机器学习

第三章 感知机与神经网络

鲍鹏
北京交通大学

目录

神经网络发展史

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,引入了计算能量概念，给出了网络稳定性判断

- 1986年, Rumelhart 等编辑的著作《Parallel Distributed Proceesing:Explorations in the Microstructures of Cognition》提出了反向传播算法，即BP（Back Propagation）算法，是一种按照误差逆向传播算法训练的多层前馈神经网络，目前是应用最广泛的神经网络

- 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

神经网络发展史

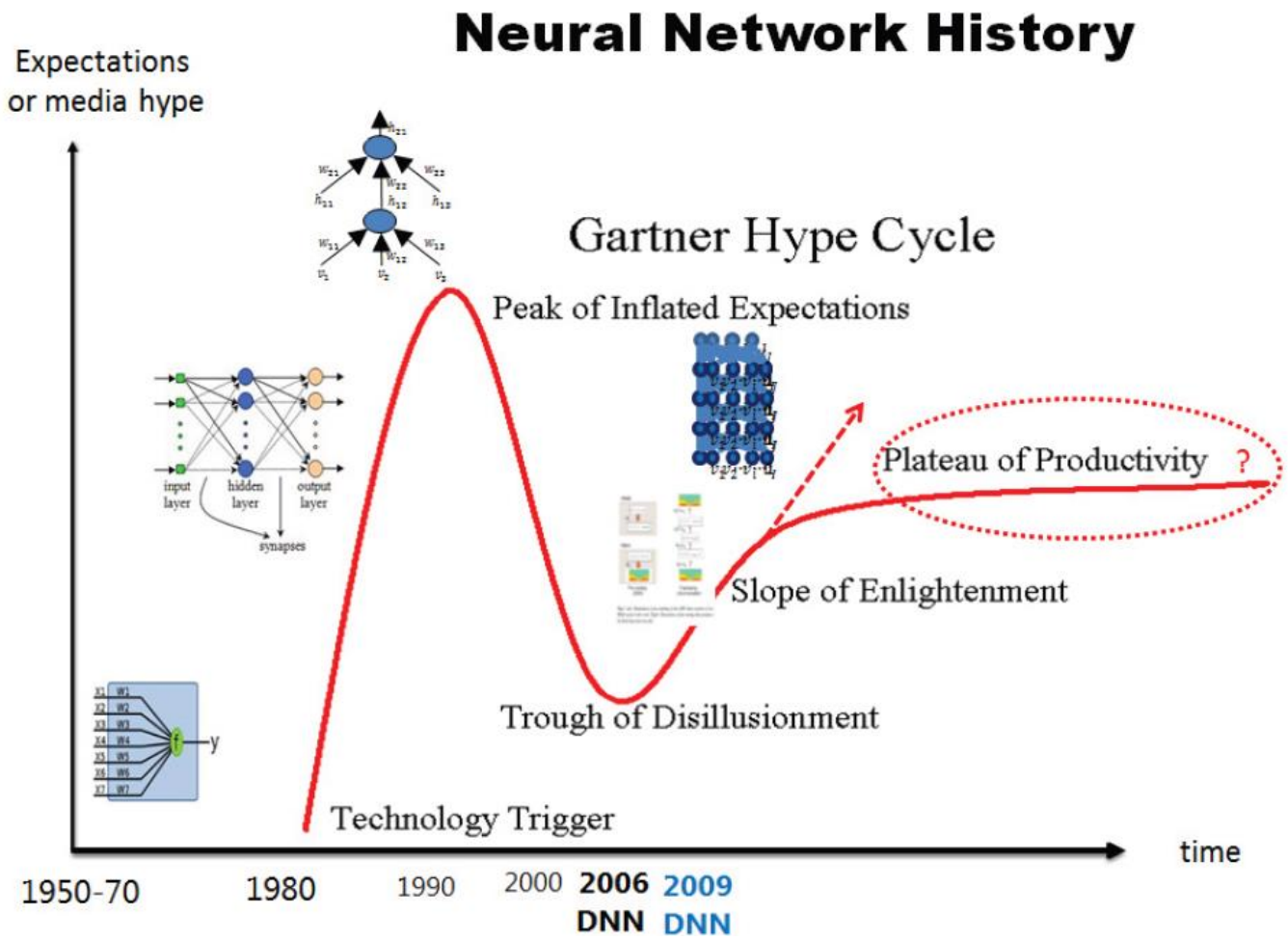


图3.5 神经网络发展历史

神经元模型

神经元模型

口 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：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

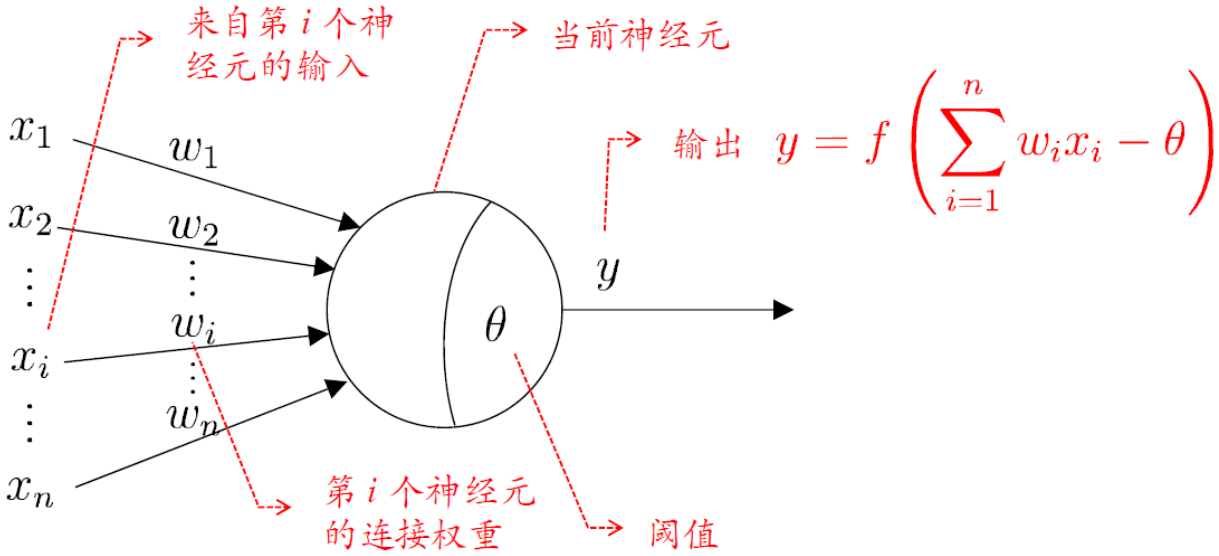
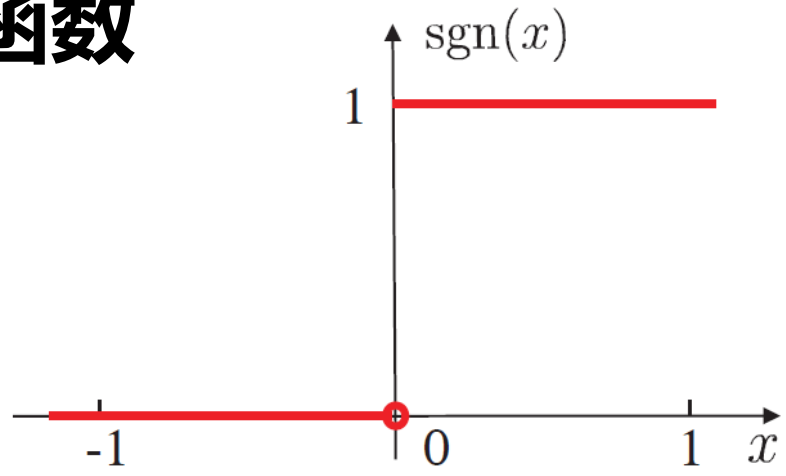


图3.7 M-P神经元模型

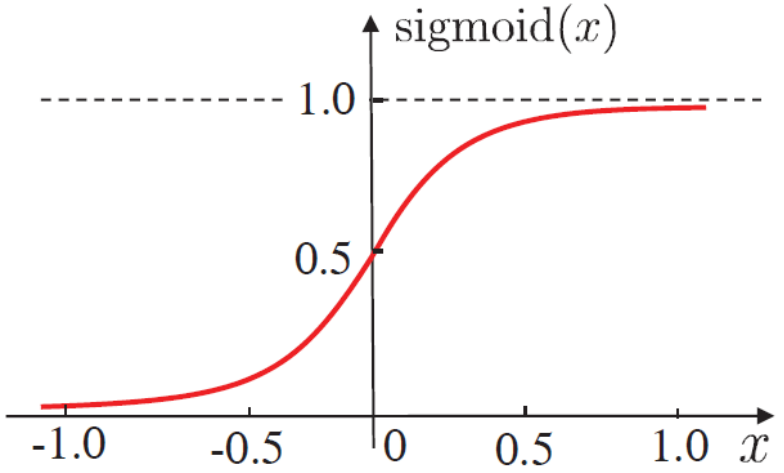
神经元模型

□ 激活函数



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

(a) 阶跃函数



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

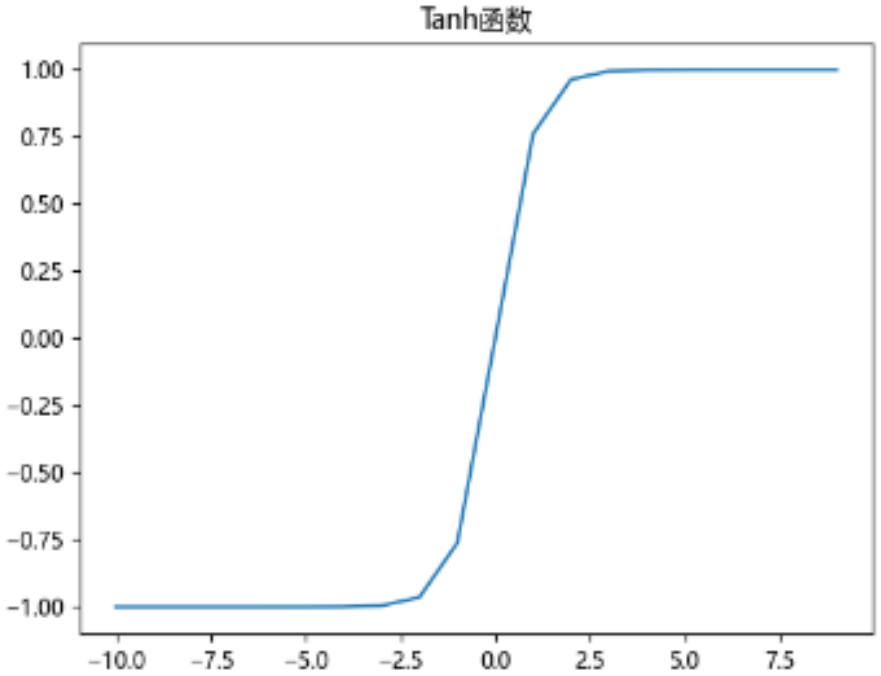
(b) Sigmoid 函数

图3.8 典型的神经元激活函数

- 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

神经元模型

口 激活函数--Tanh



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

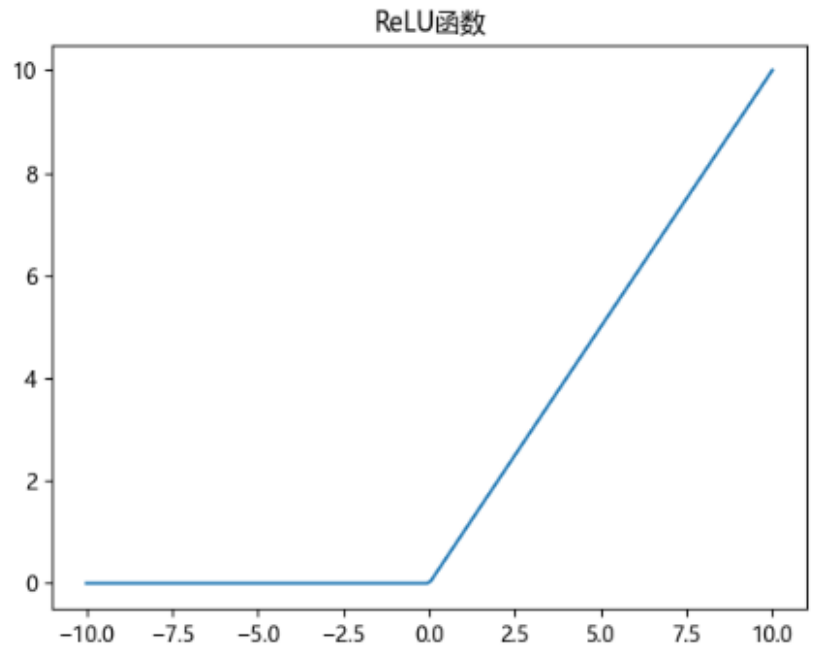
Tanh是双曲正切函数

- Tanh函数值域为（-1，1）
- 整个函数关于原点中心对称

- 形成的曲线比较平滑，梯度较小，因此，不利于反向求导。
- 包含指数级运算，计算量庞大。

神经元模型

□ 激活函数--ReLU



$$ReLU(x) = \max(0, x) = \begin{cases} 0, x < 0 \\ x, x \geq 0 \end{cases}$$

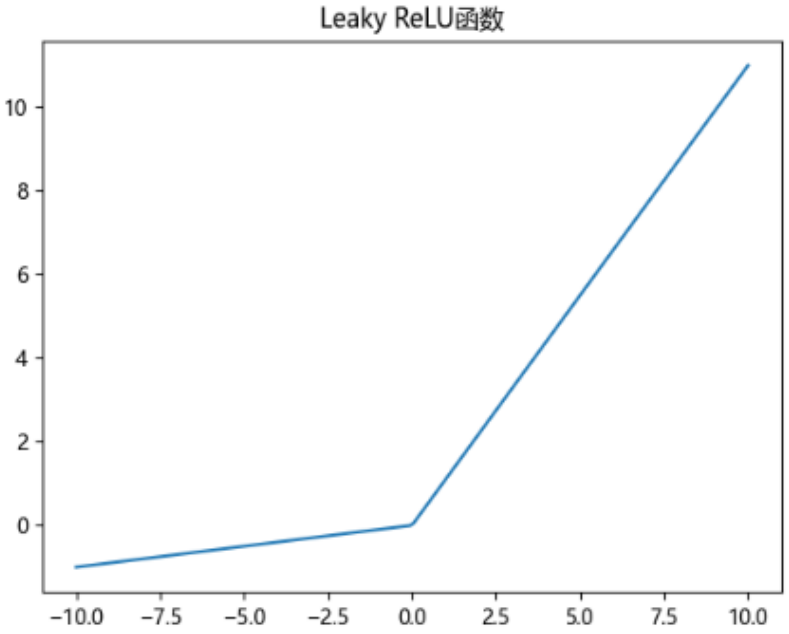
ReLU函数

- 计算速度较快，收敛速度也很快。
- Sigmoid与Tanh会发生梯度消失现象，ReLU解决了这一问题

- 有一些神经元永远不会被激活，将这些神经元命名为死神经元（Dead Neurons），进而会导致相应的参数不能更新。

神经元模型

□ 激活函数-- Leaky ReLU



$$Leaky\ ReLU(x) = \begin{cases} ax, x < 0 \\ x, x \geq 0 \end{cases}$$

Leaky ReLU函数中，较小的参数α用来调整当$x<0$时的梯度，使梯度不为0，解决了ReLU中存在的死神经元问题，更好的保证神经网络的训练。

感知机

感知机

□ 感知机模型（Perceptron）[Rosenblatt,1957]

- **定义:**

  感知机是二类分类的线性模型，其输入是实例的特征向量，输出为实例的类别。

- **数学定义：**

假设输入空间(特征空间)是$X \subseteq R^n$，输出空间是$Y = \{-1, +1\}$。输入$x \epsilon X$表示实例的特征向量，对应于输入空间（特征空间）的点，输出$y \in Y$表示实例的类别，由输入空间到输出空间的如下函数：

$$f(x) = sign(w \cdot x + b)$$

称为感知机。其中w和b为感知机模型参数，$w \in R^n$叫做权值（weight）或权值向量（weight vector），$b \in R$叫做偏置（bias），$w \cdot b$表示w和x的内积。sign是符号函数。即

$$sign(x) = \begin{cases} +1, & x \geq 0 \\ -1, & x < 0 \end{cases}$$

感知机几何解释

- 感知机的几何解释是线性方程：$w \cdot x + b = 0$。
- 对应于特征空间R^n中的一个超平面S，其中w是从超平面的法向量，b是超平面的截距。
- 这个超平面将特征空间划分为两个部分。位于两部分的点（特征向量）分别被分为正、负两类。因此超平面S成为分离超平面（separating hyperplane），如图3.9所示。

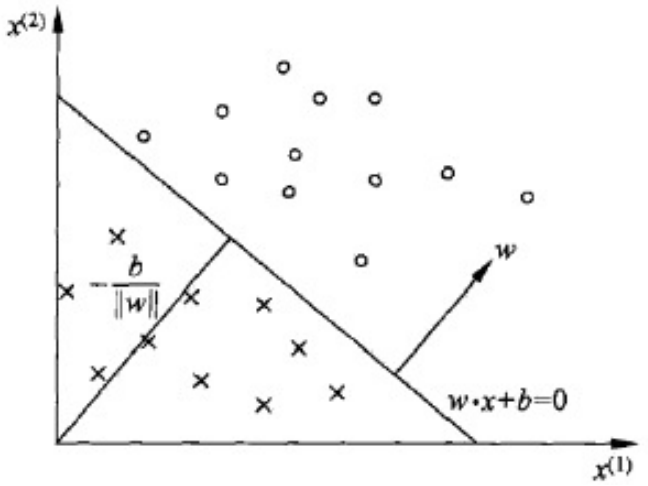


图3.9 感知机模型

感知机模型

- 作为监督学习的一种方法，感知机学习由训练集

$$T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$$

求得感知机模型，即求得模型参数w，b，这里x和y分别是特征向量和类别。通过学习得到的感知机模型，对新的输入样本进行分类。

感知机学习策略

- **数据集的线性可分性**

给定一个数据集$T = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$,其中$x_i \in R^n, y_i \in Y = \{+1, -1\}, i = 1,2, \dots, N$,如果存在某个超平面S（$w \cdot x + b = 0$），能够将数据集的正实例点和负实例点完全正确地划分到超平面的两侧，即对所有的$y_i = -1$的实例i,有$w \cdot x_i + b < 0$,则称数据集T为线性可分数据集（linearly separable data set）;否则，称数据集T线性不可分。

感知机学习策略

- **感知机的目的**

假设训练数据集是线性可分的，感知机学习的目的是求得一个能够将训练集整实例点和负实例点完全正确分开的分离超平面。为了找出这样的超平面，即确定感知机模型参数w和b,需要确定一个学习策略，即定义(经验)损失函数并将损失函数极小化。

感知机学习策略

口 损失函数选择依据

- 自然选择：误分类点的总数，但这样损失函数不是参数w和b的连续可导函数，不易优化。
- 另一选择：误分类点到超平面S的总距离。

空间R^n中任一点x_0到超平面S的距离：$\frac{1}{\|w\|}|w \cdot x_0 + b|$，其中$\|w\|$是w的$L_2$范数。其次，对于误分类的数据$(x_i, y_i)$来说，$-y_i(w \cdot x_i + b) > 0$成立。因此，误分类点$x_i$到超平面S的距离是$-\frac{1}{\|w\|} y_i(w \cdot x_i + b)$。这样，假设超平面S的误分类点集合为M，那么所有误分类点到超平面S的总距离为$-\frac{1}{\|w\|} \sum_{x_i \in M} y_i(w \cdot x_i + b)$。

感知机学习策略

给定训练数据集

$$T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$$

其中，$x_i \in \mathcal{X}$=R^{n} ，$y_i \in \mathcal{Y} = \{+1, -1\}$，$\mathrm{i} = 1,2,\cdots, \mathrm{N}.$

感知机$sign(w \cdot x + b)$学习的损失函数定义为

$$L(w, b) = -\sum_{x_i \in M} y_i(w \cdot x_i + b)$$

其中M为误分类点的集合。

这个损失函数就是感知机学习的经验风险函数。

感知机损失函数

- 损失函数特点：
  - ✓ 该损失函数非负；
  - ✓ 如果没有误分类点，损失函数值为0；
  - ✓ 误分类点越少，误分类点距离超平面越近，损失函数值越小；
  - ✓ 一个特定的样本点的损失函数，在误分类时是参数w,b的线性函数，在正确分类时是0。

因此，给定训练数据集T，损失函数L(w,b)是w,b的连续可导函数。感知机学习的策略是在假设空间中选取使损失函数最小的模型参数w,b，即感知机模型。

$$L(w,b) = -\sum_{x_i \in M} y_i(w \cdot x_i + b)$$

感知机学习算法

- 感知机学习问题转化为求解损失函数式的最优化问题，最优化的方法是**随机梯度下降法**。

$$L(w,b) = -\sum_{x_i \in M} y_i(w \cdot x_i + b)$$

感知机学习算法

- **求解最优化问题**

给定一个数据集$T = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$,其中$x_i \in X = R^n, y_i \in Y = \{+1, -1\}, i = 1,2, \dots, N$。

求参数w和b，使其为以下损失函数极小化问题的解。

$$min_{w,b} L(w, b) = -\sum_{x_i \in M} y_i (w \cdot x_i + b),$$

其中M为误分类点的集合。

感知机学习算法的原始形式

感知机学习算法是误分类驱动的，具体采用随机梯度下降法（stochastic gradient descent）。首先，任意选取一个超平面w_0, b_0,然后用梯度下降法不断地极小化目标函数。极小化过程中不是一次使M中所有误分类点的梯度下降，而是一次随机选取一个误分类点使其梯度下降。

感知机学习算法的原始形式

假设误分类点集合M是固定的，那么损失函数$L(w,b)$的梯度由

$$\min_{w,b} L(w,b) = -\sum_{x_i \in M} y_i \left(w \cdot x_i + b\right)$$

$$\nabla_w L(w,b) = -\sum_{x_i \in M} y_i x_i$$

$$\nabla_b L(w,b) = -\sum_{x_i \in M} y_i$$

给出。

感知机学习算法的原始形式

随机选取一个误分类点(x_i, yi)，对w, b进行更新：

$$w \leftarrow w + \eta y_i x_i$$

$$b \leftarrow b + \eta y_i$$

式中$\eta(0 < \eta \leq 1)$是步长，在统计学习中又称学习率(learning rate)。这样，通过迭代可以期待损失函数$L(w, b)$不断减小，直到0.

感知机学习算法的原始形式

算法1(感知机学习算法的原始形式)：

输入：训练数据集$T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_n,y_n)\}$，其中，$x_i\in\mathcal{X}=\mathrm{R^n}$，$y_i\in\mathcal{Y}=\{+1,-1\}$，$\mathrm{i}=1,2,\cdots,\mathrm{N}$；学习率$\eta(0<\eta\leq 1)$；

输出：w,b；感知机模型$f(x)=sign(w\cdot x+b)$.

(1) 选取初值w_0,b_0

(2) 在训练集汇总选取数据(x_i,y_i)

(3) 如果$y_i(\mathrm{w}\cdot x_i+b)\leq 0$

$$w\leftarrow w+\eta y_i x_i$$

$$b\leftarrow b+\eta y_i$$

(4)转至(2)，直至训练集中没有误分类点。

感知机学习算法的原始形式

口 直观解释

当一个实例点被误分类，即位于分离超平面的错误一侧时，则调整w, b的值，使分离超平面向该误分类点的一侧移动，以减少该误分类点与超平面的距离，直至超平面越过该误分类点使其被正确分类。

感知机学习算法的原始形式

□ 感知机算法实例

如图训练集，正实例点$x_1 = (3,3)^T$， $x_2 = (4,3)^T$，负实例点是$x_3 = (1,1)^T$，试用感知机学习算法的原始形式求感知机模型$f(x) = sign(w \cdot x + b)$。这里，$w = (w^{(1)}, w^{(2)})^T$， $\mathrm{x} = (x^{(1)}, x^{(2)})^T$。

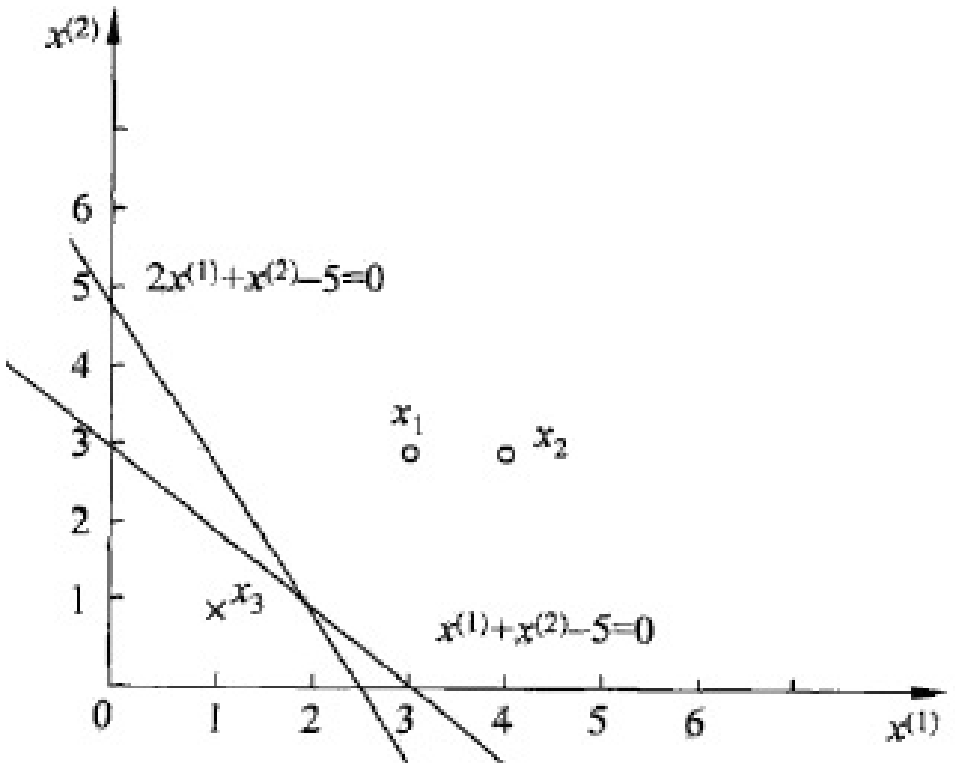


图3.10 感知机示例

感知机学习算法的原始形式

解 构建最优化问题：

$$\min_{w,b} L(w,b) = -\sum_{x_i \in M} y_i \left(w \cdot x_i + b\right)$$

按算法求解w, b。η=1.

(1) 取初值$w_0 = 0, b_0 = 0$

(2) 对x_1=$(3,3)^{\mathrm{T}}$，$y_1(w_0 \cdot x_1 + b_0)$=0,未能正确分类，更新w, b

$$w_1 = w_0 + y_1 x_1 = (3,3)^{\mathrm{T}},\ b_1 = b_0 + y_1 = 1$$

得到线性模型

$$w_1 \cdot x + b_1 = 3x^{(1)} + 3x^{(2)} + 1$$

感知机学习算法的原始形式

(3) 对x_1，x_2，显然，$y_i(w_1 \cdot x_i + b_1)$ >0,被正确分类，不修改w, b；对x_3=(1,1)^T，$y_3(w_1 \cdot x_3 + b_1)$ <0,被误分类，更新w, b。

$$w_2 = w_1 + y_3 x_3 = (2,2)^T, \quad b_2 = b_1 + y_3 = 0$$

得到线性模型

$$w_2 \cdot x + b_2 = 2x^{(1)} + 2x^{(2)}$$

如此继续下去，直到

$$w_7 = (1,1)^T, \quad b_7 = -3$$

$$w_7 \cdot x + b_7 = x^{(1)} + x^{(2)} - 3$$

感知机学习算法的原始形式

对所有数据点$y_i(w_7 \cdot x_i + b_7) > 0$，没有误分类点，损失函数达到极小。

分离超平面为　　$x^{(1)} + x^{(2)} - 3 = 0$

感知机模型为　　$f(x) = sign(x^{(1)} + x^{(2)} - 3)$

迭代过程见表：

迭代次数	误分类点	w	b	$w \cdot x + b$
0		0	0	0
1	x_1	$(3,3)^T$	1	$3x^{(1)} + 3x^{(2)} + 1$
2	x_3	$(2,2)^T$	0	$2x^{(1)} + 2x^{(2)}$
3	x_3	$(1,1)^T$	−1	$x^{(1)} + x^{(2)} - 1$
4	x_3	$(0,0)^T$	−2	−2
5	x_1	$(3,3)^T$	−1	$3x^{(1)} + 3x^{(2)} - 1$
6	x_3	$(2,2)^T$	−2	$2x^{(1)} + 2x^{(2)} - 2$
7	x_3	$(1,1)^T$	−3	$x^{(1)} + x^{(2)} - 3$
8	0	$(1,1)^T$	−3	$x^{(1)} + x^{(2)} - 3$

这是在计算中误分类点先后取x_1，x_3，x_3，x_3，x_1，x_3，x_3得到的分离超平面和感知机模型。

感知机学习算法的原始形式

- 如果在计算中误分类点依次取x_1，x_3，x_3，x_3，x_2，x_3，x_3，x_3，x_1，x_3，x_3，那么得到的分离超平面是$2x^{(1)} + x^{(2)} - 5 = 0.$

- 可见，感知机学习算法由于采用不同的初值或选取不同的误分类点，解可以不同。

感知机学习算法的对偶形式

- 对偶形式的基本想法是，将w和b表示为实例x_i和标记y_i的线性组合的形式，通过求解其系数而求得w和b。不失一般性，在算法中可假设初始值w_0, b_0均为0。对误分类点(x_i, y_i)通过

$$w \leftarrow w + \eta y_i x_i$$

$$b \leftarrow b + \eta y_i$$

感知机学习算法的对偶形式

逐步修改w,b，设修改n次，则w，b关于(x_i, y_i)的增量分别是$\alpha_i y_i x_i$和$\alpha_i y_i$，这里$\alpha_i = n_i \eta$. 这样，最后学习到的w，b可以分别表示为

$$w = \sum_{i=1}^{N} \alpha_i y_i x_i \qquad b = \sum_{i=1}^{N} \alpha_i y_i$$

这里，$\alpha_i \geq 0$，$\mathrm{i} = 1,2,\cdots,\mathrm{N}$，当$\eta$ =1时，表示第i个实例点由于误分而进行更新的次数，实例点更新次数越多，意味着它距离超平面越近，也就越难正确分类。

感知机学习算法的对偶形式

算法2（感知机学习算法的对偶形式）

输入线性可分的数据集$T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_n,y_n)\}$，其中，$x_i\in \mathrm{R}^{\mathrm{n}}$，$y_i\in\{+1,-1\}$，$\mathrm{i}=1,2,\cdots,\mathrm{N}$；学习率$\eta(0<\eta\leq 1)$；

输出：α,b；感知机模型$f(x)=sign(\sum_{j=1}^{N}\alpha_j y_j x_j\cdot x+b)$.

其中$\alpha=(\alpha_1,\alpha_2,\cdots,\alpha_N)^T$

(1) $\alpha\leftarrow 0,b\leftarrow 0$

(2) 在训练集中选取数据(x_i,y_i)

感知机学习算法的对偶形式

(3) 如果$y_i\left(\sum_{j=1}^{N}\alpha_j y_j x_j \cdot x + b\right) \leq 0$

$$\alpha_i \leftarrow \alpha_i + \eta$$

$$b \leftarrow b + \eta y_i$$

(4)转至(2)直到没有误分类数据。

对偶形式中训练实例仅以内积的形式出现。为了方便，可以预先将训练集中实例间的内积计算出来并以矩阵的形式存储，这个矩阵就是所谓的**Gram矩阵**（Gram matrix）。

$$G = \left[x_i \cdot x_j\right]_{N\times N}$$

感知机学习算法的对偶形式

例2 数据同例1，正样本点是$x_1=(3,3)^{\mathrm{T}}$，$x_2=(4,3)^{\mathrm{T}}$，负样本点是$x_3=(1,1)^{\mathrm{T}}$，试用感知机学习算法对偶形式求感知机模型。

解 按照算法2，

（1）取$\alpha_i=0, i=1,2,3, b=0, \eta=1$

（2）计算Gram矩阵

$$G=\begin{bmatrix}18 & 21 & 6\\ 21 & 25 & 7\\ 6 & 7 & 2\end{bmatrix}$$

感知机学习算法的对偶形式

（3）误分条件

$$y_i\left(\sum_{j=1}^{N}\alpha_j y_j x_j \cdot xi + b\right) \leq 0$$

参数更新

$$\alpha_i \leftarrow \alpha_i + 1 \ , \ b \leftarrow b + y_i$$

（4）迭代。过程结果见下表。

（5）

$$w = 2x_1 + 0x_2 - 5x_3 = (1,1)^T$$

$$b = -3$$

感知机学习算法的对偶形式

分离超平面　　$x^{(1)}+x^{(2)}-3=0$

感知机模型　$f(x)=sign\left(x^{(1)}+x^{(2)}-3\right)$

表 2.2　例 2.2 求解的迭代过程

k	0	1	2	3	4	5	6	7
		x_1	x_3	x_3	x_3	x_1	x_3	x_3
α_1	0	1	1	1	2	2	2	2
α_2	0	0	0	0	0	0	0	0
α_3	0	0	1	2	2	3	4	5
b	0	1	0	−1	0	−1	−2	−3

对照例1，结果一致，迭代步骤也是相互对应的。

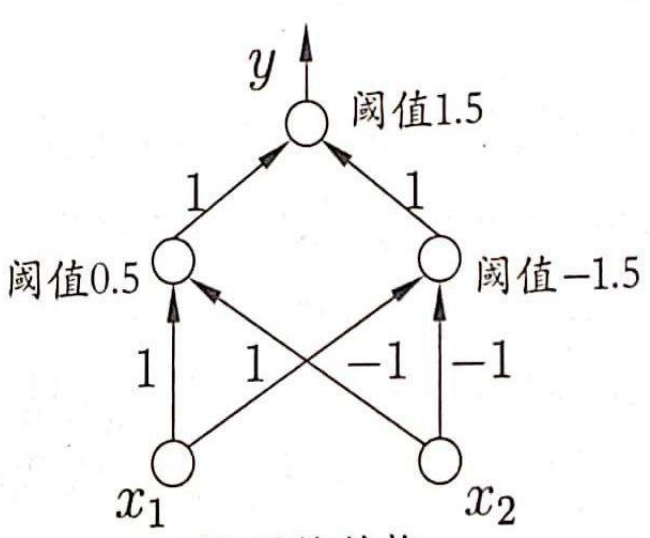
与原始形式一样，感知机学习算法的对偶形式迭代是收敛的，存在多个解

感知机学习能力

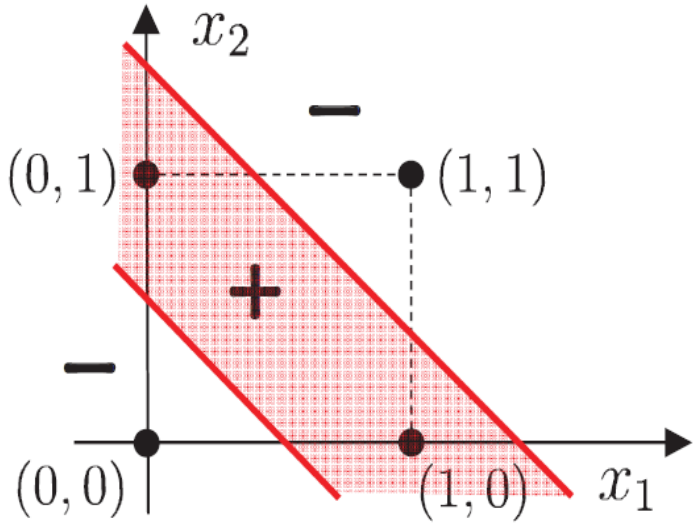
- 若两类模式线性可分，则感知机的学习过程一定会收敛；否则感知机的学习过程将会发生震荡 [Minsky and Papert, 1969]
- 单层感知机的学习能力非常有限，只能解决线性可分问题。
- 对于非线性可分问题，如何求解？——多层感知机

多层感知机

- 解决异或问题的两层感知机



(a) 网络结构

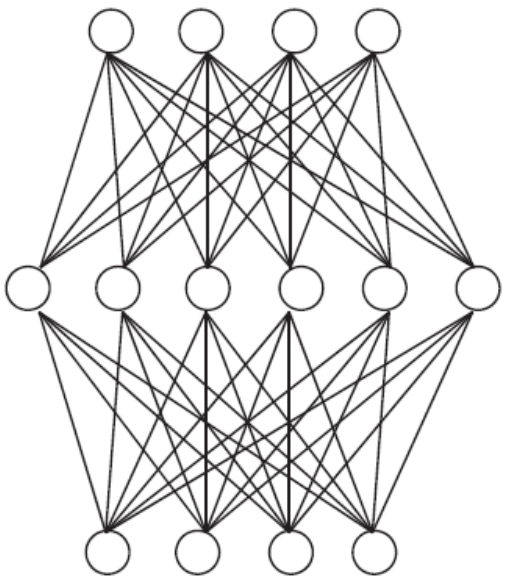


(b) 分类区域

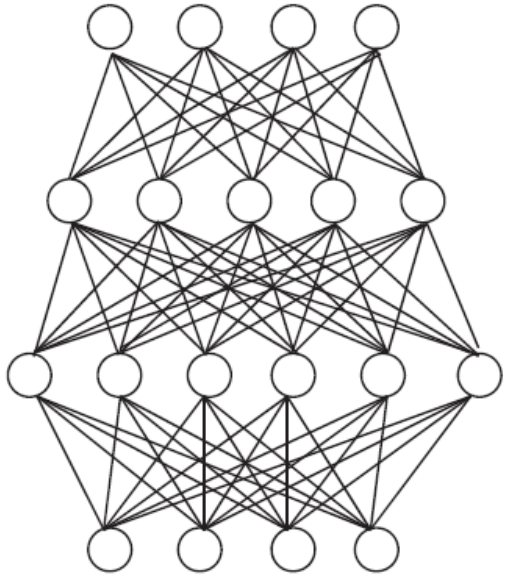
- 输出层与输入层之间的一层神经元, 被称之为隐层或隐含层, 隐含层和输出层神经元都是具有激活函数的功能神经元

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



(a) 单隐层前馈网络



(b) 双隐层前馈网络

误差逆传播算法

误差逆传播算法

误差逆传播算法（Error BackPropagation，简称BP）是最成功的训练多层前馈神经网络的学习算法。

- 给定训练集$D = \{(x_i, y_i)\}, x_i \epsilon R^d, y_i \epsilon R^l, (i = 1,2, \dots, m)$
- 即输入示例由d个属性描述, 输出l维实值向量,q个隐层神经元的多层前向前馈网络结构

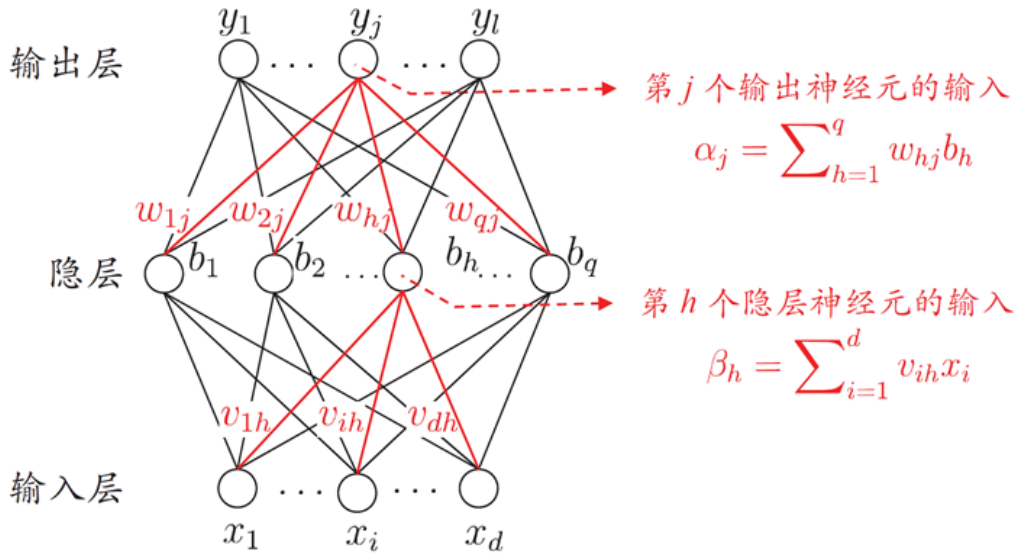
- 记号：

  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

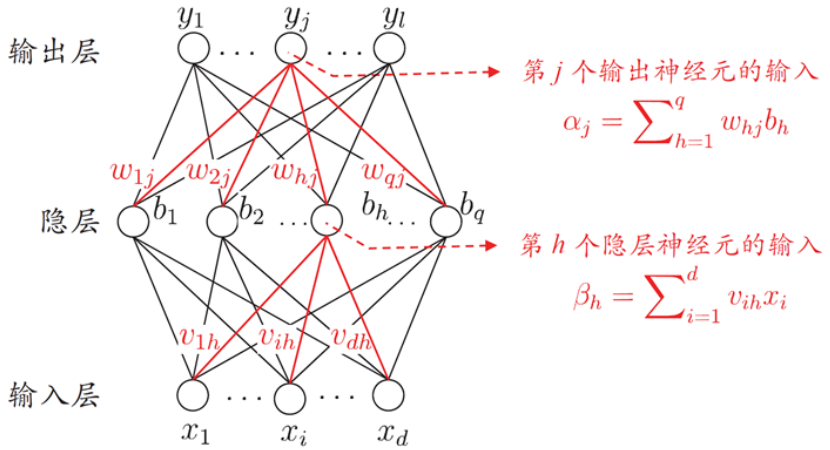
  v_{ih}: 输入层与隐层神经元之间的连接权重

  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

对于样例（x_k, y_k），假设网络的实际输出为$\hat{y}_k$



- 前向计算

  step1: $b_h = f(\beta_h - \gamma_h), \beta_h = \sum_{i=1}^{d} v_{ih} x_i$

  step2: $\hat{y}_j^k = f(\alpha_j - \theta_j), \alpha_h = \sum_{i=1}^{q} w_{hj} b_h$ ①

  step3: $E_k = \frac{1}{2} \sum_{j=1}^{l} (\hat{y}_i^k - y_j^k)^2$

- 参数数目

  权重：v_{ih}, w_{hj}；阈值：θ_j, γ_h，$(i \epsilon (1,d), h \epsilon (1,q), j \epsilon (1,l))$

  因此网络中需要$(d + l + 1)q + l$个参数需要优化

- 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

$$v \leftarrow v + \Delta v$$

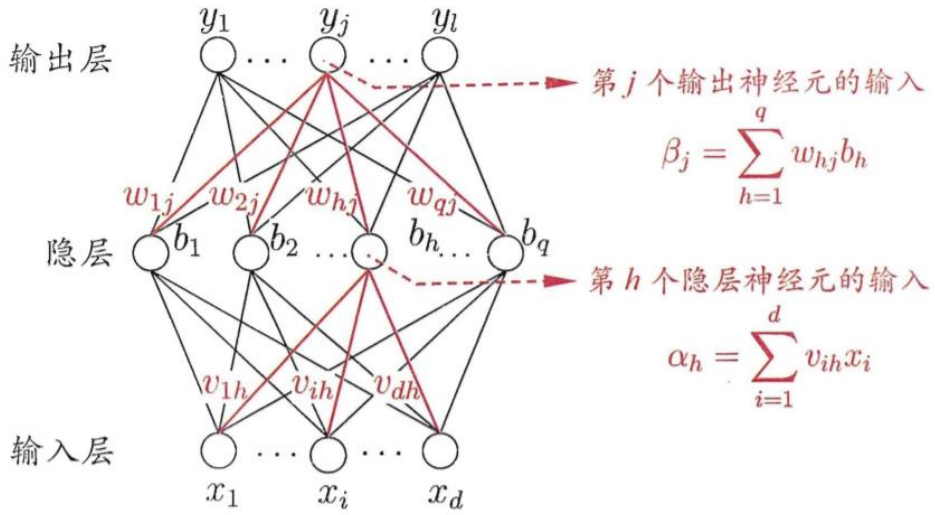
误差逆传播算法（BP学习算法）

- BP算法基于梯度下降策略，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
- 输出层权系数的调整：$\Delta w_{hj} = -\eta \frac{\partial E_k}{\partial w_{jk}}$
- 注意到w_{hj}先影响到第j个输出层神经元的输入值β_j，在影响到其输出值$\hat{y}_j^k$，然后影响到E_k，有：$\frac{\partial E_k}{\partial w_{hj}} = \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \beta_j} \cdot \frac{\partial \beta_j}{\partial w_{hj}}$
- 则定义反向误差传播信号：

$$g_i = -\frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \beta_j} = -(\hat{y}_j^k - y_j^k) f'(\beta_j - \theta_j)$$

$= \hat{y}_j^k (1 - \hat{y}_j^k)(y_j^k - \hat{y}_j^k)$ ②　　$\Delta w_{hj} = \eta b_j g_h$ ③

- 同理可推出：$\Delta\theta_j = -\eta \frac{\partial E_k}{\partial \theta_j} = -\eta \frac{\partial E_k}{\partial \hat{y}_j^k} \frac{\partial \hat{y}_j^k}{\partial \theta_j} = -\eta g_j$



误差逆传播算法

- 隐藏层权系数的调整：$\Delta v_{ih} = -\eta \frac{\partial E_k}{\partial v_{ih}} = -\eta \frac{\partial E_k}{\partial b_h} \frac{\partial b_h}{\partial a_h} \frac{\partial a_h}{\partial v_{ih}} = -\eta \frac{\partial E_k}{\partial b_h} \frac{\partial b_h}{\partial a_h} x_i$
- 一个v_{ih}权值会影响所有的β_j,定义反向误差传播信号：

$$e_h = -\frac{\partial E_k}{\partial a_h} = -\frac{\partial E_k}{\partial b_h} \frac{\partial b_h}{\partial a_h} = -\sum_{j=1}^{l} \frac{\partial E_k}{\partial \beta_j} \frac{\partial \beta_j}{\partial b_h} f'(\alpha_h - \gamma_h) = \sum_{j=1}^{l} w_{hj} g_j f'(\alpha_h - \gamma_h)$$

$$= b_h(1 - b_h) \sum_{j=1}^{l} w_{hj} g_j \qquad ③$$

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

- 同理，可推导出$\Delta \gamma_h = -\eta e_h$

误差逆传播算法

输入：训练集$D=\{(x_k,y_k)\}_{k=1}^{m}$；学习率$\eta$.

过程：

1：在（0,1）范围内随机初始化网络中所有连接权和阈值

2：repeat

3:　　for all $(x_k,y_k)\epsilon D$ do

4:　　根据当前参数和式①计算当前样本的输出$\hat{y}_k$

5：　　根据式②计算输出层神经元的梯度项g_j；

6：　　根据式③计算隐藏层神经元的梯度项e_h；

7:　　同理根据前述算式更新连接权$w_{hj,}\ v_{ih}$与阈值θ_j,γ_h

8:　　end for

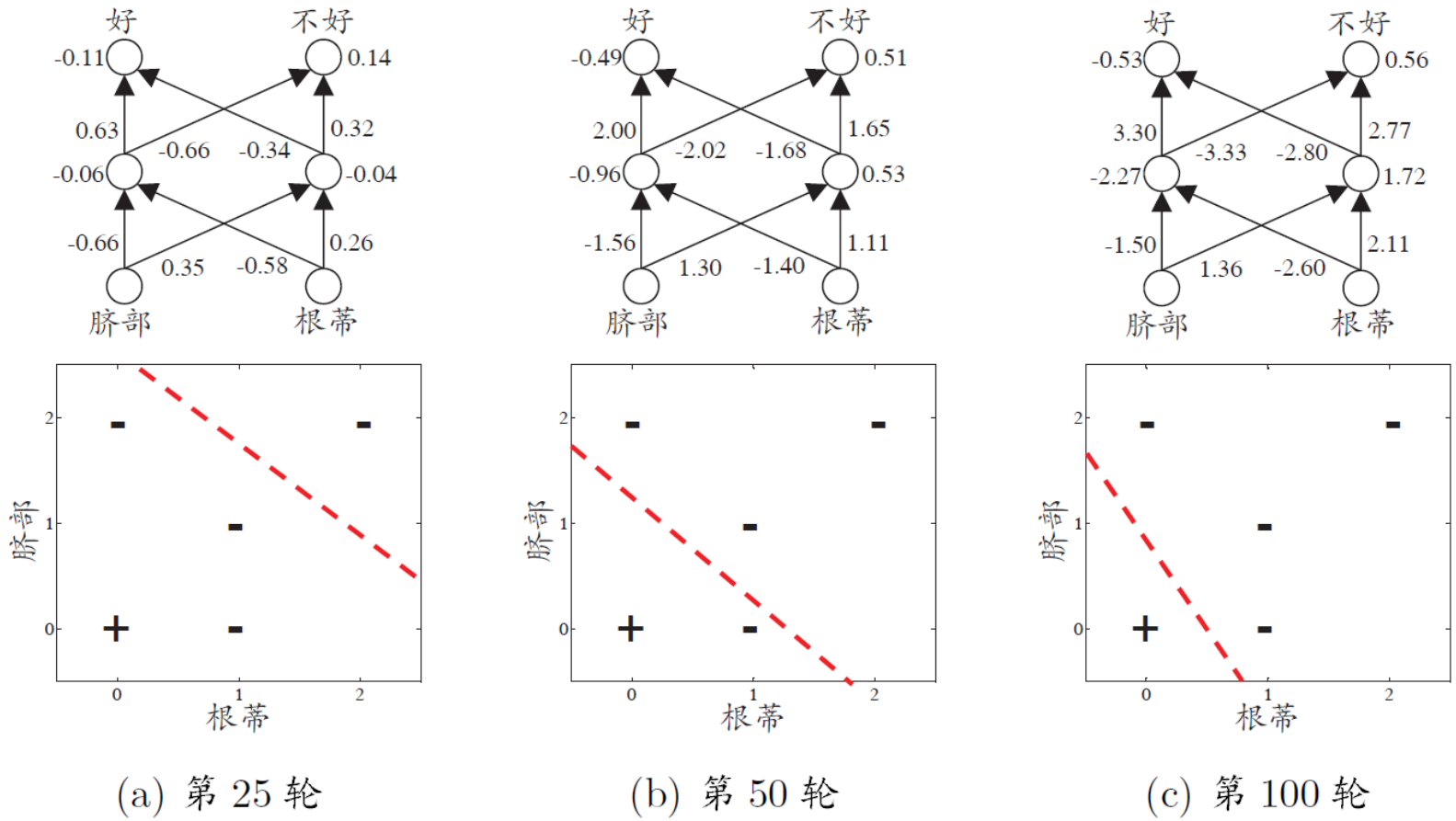
9:　until达到停止条件

输出：连接权与阈值确定的多层前馈神经网络

图3.14 误差逆传播算法

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3.15 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

口 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。

口 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。

口 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。

其他常见神经网络

其他常见神经网络

RBF网络 [Broomhead and Lowe, 1988]

- 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

假定输入为 d 维的向量 X, 输出为实值, 则RBF网络可以表示为

$$\varphi(x) = \sum_{i=1}^{q} w_i \rho(X, C_i)$$

其中 q 为隐层神经元的个数, C_i 和 w_i 分别是 i 第神经元对应的中心和权重, $\rho(X, C_i)$ 是径向基函数。

常用的高斯径向基函数形如

$$\rho(X, C_i) = e^{-\beta_i \|x - C_i\|^2}$$

其他常见神经网络

□ RBF网络 [Broomhead and Lowe, 1988]

- 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

其他常见神经网络

ART网络[Carpenter and Grossberg,1987]

- 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

其他常见神经网络

ART网络[Carpenter and Grossberg,1987]

- 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

  1.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

  2.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 [Kohonen, 1982]

- 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.

- 如图，SOM网络中的输出层神经元以矩阵方式排列在二维空间中，每个神经元都拥有一个权值向量，网络在接收输入向量后，将会确定输出层获胜神经元，它决定了该输入向量在低维空间中的位置.

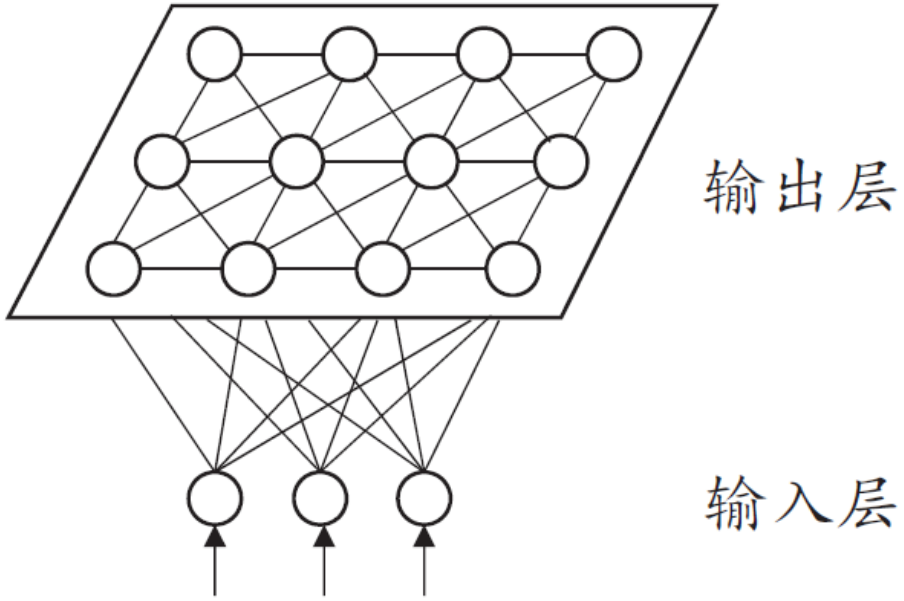


图3.16 SOM网络结构

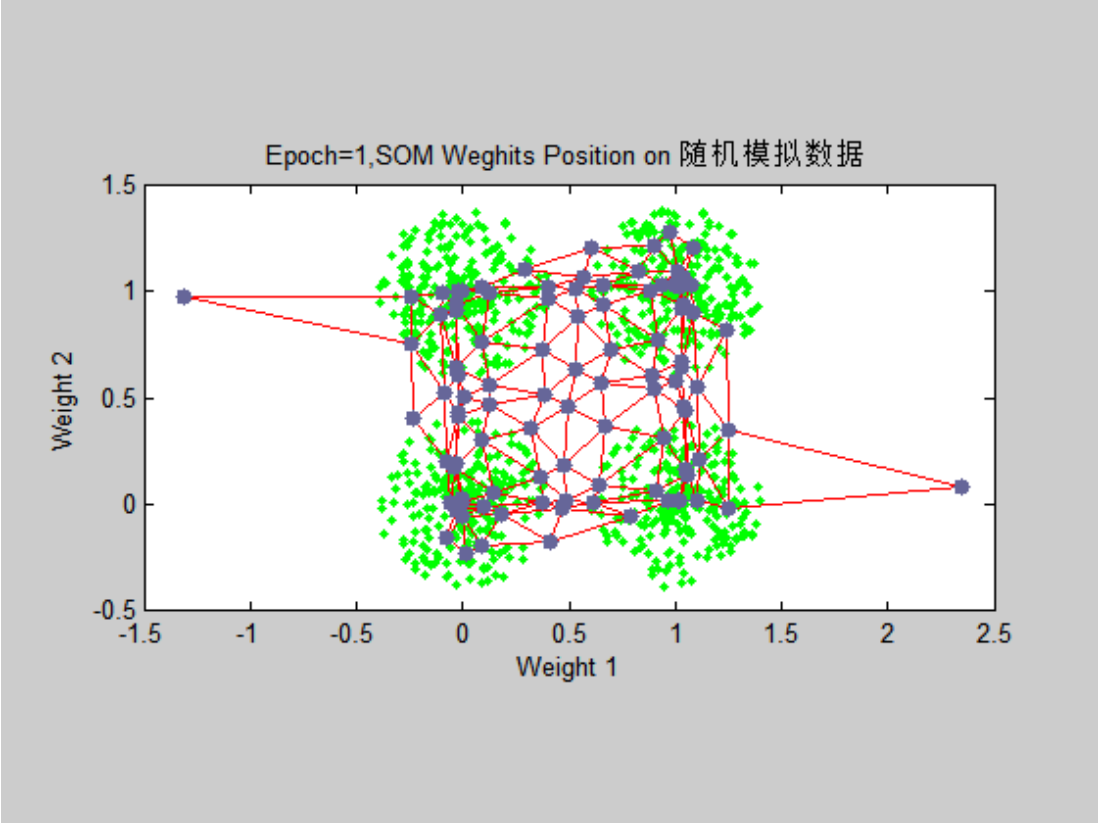
其他常见神经网络

□ SOM网络 [Kohonen, 1982]

- 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



其他常见神经网络

口 级联相关网络 [Fahlman and Lebiere 1990]

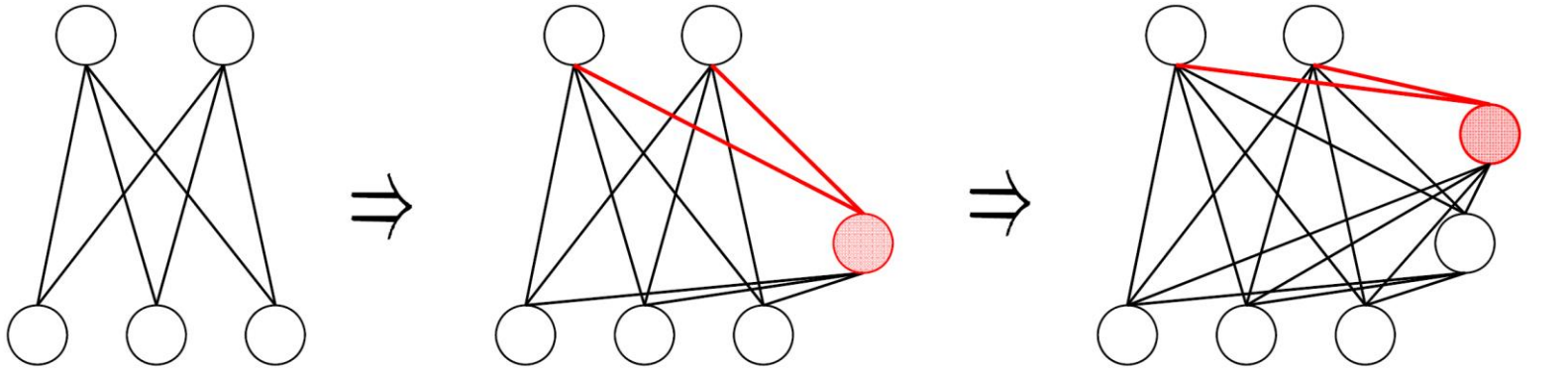
级联相关网络不仅利用训练样本优化连接权值, 阈值参数, 将网络的结构也当做学习的目标之一, 希望在训练过程中找到适合数据的网络结构.

- 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的相关性来训练相关参数

- 网络优化演示



(a) 初始状态　(b) 增加一个隐层结点　(c) 增加第二个隐层结点

其他常见神经网络

□ Elman网络 [Elman 1990]

- 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 时刻的网络状态绝定

- Elman网络

Elman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

- 训练算法

  推广的BP算法.[Pineda, 1987]

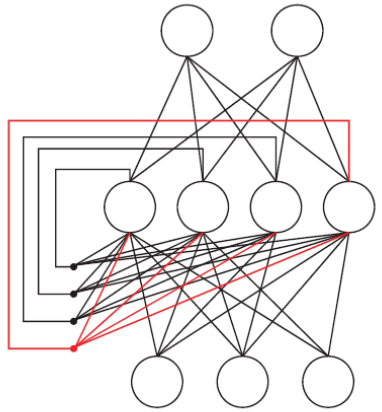


图3.17 Elman网络结构

其他常见神经网络

□ Boltzmann机 [Ackley et al.,1985]

- 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

1. 结构：显层与隐层

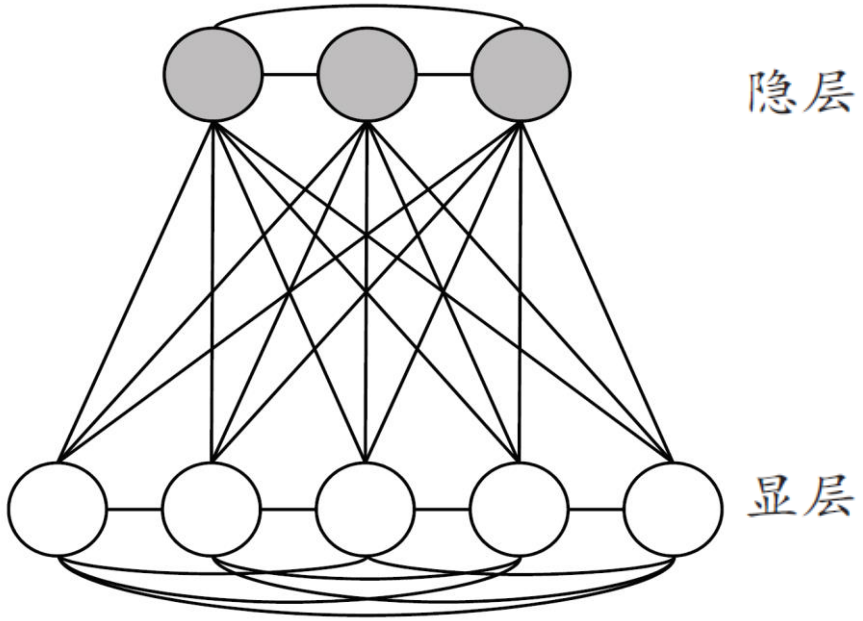
显层：数据的输入输出

隐层：数据的内在表达

2. 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

深度学习

深度学习模型

典型的深度学习模型就是很深层的神经网络。

◆模型复杂度

- 增加隐层神经元的数目（模型宽度）
- 增加隐层数目（模型深度）
- 从增加模型复杂度的角度看，增加隐层的数目比增加隐层神经元的数目更有效。这是因为增加隐层数不仅增加额拥有激活函数的神经元数目，还增加了激活函数嵌套的层数。

◆复杂模型难点

- 多隐层网络难以直接用经典算法（例如标准BP算法）进行训练，因为误差在多隐层内逆传播时，往往会“发散”而不能收敛到稳定状态。

复杂模型训练方法

◆预训练+微调

- 预训练：监督逐层训练是多隐层网络训练的有效手段，每次训练一层隐层结点，训练时将上一层隐层结点的输出作为输入，而本层隐结点的输出作为下一层隐结点的输入，这称为“预训练”。
- 微调：在预训练全部完成后，再对整个网络进行微调训练。微调一般使用BP算法。

◆例子：深度信念网络 [Hintonet al. , 2006]

- 结构：每一层都是一个受限Boltzmann机
- 训练方法：无监督预训练+BP微调

◆分析

- 预训练+微调的做法可以视为将大量参数分组，对每组先找到局部看起来比较好的设置，然后再基于这些局部较优的结果联合起来进行全局寻优。

口复杂模型训练方法

◆权共享

- 一组神经元使用相同的连接权值。
- 权共享策略在卷积神经网络(CNN)[LeCun and Bengio, 1995; LeCun et al. , 1998]中发挥了重要作用。

◆卷积神经网络

- 结构：CNN复合多个卷积层和采样层对输入信号进行加工,然后在连接层实现与输出目标之间的映射。

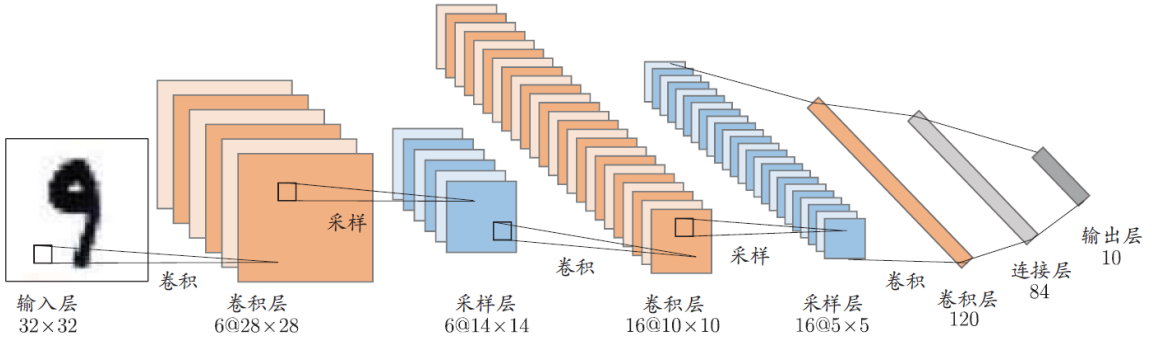


图3.19 卷积神经网络用于手写数字识别

深度学习

卷积神经网络

- **卷积层**：每个卷基层包含多个特征映射,每个特征映射是一个由多个神经元构成的“平面”， 通过一种卷积滤波器提取的一种特征。
- **采样层**：亦称“汇合层”,其作用是基于局部相关性原理进行亚采样,从而在减少数据量的同时保留有用信息。
- **连接层**：每个神经元被全连接到上一层每个神经元,本质就是传统的神经网络,其目的是通过连接层和输出层的连接完成识别任务。

卷积神经网络激活函数

- 在CNN中通常将 sigmoid 激活函数替换为修正的线性函数 $f(x) = \max(0, x)$

卷积神经网络训练

- CNN可以用BP进行训练,但在训练中,无论是卷积层还是采样层,每一组神经元都是用相同的连接权,从而大幅减少了需要训练的参数数目。

深度学习

理解深度学习

- “特征工程”VS“特征学习”或者“表示学习”
- 特征工程由人类专家根据现实任务来设计,特征提取与识别是单独的两个阶段;

手工设计特征 → 分类识别

- 特征学习通过深度学习技术自动产生有益于分类的特征,是一个端到端的学习框架。

学习生成特征 → 分类识别

参考文献

[1]《统计学习方法》，清华大学出版社，李航著，2019年出版

[2]《机器学习》，清华大学出版社，周志华著，2016年出版

[3] Andrew Ng. Machine Learning[EB/OL]. Stanford University,2014. https://www.coursera.org/course/ml

[4] Christopher M. Bishop, Pattern Recognition and Machine Learning, Springer-Verlag, 2006

[5] Minsky, Marvin and Papert, Seymour. Perceptrons : An Introduction to Computational Geometry. The MIT Press, 1969.

[6] DE Rumelhart, Hinton G E, Williams R J. Learning Representations by Back Propagating Errors[J]. Nature, 1986, 323(6088):533-536.

[7] Bishop C M. Neural Networks for Pattern Recognition[J]. Advances in Computers, 1993, 37:119-166.

[8] Lecun Y, Bengio Y. Convolutional Networks for Images, Speech, and Time-Series[J]. Handbook of Brain Theory & Neural Networks, 1995.

谢 谢！