》及其课件，致谢！

误差逆传播算法（BP算法）

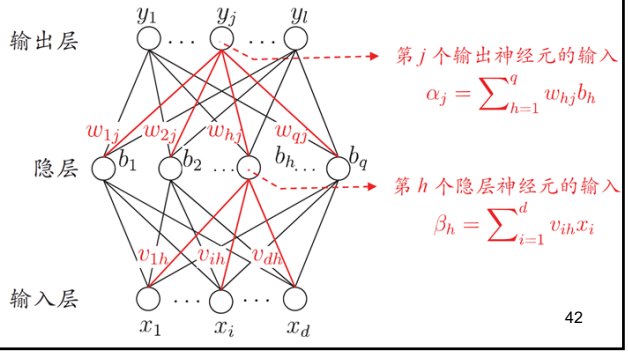
- 网络参数

θ_j：输出层第j个神经元阈值；

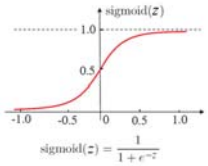
γ_h：隐含层第h个神经元阈值；

v_{ih}：输入层与隐层神经元之间的连接权重；

w_{hj}：隐层与输出层神经元之间的连接权重；



误差逆传播算法（BP算法）

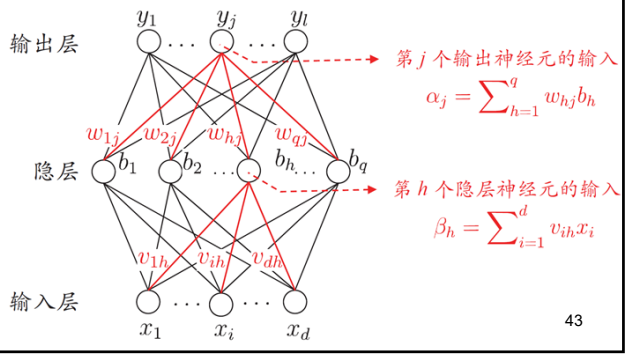


网络参数

对训练例 $(\boldsymbol{x}_k, \boldsymbol{y}_k)$, 假定神经网络的输出为 $\hat{\boldsymbol{y}}_k = (\hat{y}_1^k, \hat{y}_2^k, \ldots, \hat{y}_l^k)$, 即

$$\hat{y}_j^k = g\left(\beta_j - \theta_j\right)$$

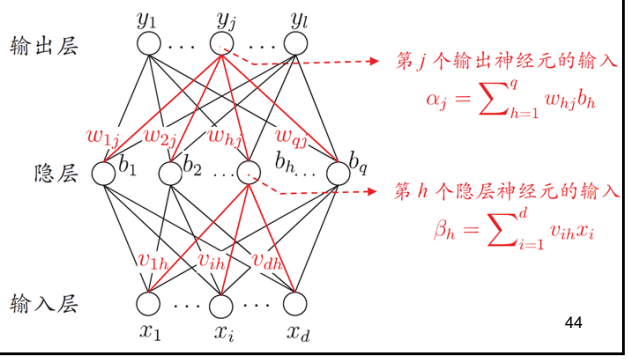
则网络在 $(\boldsymbol{x}_k, \boldsymbol{y}_k)$ 上的均方误差为 $E_k = \frac{1}{2}\sum_{j=1}^{l}(\hat{y}_j^k - y_j^k)^2$



误差逆传播算法（BP算法）

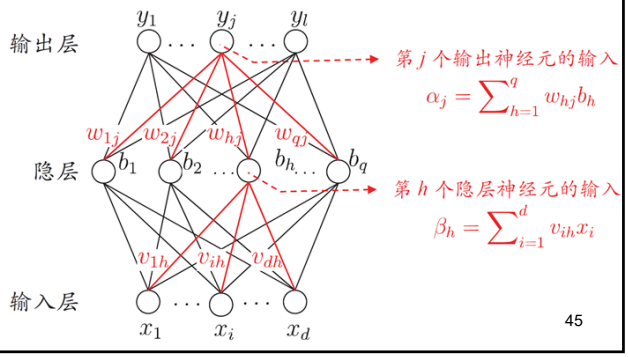
参数个数

- 权重：v_{ih}, w_{hj} , 阈值：θ_j, γ_h
- 网络中需要 $d*q+l*q+q+l$ 个参数需要优化



误差逆传播算法（BP算法）

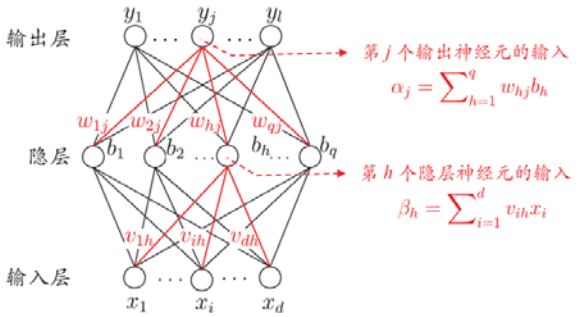
- 前向计算 $(i=1,\cdots,d, h=1,,\cdots,q, j=1,\cdots,l)$
  - Step 1: $b_h = g(\beta_h - \gamma_h), \beta_h = \sum_{i=1}^{d} v_{ih} x_i$
  - Step 2: $\hat{y}_j^k = g(\alpha_j - \theta_j), \alpha_j = \sum_{i=1}^{q} w_{hj} b_h$
  - Step 3: $E_k = \frac{1}{2}\sum_{j=1}^{l}(\hat{y}_j^k - y_j^k)^2$



误差逆传播算法（BP算法）

- 参数优化

BP是一个迭代学习算法, 在迭代的每一轮中采用广义的感知机学习规则对参数进行更新估计, 任意的参数 v 的更新估计式为 $v \leftarrow v + \triangle v.$



误差逆传播算法（BP算法）

- 参数的学习（以w_{hj}为例）

$$\hat{y}_j^k = g\,(\alpha_j - \theta_j),\ \alpha_j = \sum_{i=1}^{q} w_{hj} b_h$$

$$E_k = \frac{1}{2}\sum_{j=1}^{l}(\hat{y}_j^k - y_j^k)^2$$

  - 基于梯度下降策略, 以目标的负梯度方向对参数进行调整.
  - 对误差E_k, 给定学习率 η，得到 $\Delta w_{hj} = -\eta \frac{\partial E_k}{\partial w_{hj}}$

$$\frac{\partial E_k}{\partial w_{hj}} = \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \alpha_j} \cdot \frac{\partial \alpha_j}{\partial w_{hj}} = (\hat{y}_j^k - y_j^k) \cdot \frac{\partial g(\alpha_j - \theta_j)}{\partial \alpha_j} \cdot b_h$$

线性模型一章讲过sigmoid函数的性质 $g(z)' = g(z) \cdot (1 - g(z))$

所以：

$$g_j = -\frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \alpha_j} = (y_j^k - \hat{y}_j^k) \cdot g(\alpha_j - \theta_j)(1 - g(\alpha_j - \theta_j))$$
$$= (y_j^k - \hat{y}_j^k) \cdot \hat{y}_j^k \cdot (1 - \hat{y}_j^k)$$

则： $\Delta w_{hj} = \eta \cdot g_j \cdot b_h$

类似的： $\Delta \theta_j = -\eta \cdot g_j$

误差逆传播算法（BP算法）

- 参数的学习（以v_{ih}为例）

$$b_h = g\,(\beta_h - \gamma_h),\ \beta_h = \sum_{i=1}^{d} v_{ih} x_i$$

$$\hat{y}_j^k = g\,(\alpha_j - \theta_j),\ \alpha_j = \sum_{i=1}^{q} w_{hj} b_h$$

$$E_k = \frac{1}{2}\sum_{j=1}^{l}(\hat{y}_j^k - y_j^k)^2$$

  - 要求解 $\Delta v_{ih} = -\eta \frac{\partial E_k}{\partial v_{ih}}$

$$\frac{\partial E_k}{\partial v_{ih}} = \sum_{j=1}^{l} \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \alpha_j} \cdot \frac{\partial \alpha_j}{\partial b_h} \cdot \frac{\partial b_h}{\partial \beta_h} \cdot \frac{\partial \beta_h}{\partial v_{ih}} = \sum_{j=1}^{l} \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \alpha_j} \cdot \frac{\partial \alpha_j}{\partial b_h} \cdot \frac{\partial b_h}{\partial \beta_h} \cdot x_i$$

$$\frac{\partial \alpha_j}{\partial b_h} \cdot \frac{\partial b_h}{\partial \beta_h} = w_{hj} \cdot g'(\beta_h - \gamma_h) = b_h(1 - b_h) w_{hj}$$

令 $e_h = b_h(1 - b_h)\sum_{j=1}^{l} w_{hj} \cdot g_j$

则 $\Delta v_{ih} = \eta \cdot e_h \cdot x_i$

$\Delta \gamma_j = -\eta \cdot e_h$

误差逆传播算法（BP算法）

- 学习率$\eta \in (0, 1)$控制着算法每一轮迭代中的更新步长，若太长则让容易震荡，太小则收敛速度又会过慢

输入：训练集 $D = \{(\boldsymbol{x}_k, \boldsymbol{y}_k)\}_{k=1}^{m}$；
　　学习率 η.

过程：

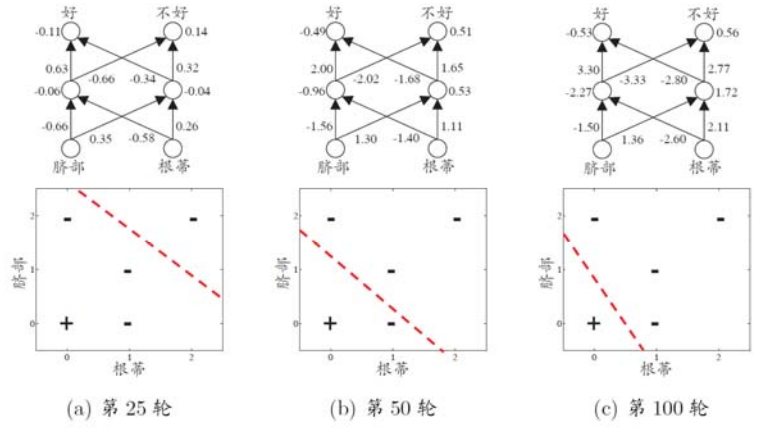
```
在(0,1)范围内随机初始化网络中所有连接权和阈值
repeat
    for all (x_k, y_k) ∈ D do
        根据当前参数和式(5.3) 计算当前样本的输出 ŷ_k;
        根据式(5.10) 计算输出层神经元的梯度项 g_j;
        根据式(5.15) 计算隐层神经元的梯度项 e_h;
        根据式(5.11)-(5.14) 更新连接权 w_hj, v_ih 与阈值 θ_j, γ_h
    end for
until 达到停止条件
```

输出：连接权与阈值确定的多层前馈神经网络

图 5.8　误差逆传播算法

5.3 误差逆传播算法

BP 算法实验



(a) 第 25 轮　(b) 第 50 轮　(c) 第 100 轮

图 5.9　在 2 个属性、5 个样本的西瓜数据上，BP网络参数更新和分类边界的变化情况

误差逆传播算法

标准 BP 算法

- 每次针对单个训练样例更新权值与阈值.
- 参数更新频繁, 不同样例可能抵消, 需要多次迭代.

累计 BP 算法

- 其优化的目标是最小化整个训练集上的累计误差

$$E = \frac{1}{m}\sum_{k=1}^{m} E_k$$

- 读取整个训练集一遍才对参数进行更新, 参数更新频率较低.

实际应用

但在很多任务中, 累计误差下降到一定程度后, 进一步下降会非常缓慢, 这时标准BP算法往往会获得较好的解, 尤其当训练集非常大时效果更明显.

误差逆传播算法

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

提纲

- 神经网络发展史
  - 发展史
  - 受生物神经元网络的启发
- 神经元网络
  - 神经元模型
  - 感知机与多层网络
  - 误差逆传播算法
  - 全局最小与局部最小
- 其他神经元网络

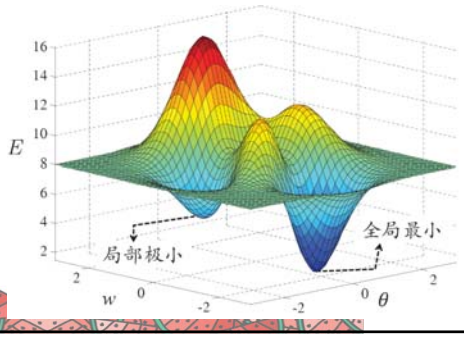
全局最小与局部极小

- 对 $\boldsymbol{w}^*$和 θ^*, 若存在$\epsilon > 0$使得

$$\forall(\boldsymbol{w};\theta) \in \{\|(\boldsymbol{w};\theta) - (\boldsymbol{w}^*;\theta^*)\| \leq \epsilon\},$$

都有$E(\boldsymbol{w};\theta) \geq E(\boldsymbol{w}^*;\theta^*)$成立, 则$(\boldsymbol{w}^*;\theta^*)$为局部极小解;
若度参数空间中任意的$(\boldsymbol{w};\theta)$, 都有 $E(\boldsymbol{w};\theta) \geq E(\boldsymbol{w}^*;\theta^*)$, 则$(\boldsymbol{w}^*;\theta^*)$为全局最小解. 两者对应的$E(\boldsymbol{w}^*;\theta^*)$ 分别称为误差函数的局部最小解和全局最小值.

- 显然参数空间梯度为零的点, 只要其误差函数值小于邻点的误差函数值, 就是局部极小点
- 可能存在多个局部极小值, 但却只会有一个全局极最小值



本页课件来源：周志华《机器学习》及其课件，致谢： 高斌斌

全局最小与局部极小

- “跳出”局部最小的策略
  - 基于梯度的搜索是使用最为广泛的参数寻优方法.
  - 如果误差函数仅有一个局部极小，那么此时找到的局部极小就是全局最小；
  - 然而, 如果误差函数具有多个局部极小，则不能保证找到的解是全局最小.
  - 在现实任务中，通常采用以下策略“跳出”局部极小，从而进一步达到全局最小.

全局最小与局部极小

- “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.

提纲

- 神经网络发展史
  - 发展史
  - 受生物神经元网络的启发
- 神经元网络
  - 神经元模型
  - 感知机与多层网络
  - 误差逆传播算法
  - 全局最小与局部最小
- 其他神经元网络

其他常见神经网络 – RBF网络

- RBF 网络是一种单隐层前馈神经网络, 它使用径向基函数作为隐层神经元激活函数, 而输出层则是隐层神经元输出的线性组合.
- RBF网络模型[Broomhead and Lowe, 1988]

假定输入为d维的向量$\boldsymbol{x}$, 输出为实值, 则RBF 网络可以表示为

$$\varphi(\boldsymbol{x}) = \sum_{i=1}^{q} w_i \rho(\boldsymbol{x}, \boldsymbol{c}_i)$$

其中q为隐层神经元的个数, c_i和 w_i 分别是第 i 神经元对应的中心和权重,$\rho(\boldsymbol{x}, \boldsymbol{c}_i)$是径向基函数.

常用的高斯径向基函数形如

$$\rho(\boldsymbol{x}, \boldsymbol{c}_i) = e^{-\beta_i \|\boldsymbol{x}-\boldsymbol{c}_i\|^2}$$

本页课件来源：周志华《机器学习》及其课件，致谢： 高斌斌

其他常见神经网络 – RBF网络

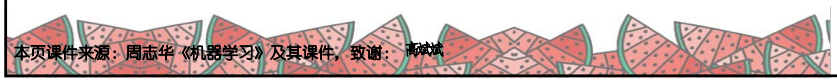
RBF网络性质

具有足够多隐层神经元RBF 神经网络能以任意精度逼近任意连续函数.

[Park and Sandberg, 1991]

RBF网络训练

- Step1:确定神经元中心，常用的方式包括随机采样、聚类等
- Step2:利用BP算法等确定参数



本页课件来源：周志华《机器学习》及其课件，致谢：

其他常见神经网络 – ART网络

竞争学习

竞争学习是神经网络中一种常用的无监督学习策略, 在使用该策略时, 网络的输出神经元相互竞争, 每一时刻仅有一个神经元被激活, 其他神经元的状态被抑制.

ART 网络 [Carpenter and Grossberg, 1987]

- ART 网络是竞争学习的重要代表
- ART 网络由比较层、识别层、识别阈值和重置模块构成
- 比较层负责接收输入样本, 并将其传送给识别层神经元
- 识别层每个神经元对应一个模式类, 神经元的数目可在训练过程中动态增长以增加新的模式类

本页课件来源：周志华《机器学习》及其课件，致谢：

其他常见神经网络 – ART网络

- ART 网络性能依赖于识别阈值
  - 识别阈值高时, 输入样本将会分成比较多、得到较精细分类
  - 识别阈值低时, 输入样本将会分成比较少、产生较粗略分类
- ART 网络的优势
  - ART较好的解决了竞争学习中的“可塑性-稳定性窘境”, 可塑性是指神经网络要有学习新知识的能力；稳定性是指神经网络在学习新知识时要保持对旧知识的记忆.
  - ART 网络可以增量学习或在线学习
- ART 网络的发展

  ART2 网络、FuzzyART 网络、ARTMAP 网络

其他常见神经网络 – SOM 网络

- SOM 网络是一种竞争型的无监督神经网络, 它能将高维数据映射到低维空间（通常为2维）, 同时保持输入数据在高维空间的拓扑结构, 即将高维空间中相似的样本点映射到网络输出层中邻近神经元. [Kohonen, 1982]
- 如图，SOM 网络中的输出层神经元以矩
  神经元都拥有一个权值向量, 网络在接收
  胜神经元, 它决定了该输入向量在低维空

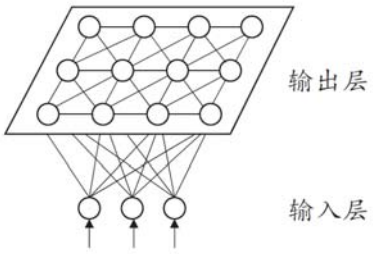


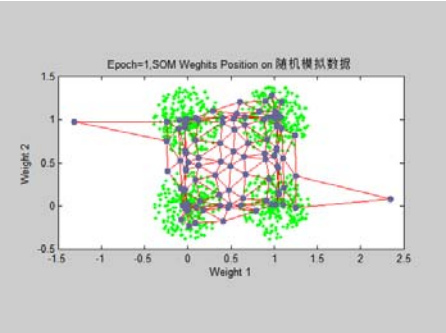
图 5.11 SOM 网络结构

5.5 其他常见神经网络 – SOM 网络

SOM 网络训练

Step1:接受到一个训练样本后, 每个输出层神经元计算该样本与自身携带的权向量之间的距离, 距离最近的神经元成为竞争获胜者

Step2:最佳匹配单元及其近邻神经元的权值将被调整, 使得这些权向量与当前输入样本的距离缩小



本页课件来源：周志华《机器学习》及其课件，致谢： 高斌斌

其他常见神经网络 – 级联相关网络

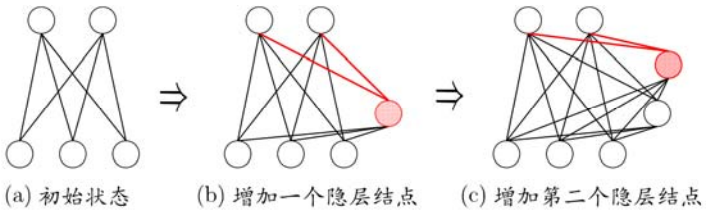
级联相关网络不仅利用训练样本优化连接权值, 阈值参数, 将网络的结构也当做学习的目标之一, 希望在训练过程中找到适合数据的网络结构.
[Fahlman and Lebiere 1990]

级联与相关

级联：建立层次连接的层级结构

相关：最大化神经元的输出与网络误差时间的相关性来训练相关参数

网络优化演示



本页课件来源：周志华《机器学习》及其课件，致谢： 高斌斌

其他常见神经网络 – Elman 网络

递归神经网络

- 允许网络中出现环形结构, 使得神经元的输出反馈回来作为输入信号
- t 时刻网络的输出状态：由 t 时刻的输入状态和 t-1 时刻的网络状态决定

Elman 网络 [Elman 1990]

Elamn 网络是最常用的递归神经网络之一, 结构如图所示, 这种结构与前馈神经网络很相似, 但是隐层神经元的输出被反馈回来, 与下一时刻输入层神经元提供的信号一起, 作为隐层神经元在下一时刻的输入

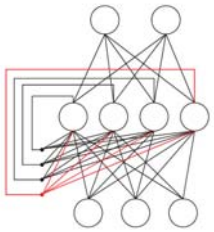


图 5.13 Elman 网络结构

训练算法

推广的BP算法. [Pineda, 1987]

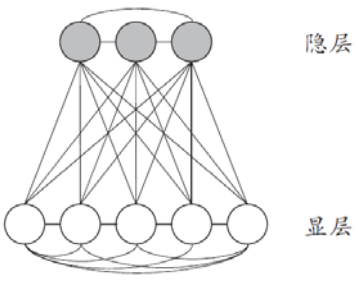
其他常见神经网络 – Boltzmann 机

能量模型

神经网络中有一类模型为网络定义一个“能量”，能量最小化时网络达到理想状态，而网络的训练就是在最小化这个能量函数.

Boltzmann 机

- Boltzmann 机就是一种基于能量的模型
- 结构：显层与隐层
  显层：数据的输入输出
  隐层：数据的内在表达
- 神经元
  布尔型，即只能取0和1 两种状态,
  其中1 表示激活，0表示抑制.



(a) Boltzmann机

5.5 其他常见神经网络 – Boltzmann 机

Boltzmann 机能量 [Ackley et al.，1985]

令状态向量 $\boldsymbol{s} \in \{0,1\}^n$，则其对应的Boltzmann 机能量定义为

$$E(\boldsymbol{s}) = -\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} w_{ij}s_i s_j - \sum_{i=1}^{n}\theta_i s_i$$

其中w_{ij}表示两个神经元之间的连接权值，θ_i 表示神经元的阈值.

Boltzmann 分布

网络中的神经元以任意不依赖与输入值得顺序进行更新，则网络最终将达到 Boltzmann 分布，此时状态向量出现的概率将仅由其能量与所有可能状态向量的能量确定：

$$P(\boldsymbol{s}) = \frac{e^{-E(\boldsymbol{s})}}{\sum_t e^{-E(t)}}$$

本页课件来源：周志华《机器学习》及其课件，致谢： 高斌斌

其他常见神经网络 -- Boltzmann 机

Boltzmann 机训练 [Ackley et al.，1985]

- 将每个训练样本视为一个状态向量，使其出现的概率尽可能大
- 标准的 Boltzmann 机是一个全连接图，训练网络的复杂度很高，这使其难以用于解决现实任务
- 现实中常用受限 Boltzmann 机，简称RBM. RBM仅保留显层与隐层之间的连接，从而将Boltzmann机结构有完全图简化为二部图

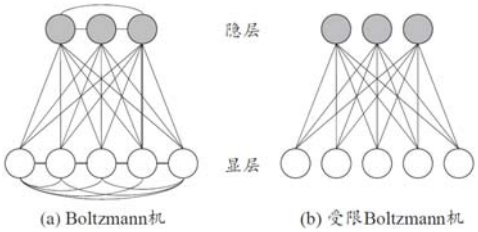


图 5.14 Boltzmann 机与受限 Boltzmann 机

本页课件来源：周志华《机器学习》及其课件，致谢： 高斌斌

其他常见神经网络 --受限 Boltzmann 机

- 受限Boltzmann 机常用“对比散度”（简称：CD）算法 [Hinton, 2010] 来进行训练
  - 假定网络中有d个显层神经元q个隐层神经元，令v和h分别是显层与隐层的状态向量，由于同一层内不存在连接，有

$$P(\boldsymbol{v}|\boldsymbol{h}) = -\prod_{i=1}^{d} P(v_i|\boldsymbol{h}),$$

$$P(\boldsymbol{h}|\boldsymbol{v}) = -\prod_{i=1}^{q} P(h_i|\boldsymbol{v}),$$

  - CD算法对每个训练样本v, 先计算出隐层神经元状态的概率分布, 然后根据这个概率分布采样得到h；类似的方法从h中产生v', 再从v 中产生h'；连接权重的更新公式为：

$$\triangle w = \boldsymbol{v}\boldsymbol{h}^T - \boldsymbol{v}'\boldsymbol{h}'^T$$

本页课件来源：周志华《机器学习》及其课件，致谢： 高斌斌

谢谢大家！