南昌大学
NANCHANG UNIVERSITY

统计机器学习

主讲人：彭振华

数学与计算机学院

2026年

目录

CONTENTS

•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

•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

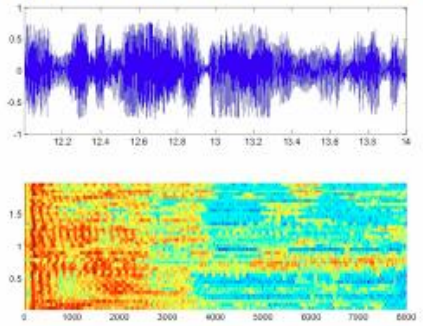
•第三阶段

- 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、大模型等众多领域都取得了较大的成功

Images & Video

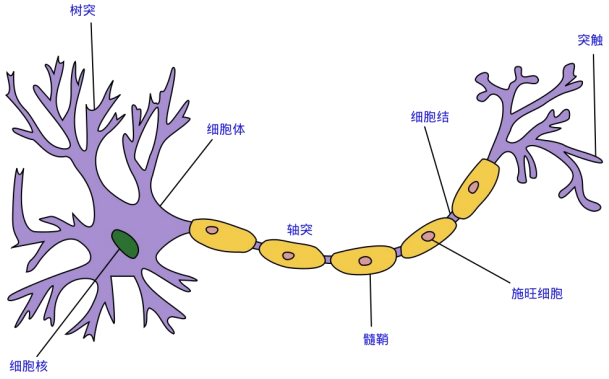
Text & Language

Speech & Audio

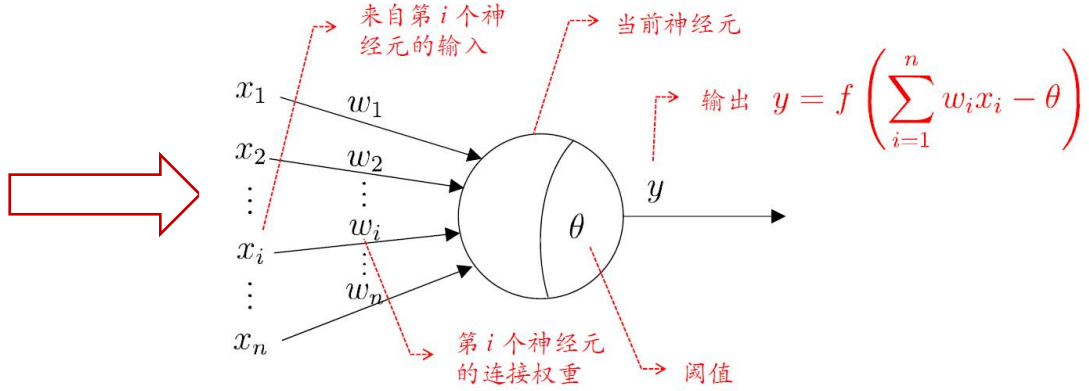


神 经 网 络 基 础

- 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]
- 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
- M-P模型，1943年，沃伦·麦卡洛克（Warren McCulloch）和沃尔特·皮兹（Walter Pitts）

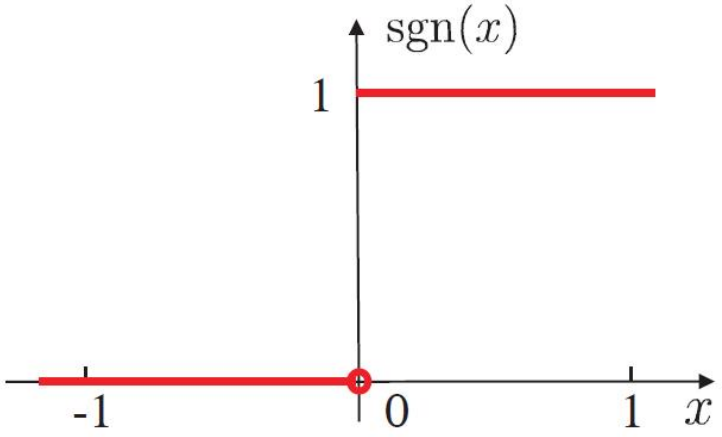


生物神经元



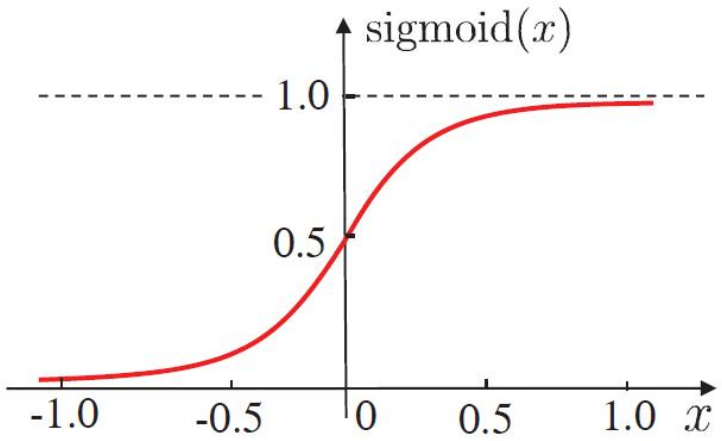
人工神经元：M-P模型

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

- 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

- 输入为实例的特征向量，输出为实例的类别，取+1和-1；
- 感知机对应于输入空间中将实例划分为正负两类的分离超平面，属于**判别模型**；
- 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
- 导入基于误分类的**损失函数**；
- 利用**梯度下降法**对损失函数进行极小化；
- 1957年由Rosenblatt提出，是神经网络与支持向量机的基础。

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

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

- 称为感知机，
- 模型参数：权值向量w，偏置b，
- 符号函数：

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

- **误分类点到超平面的总距离**

- 距离：$\frac{1}{\|w\|}|w \cdot x_0 + b|$

误分类点：$-y_i(w \cdot x_i + b) > 0$

误分类点距离：$-\frac{1}{\|w\|} y_i(w \cdot x_i + b)$

总距离：

$$-\frac{1}{\|w\|}\sum_{x_i \in M} y_i(w \cdot x_i + b)$$

损失函数(M为误分类点的数目)：

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

- 求解最优化问题：

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

- 随机梯度下降法，
- 首先任意选择一个超平面，w，b，然后不断极小化目标函数，损失函数L的梯度：

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

- 选取误分类点更新：

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

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

学习率 $\eta(0<\eta\leqslant 1)$；

输出：w,b；感知机模型 $f(x)=\mathrm{sign}(w\cdot x+b)$

（1）选取初值 w_0,b_0

（2）在训练集中选取数据 (x_i,y_i)

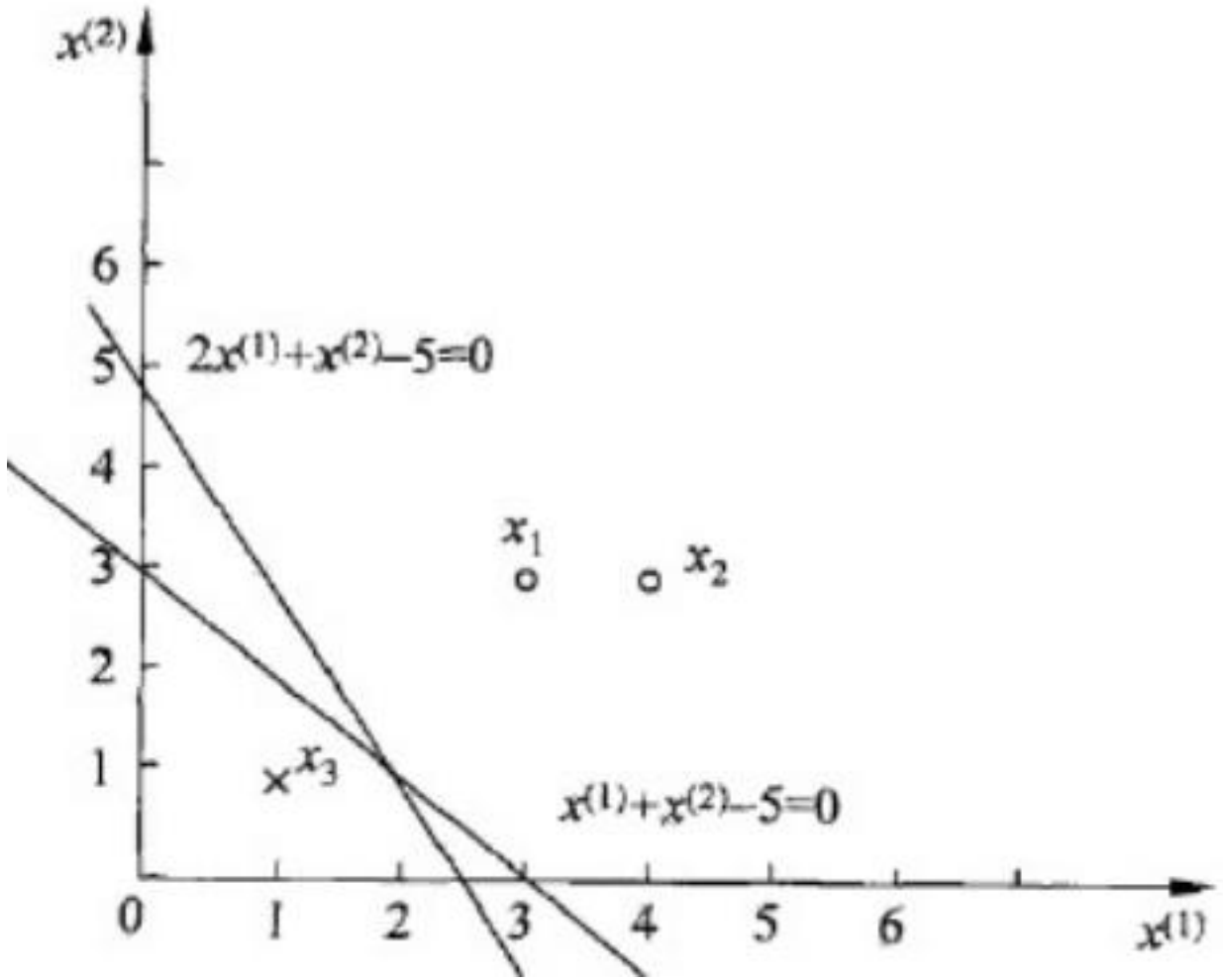
（3）如果 $y_i(w\cdot x_i+b)\leqslant 0$

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

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

（4）转至（2），直至训练集中没有误分类点

●例：正例：$x_1=(3,3)^T$，$x_2=(4,3)^T$　　负例：$x_3=(1,1)^T$



- 解：构建优化问题：$\min_{w,b} L(w,b) = -\sum_{x_i \in M} y_i(w \cdot x + b)$
- 求解：w，b，

$\eta = 1$

（1）取初值 $w_0 = 0$，$b_0 = 0$

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

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

- 得线性模型：$w_1 \cdot x + b_1 = 3x^{(1)} + 3x^{(2)} + 1$
-

（3）x_2，显然，$y_i(w_1 \cdot x_i + b_1) > 0$，被正确分类，

对 $x_3 = (1,1)^T$，$y_3(w_1 \cdot x_3 + b_1) < 0$，被误分类，

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

- 得到线性模型：$w_2 \cdot x + b_2 = 2x^{(1)} + 2x^{(2)}$
- 如此继续下去：

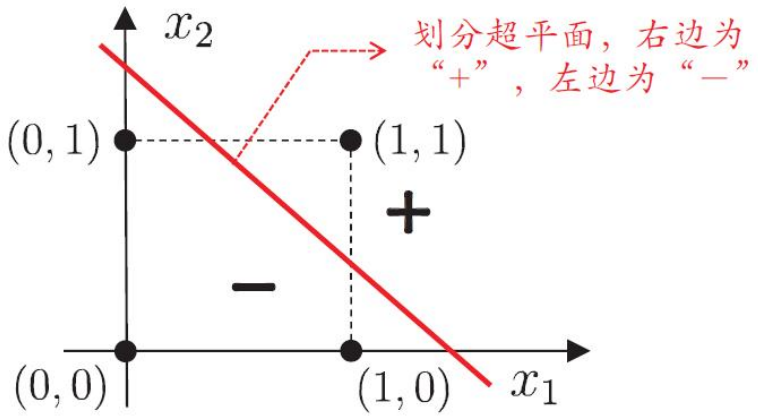
$$w_7 = (1,1)^T, \quad b_7 = -3$$

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

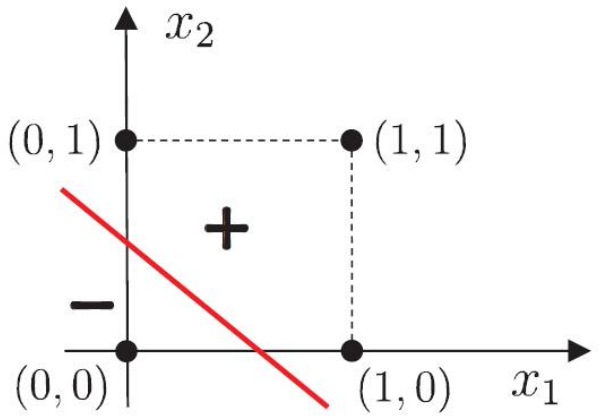
- 分离超平面：$x^{(1)} + x^{(2)} - 3 = 0$
- 感知机模型：$f(x) = \text{sign}(x^{(1)} + x^{(2)} - 3)$

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	$x^{(1)} + x^{(2)} - 3$

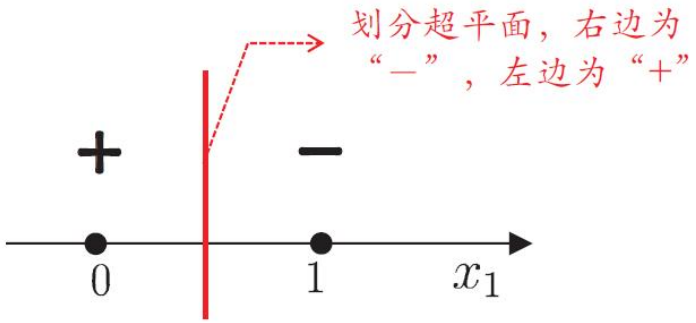
- 感知机求解异、或、非问题



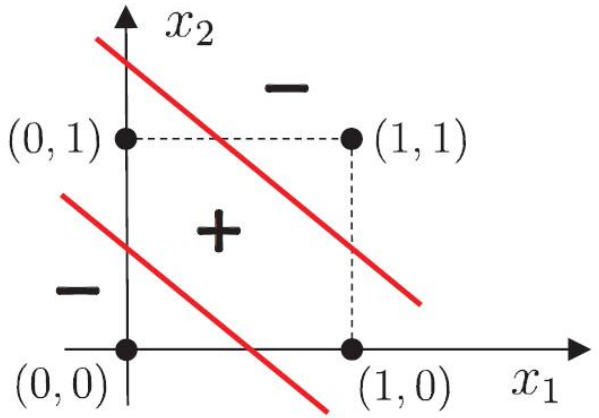
(a) “与”问题 $(x_1 \wedge x_2)$



(b) “或”问题 $(x_1 \vee x_2)$



(c) “非”问题 $(\neg x_1)$



(d) “异或”问题 $(x_1 \oplus x_2)$

感知机练习

1.正例：

$$x_1 = (4,2), x_2 = (3,4)$$

反例：

$$x_3 = (0,1)$$

2.正例：

$$x_1 = (0,0), x_2 = (1,0), x_3 = (0,1)$$

反例：

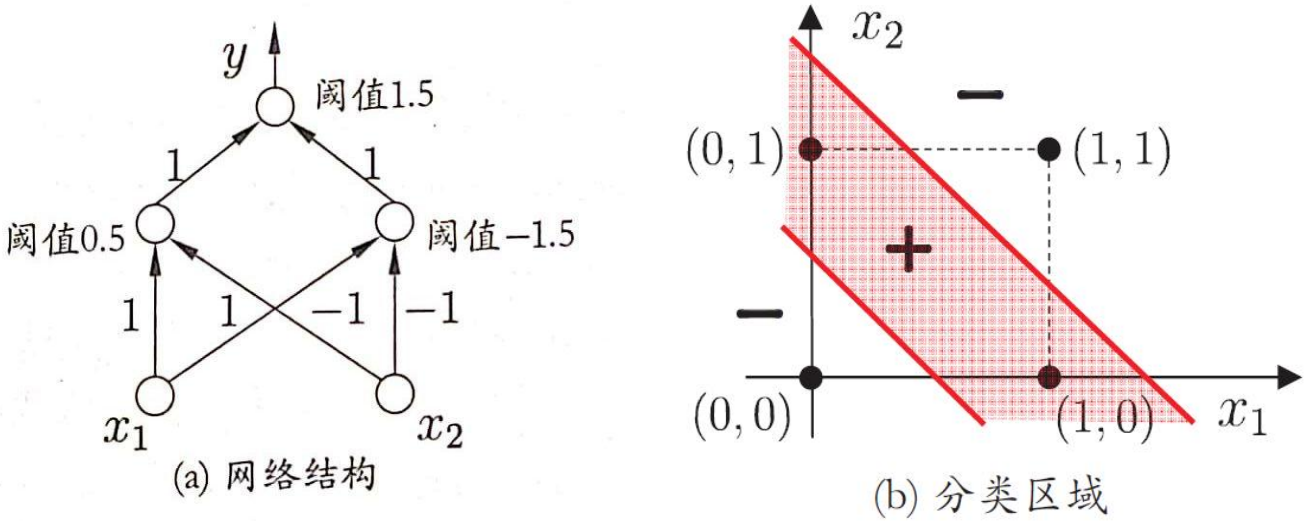
$$x_4 = (1,1)$$

- 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

●多层感知机

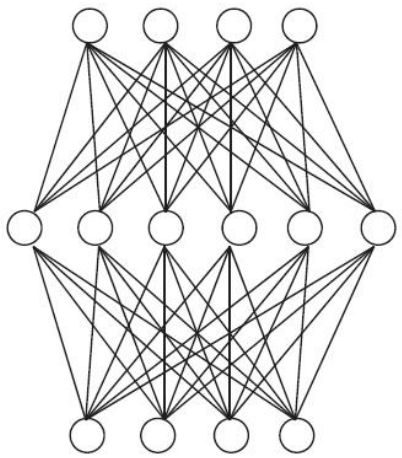
□解决异或问题的两层感知机



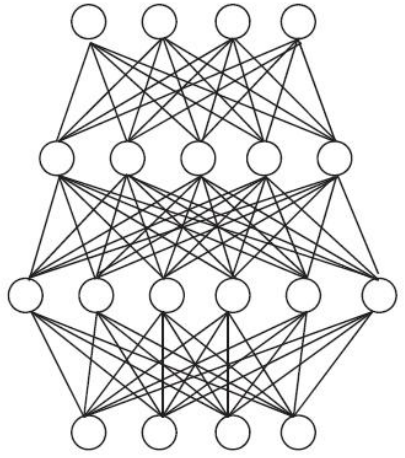
- 输出层与输入层之间的一层神经元，被称之为隐层或隐含层，隐含层和输出层神经元都是具有激活函数的功能神经元

●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（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.

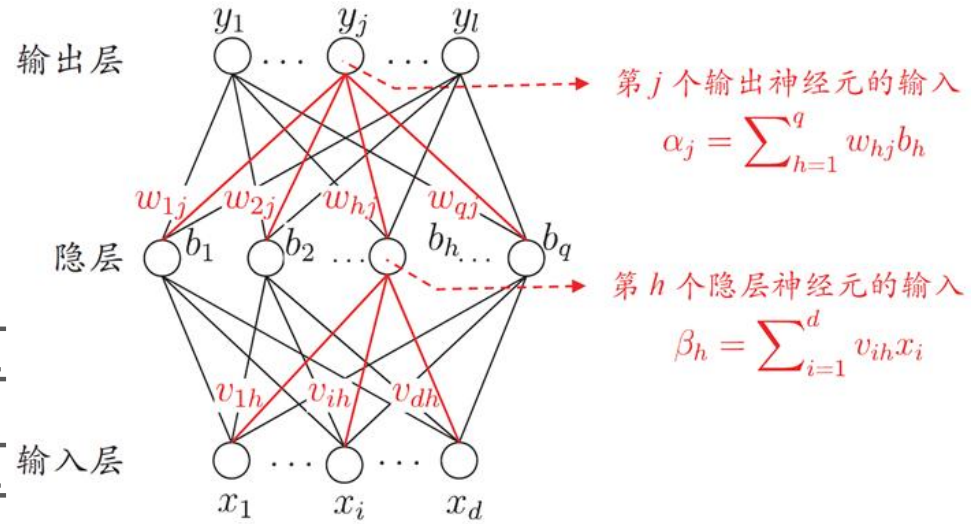
- 给定训练集 $D=\{(\boldsymbol{x}_i,y_i)\},\boldsymbol{x}_i\in R^d,y_i\in R^l,(i=1,2,\cdots,m)$
- 即输入示例由 d 个属性描述，输出 l 维实值向量，q 个隐层神经元的多层前向前馈网络结构
- 记号：

θ_j ：输出层第 j 个神经元阈值

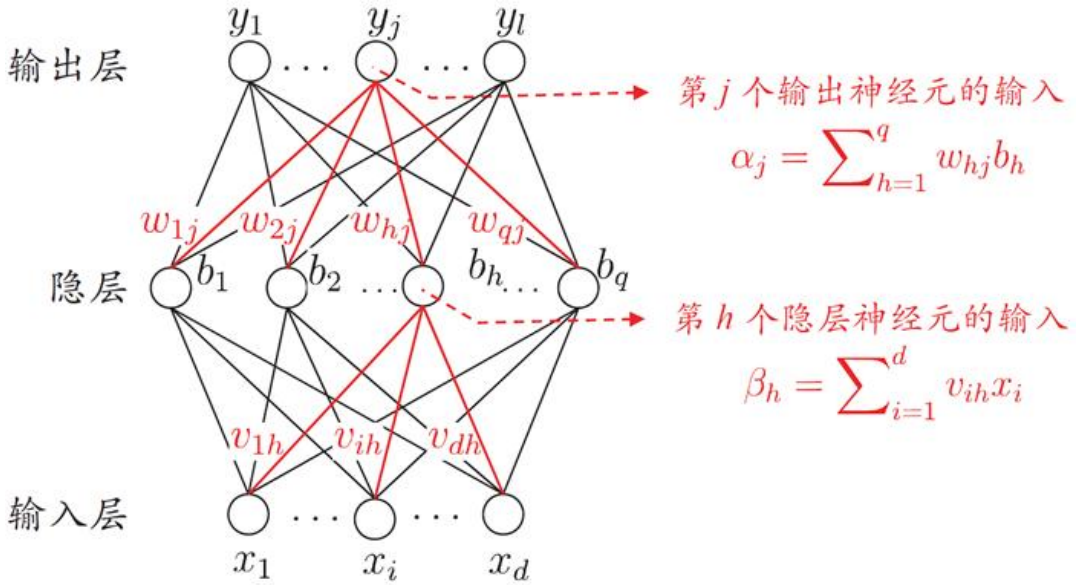
γ_h ：隐含层第 h 个神经元阈值

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



对于样例 $(\boldsymbol{x}_k, \boldsymbol{y}_k)$，假设网络的实际输出为 $\hat{\boldsymbol{y}}_k$



- **前向计算**

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

  step2: $\hat{y}_j^k = f(\alpha_j - \theta_j), \alpha_h = \sum_{i=1}^{q} w_{hj} b_h$ (5.3)

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

- **参数数目**

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

  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

$$v \leftarrow v + \triangle v.$$

- 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

$$\triangle w_{hj} = -\eta \frac{\partial E_k}{\partial w_{jk}}$$

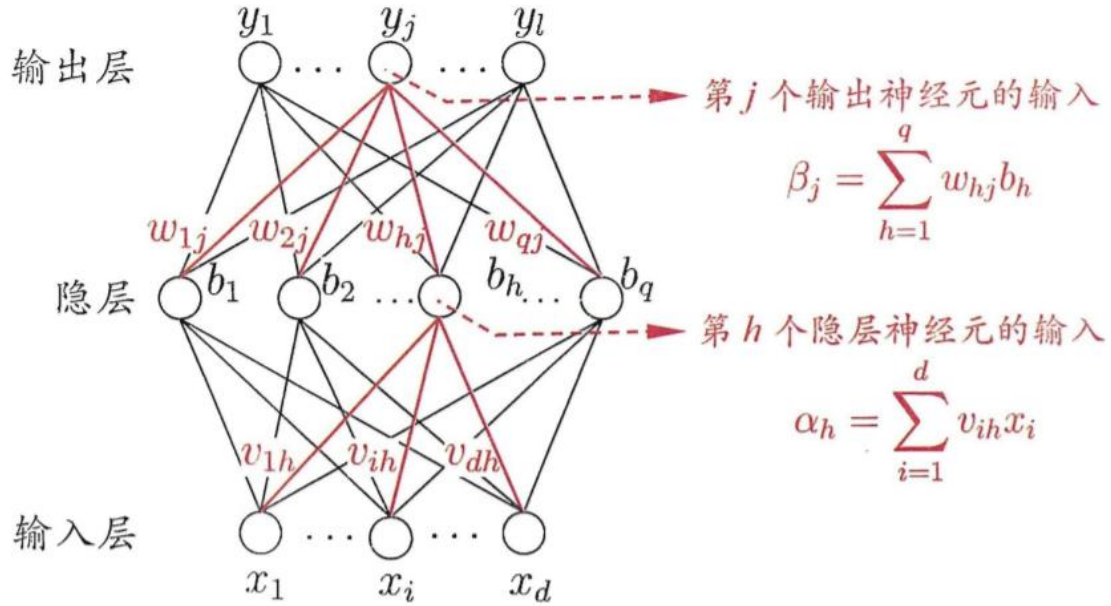
$$\frac{\partial E_k}{\partial w_{hj}} = \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \beta_j} \cdot \frac{\partial \beta_j}{\partial_{hj}}$$

$$g_j = -\frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \beta_j}$$

$$= -(\hat{y}_j^k - y_j^k) f'(\beta_j - \theta_j)$$

$$= \hat{y}_j^k (1 - \hat{y}_j^k)(y_j^k - \hat{y}_j^k) \quad (5.10)$$

$$\triangle w_{hj} = \eta b_j g_h \quad (5.11)$$



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 $(\boldsymbol{x}_k, \boldsymbol{y}_k) \in D$ **do**
    根据当前参数和式(5.3) 计算当前样本的输出 $\hat{\boldsymbol{y}}_k$;
    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

- 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

- “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.

- 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.

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

●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

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.
- 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.

- Torch http://torch.ch/

感 谢 观 看

统 计 机 器 学 习

主讲人：彭振华

数学与计算机学院

2026年