第10章　图像模式分类

最困难的事情就是认识自己。

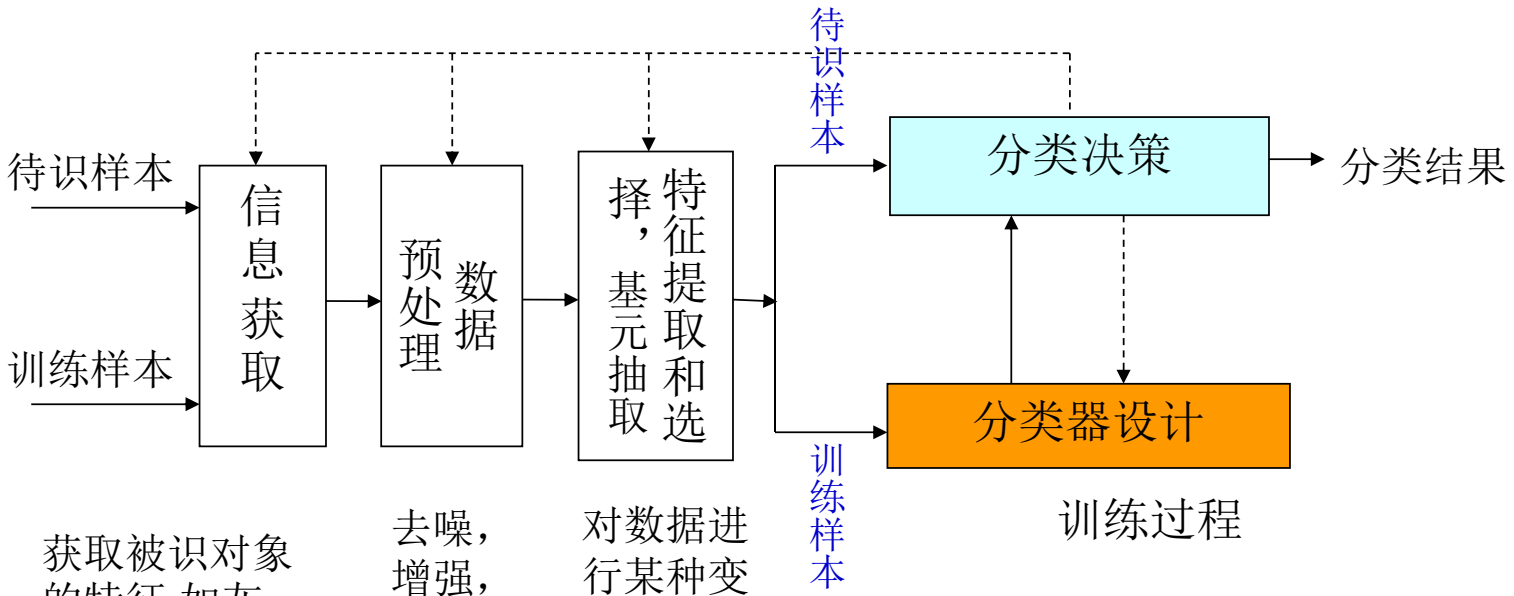
——古希腊名言

模式识别基本概念

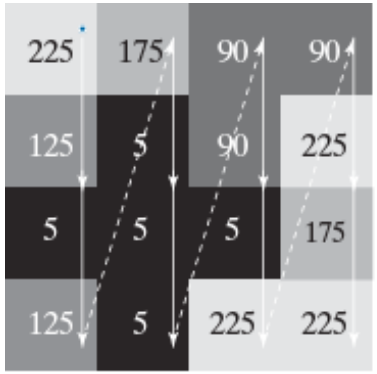
- 模式是对客体的定量或结构的描述
  - 模式采用特征进行描述
    - 如重量、颜色（如红、绿、蓝三个分量的值）、尺寸等、形状、结构，都可作为模式的特征。
- 模式向量（特征矢量）：如果一组特征可用一组测量数据表征，则可表示成矢量形式，称之为模式向量
- 模式类：具有某些共同特征的模式的集合
- 模式识别：是根据特征或属性识别模式的自动技术，即依据一定的量度或观测基础把待识别模式划分到各自的模式类的过程

- 模式识别方法
  - 统计模式识别
    - 根据模式在特征空间中的分布，利用统计决策原理对特征空间进行划分，从而识别不同特征的对象
  - 人工神经网络
    - 将若干个处理单元(即神经元)通过一定的互连模型连结成一个网络，这个网络通过一定的机制可以一定程度上模仿人的神经系统的动作过程，从而可用于识别分类
  - 句法(结构)模式识别
    - 利用待识别对象的结构特征，同一类模式对象的结构描述(关系描述)可由一定的规则产生。通过分析待识对象的结构描述对应的产生规则(语法)达到分类目的。

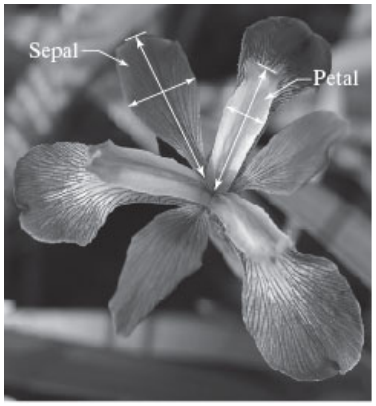
一般模式识别过程


待识样本
训练样本
信息获取
数据预处理
特征提取和选择，基元抽取
待识样本
训练样本
分类决策
分类结果
分类器设计
训练过程
获取被识对象的特征,如灰度、色彩、声音、压力、温度等
去噪，增强，恢复，图像的二值化、分割等
对数据进行某种变换，得到最能反映分类本质的特征
特征：如形状特征参数，声音的频率，物体的重量 等

模式向量示例

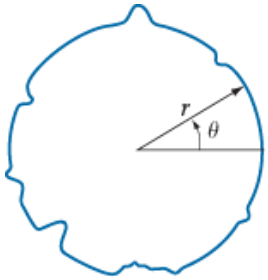


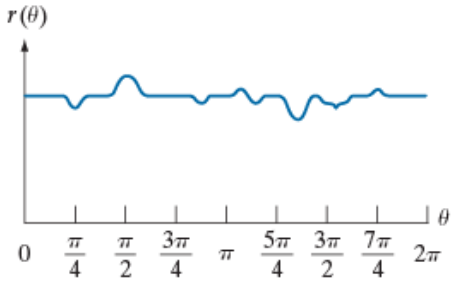
$$\mathbf{x} = \begin{bmatrix} 225 \\ 125 \\ 5 \\ \vdots \\ 225 \\ 175 \\ 225 \end{bmatrix}$$



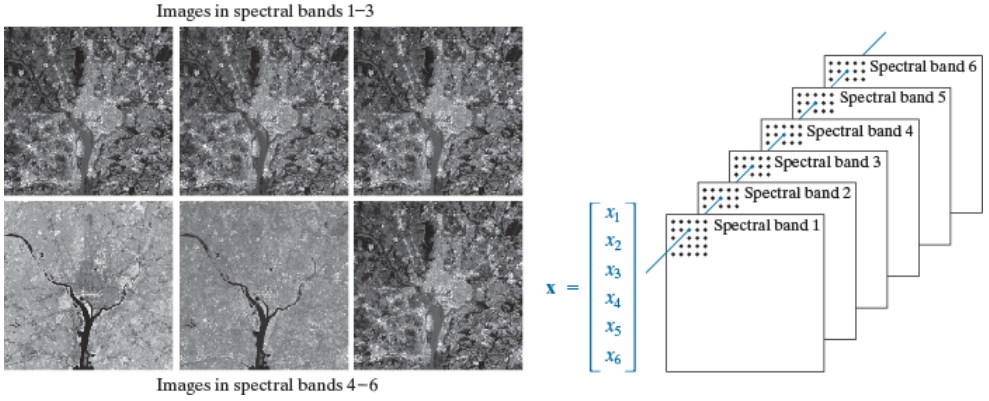
$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

x_1 = Petal width
x_2 = Petal length
x_3 = Sepal width
x_4 = Sepal length





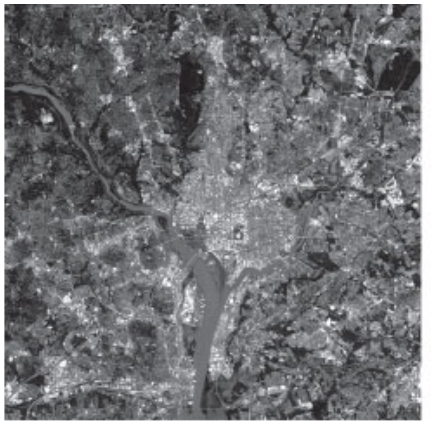
$$\mathbf{x} = \begin{bmatrix} g(r(\theta_1)) \\ g(r(\theta_2)) \\ \vdots \\ g(r(\theta_n)) \end{bmatrix}$$

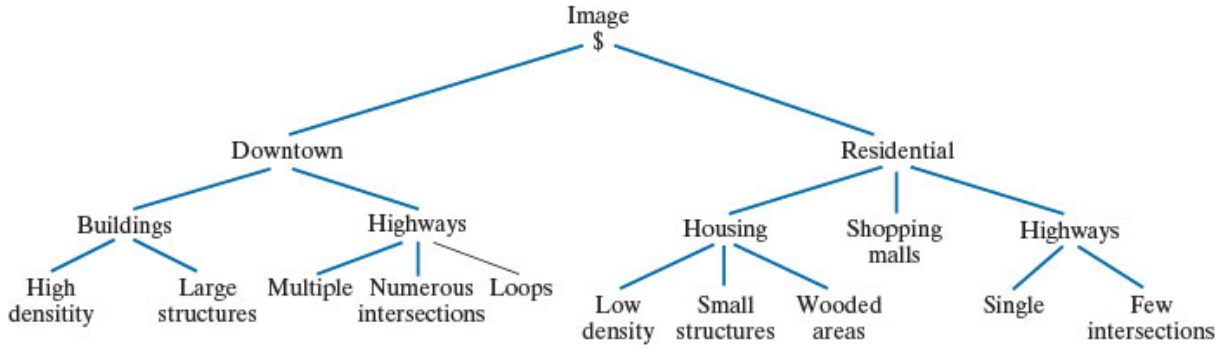


$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix}$$

结构模式（使用较少）

- 树形描述
  - 适于表示多层次排列的表达，可以是语义层概念的大小分层。





基于决策理论方法的识别

- 决策理论方法识别是以使用决策（或判别）函数为基础的。
- 令 $\mathbf{x}=(x_1,x_2,\cdots x_n)^{\mathrm{T}}$ 表示一个**n**维模式向量，对于**w**个模式类 $\omega_1,\omega_2,\cdots\omega_W$ ，决策理论模式识别的基本问题是依据如下属性来找到**W**个决策函数 $d_1(x),d_2(x),\cdots d_W(x)$ ：

  如果模式**x**属于类 ω_i ，则一定满足：

$$d_i(x)>d_j(x),\quad j=1,2,\cdots W;\ j\neq i$$

从类 ω_j 中分离出类 ω_i 的决策边界，由满足如下条件的**x**给出：$d_{ij}(x)=d_i(x)-d_j(x)=0$。$d_{ij}(x)$为两类之间的决策边界。

模式匹配

- 基于匹配的识别技术通过一个原型模式向量来表示每个类。根据一种预先定义的度量，将一个未知模式赋予最接近的类。最简单的方式是最小距离分类器。
- 若定义每个模式的原型定位为该类模式的平均向量：$m_j = \frac{1}{N_j}\sum_{x\in\omega_j} x_j$，使用欧氏距离 $D_j(x) = \|x - m_j\|$ 计算距离。
- 可以证明，选择最小距离等同于计算函数 $d_j(x) = x^T m_j - \frac{1}{2} m_j^T m_j, \quad j = 1,2,\cdots W$ 的最大值。

- 对于一个最小距离分类器，ω_i ω_j两类之间的决策边界为：

$$d_{ij}(x) = d_i(x) - d_j(x)$$

$$= x^T(m_i - m_j) - \frac{1}{2}(m_i - m_j)^T(m_i + m_j) = 0$$

- 该决策面是连接ω_i和ω_j的线段的垂直等分线。
- **n=2**时，垂直等分线是一条直线；**n=3**时是一个平面；**n>3**时，称其为一个超平面。

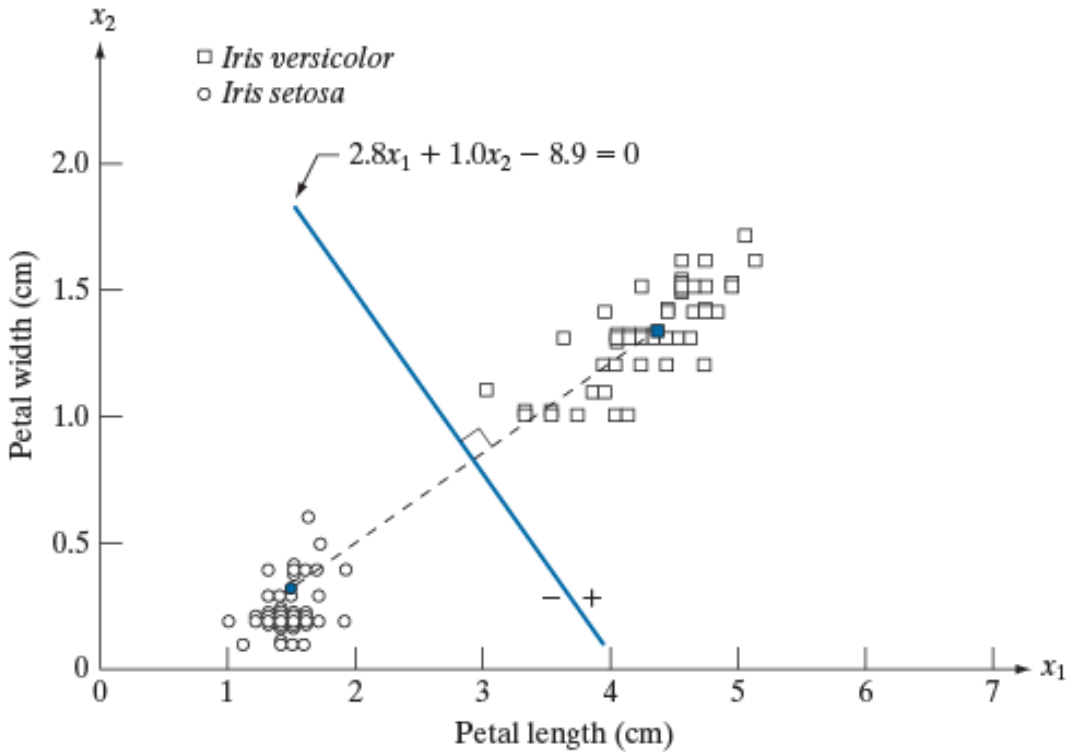
最小距离分类器示例

- 两个模式类均值分别为:

$$m_1 = (4.3, 1.3)^T;$$

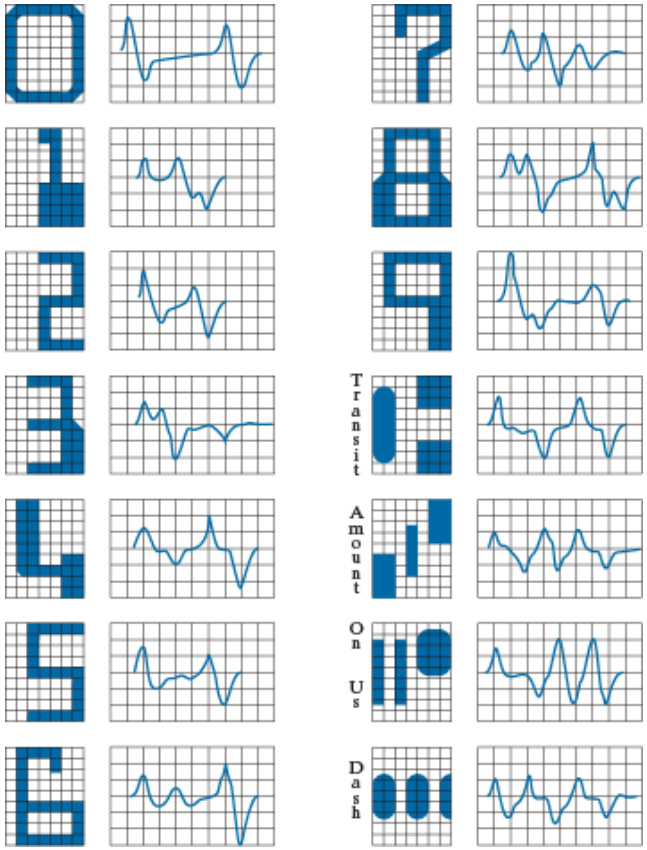
$$m_2 = (1.5, 0.3)^T$$

- 据此获得分类决策面方程。



- 当均值间的距离与每个类的分散度或者随机性相比较大时，最小距离分类器工作得很好。
- 可以证明：当每个类关于其均值的分布在**n**为模式空间中表现为一种球形的“超云团”形式时，最小距离分类器（在最小化错误分类的平均损失方面）会产生最佳性能。

- 实际中很少出现较大的均值分离和较小的散布，除非设计人员控制输入的性质，如美国银行家协会的**E-13B**字型的字符集。



最近邻分类法

- 最近邻分类法

$$d(X,\omega_j)=\min_{l=1,2,\cdots,S} d(X,Y_l) \qquad Y_l\in\omega_j$$

选距离最小的

$$d(X,\omega_i)<d(X,\omega_j) \qquad \forall j\neq i \Rightarrow X\in\omega_i$$

即将X判为与其最近的训练样本所属的类别

✓ 优点：简单，可分割大多数几何可分的类别 (各模式类几何意义上分开)。
✓ 缺点：要**存储和计算所有训练样本**，因只用了最近邻样本的信息，易受噪声影响，X的最近邻点为噪声时会误识。
✓ 改进方法：集群方法——分几个子集，用各子集的平均样本作最近邻法分类。
✓ K—近邻法——根据邻近的K个样本点中多数点的类别来分类。（投票）

- 非线性可分情况
  - 非线性判别函数
    - 多项式
      - 如二次判别函数的判决面是一个超二次曲面
    - 特殊函数展开
  - 广义线性判别函数
    - 在高维空间成为线性可分（核方法）

相关匹配

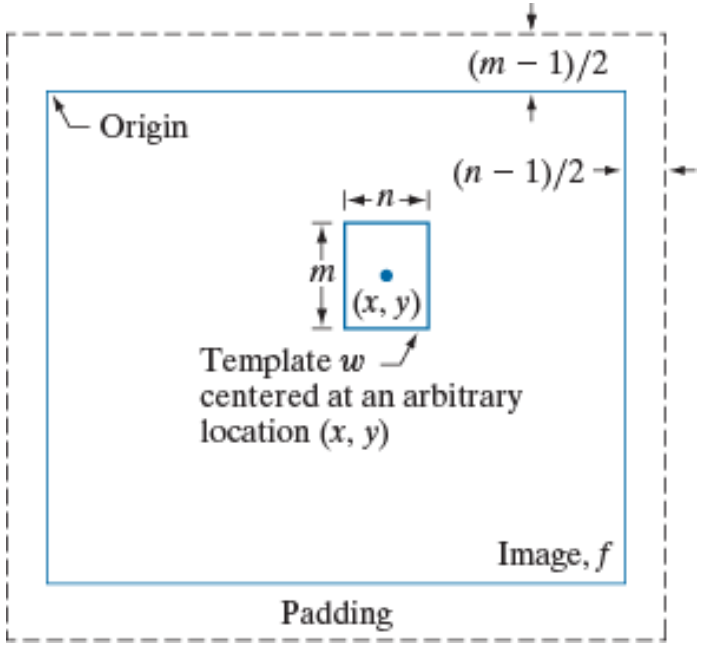
- 空间相关通过相关定理与函数的傅里叶变换相联系：

$$f(x,y) \circ w(x,y) \Leftrightarrow F^*(u,v)W(u,v)$$

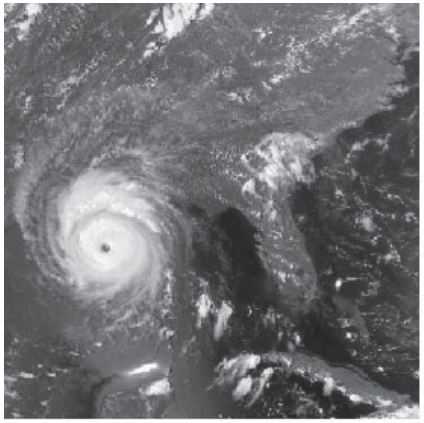
- 实用中采用如下的归一化相关系数：

$$r(x,y)=\frac{\sum_s\sum_t\left[w(s,t)-\bar{w}\right]\left[f(x+s,y+t)-\bar{f}_{xy}\right]}{\sqrt{\sum_s\sum_t\left[w(s,t)-\bar{w}\right]^2\sum_x\sum_y\left[f(x+s,y+t)-\bar{f}_{xy}\right]^2}}$$

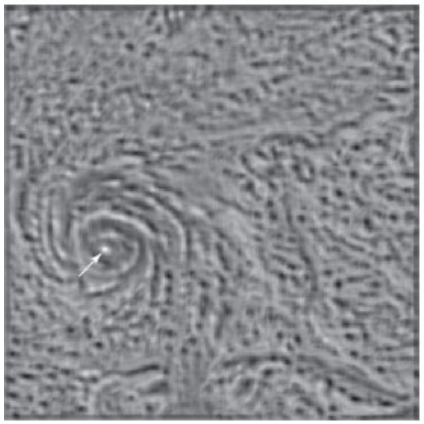
- 相关在此被称为模板匹配，最大相关即认为出现了模板匹配。



相关匹配示例



飓风眼模板的匹配定位



相关匹配目标检测的问题：

1. 计算量大；
2. 不具有旋转和尺度不变性；
3. 对形变、遮挡和光照变化等敏感。

最佳（贝叶斯）统计分类器

- 如果考虑模式类的统计特性,则可获统计平均意义上最佳的分类器。
- 有**W**类样本 $\omega_1, \omega_2, \cdots, \omega_W$，每类的先验概率为

$$P(\omega_i), i = 1, 2, \cdots, W$$

- 设将来自**k**类的样本判为**j**类导致的损失为 L_{kj} ，将模式**x**赋予类 **j** 的平均损失为：
- $$r_j(x) = \sum_{k=1}^{W} L_{kj} p(\omega_k | x)$$

- $r_j(x)$ 称为条件平均风险或者损失。

最佳统计分类器

- 对一随机矢量**X**，每类的条件概率密度为 $p(\mathbf{X}|\omega_i)$，后验概率为 $P(\omega_i|\mathbf{X})$
- **Bayes**公式：

$$P(\omega_i|\mathbf{X})=\frac{p(\mathbf{X}|\omega_i)P(\omega_i)}{p(\mathbf{X})}$$

- 平均损失为： $r_j(x)=\frac{1}{p(x)}\sum_{k=1}^{W}L_{kj}p(x|\omega_k)p(\omega_k)$
-
- 其中， $p(x|\omega_k)$ 是来自类 ω_k 的条件概率密度函数（似然概率），$p(\omega_k)$是类 ω_k 出现的概率（先验概率。）

最佳统计分类器

- $p(x)$ 为正，且对所有的 $r_j(x)$ 都一致，可以忽略，则平均损失表达式简化为：

$$r_j(x) = \sum_{k=1}^{W} L_{kj}\, p(x|\omega_k)\, p(\omega_k)$$

- 分类器有**W**个类，针对每个样本，计算判别为每个类的平均损失，并将样本对应的最低损失赋给相应的类，则关于决策的总体平均损失将是最低的。这种将总体平均损失降至最低的分类器称为贝叶斯分类器。即，若对所有的j且$j \neq i$, 有：

$$r_i(x) = \sum_{k=1}^{W} L_{ki}\, p(x|\omega_k)\, p(\omega_k) < \sum_{q=1}^{W} L_{qj}\, p(x|\omega_q)\, p(\omega_q)$$

- 则将**x**判别为类 ω_i

- 即：相对于把**x**判别成其他**j**类，判别成**i**类的总体平均损失最小。

简化

- 一般情况下，正确决策的损失通常赋值**0**，而不正确决策的损失通常被赋予相同的非零值，比如**1**。因此损失函数变为

- 因此：
$$L_{ij} = 1 - \delta_{ij}$$

- 判别函数变为：
$$r_j(x) = \sum_{k=1}^{W} (1 - \delta_{kj}) p(x|\omega_k) p(\omega_k) = p(x) - p(x|\omega_j) p(\omega_j)$$

- 或
$$p(x) - p(x|\omega_i) p(\omega_i) < p(x) - p(x|\omega_j) p(\omega_j), \quad \forall j \neq i$$

$$p(x|\omega_i) p(\omega_i) > p(x|\omega_j) p(\omega_j), \quad \forall j \neq i$$

最佳统计分类器

- 因此有决策函数：

$$d_j(x) = p(x|\omega_j)p(\omega_j)$$

  - 决策函数最大的类为样本所属的类别。
- 方法应用中的问题：
  - 必须知道每个类中模式的概率密度函数及每个类出现的概率；
  - 若所有样本等概率出现，则 $P(\omega_j) = 1/W$
  - 概率密度函数的估计相对困难。一般假设一个解析表达式，而且来自每个类的样本模式有一个必须的参数估计，比如高斯分布。

- 两边取对数:

$$\left[\log p\left(\mathbf{X}\mid\omega_i\right)+\log P(\omega_i)\right]>\left[\log p\left(\mathbf{X}\mid\omega_j\right)+\log P(\omega_j)\right]\Rightarrow \mathbf{X}\in\omega_i,\forall j\neq i$$

  - 这对于指数型分布，如正态分布，可简化运算
  - 可以证明，**Bayes分类器具有最小错误概率**

- 对于**d维正态分布**的一般形式：

$$p(X)=\frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma|^{\frac{1}{2}}}\exp\left(-\frac{(X-\mu)^T\Sigma^{-1}(X-\mu)}{2}\right)$$

记为 $p(X)\sim N(\mu,\Sigma)$ 其中 $\sum$ 是协方差矩阵

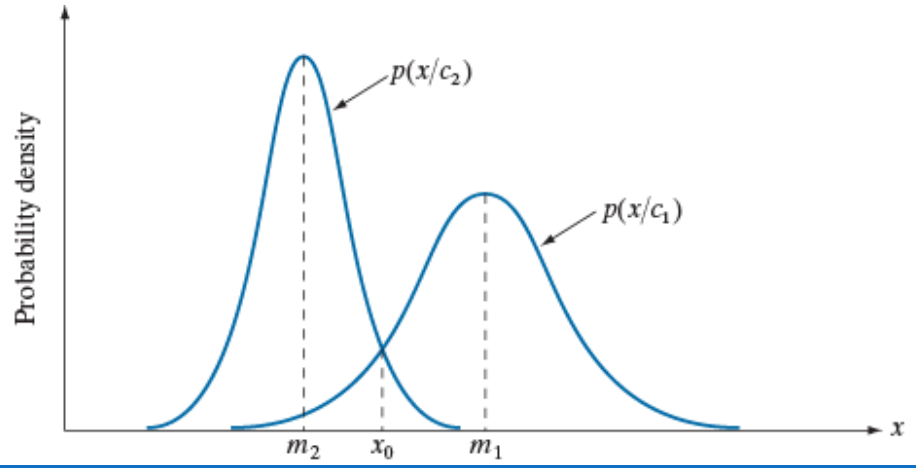
定义马氏距离 $\gamma^2=(X-\mu)^T\Sigma^{-1}(X-\mu)$

则 $$p(X)=\frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma|^{\frac{1}{2}}}\exp\left(-\frac{\gamma^2}{2}\right)$$

- 若总共两个类，他们的先验概率相同，

$$P(\omega_1) = P(\omega_2) = 1/2$$

- 则决策边界为满足　$p(x_0|\omega_1) = p(x_0|\omega_2)$　的 x_0
- 当两个类的先验概率不同时，若 ω_1出现的概率大，则 x_0左移，若 ω_2 出现的概率大，则 x_0 右移。



- 用对数形式的判决函数 $d_i(X)=\log p(X\mid\omega_i)+\log P(\omega_i)$

设 $p(X\mid\omega_i)\sim N(\mu_i,\Sigma_i)$

则 $d_i(X)=-\frac{d}{2}\log 2\pi-\frac{1}{2}\log|\Sigma_i|-\frac{1}{2}(X-\mu_i)^T\Sigma_i^{-1}(X-\mu_i)+\log P(\omega_i)$

可以省掉 $-\frac{d}{2}\log 2\pi$ 这一项，有：

$$d_i(X)=-\frac{1}{2}\log|\Sigma_i|-\frac{1}{2}(X-\mu_i)^T\Sigma_i^{-1}(X-\mu_i)+\log P(\omega_i)$$

是个超二次曲面（n维空间中的二次函数）

- 两类间的判决面

$$d_i(\mathbf{X}) = d_j(\mathbf{X})$$ ——超二次曲面方程

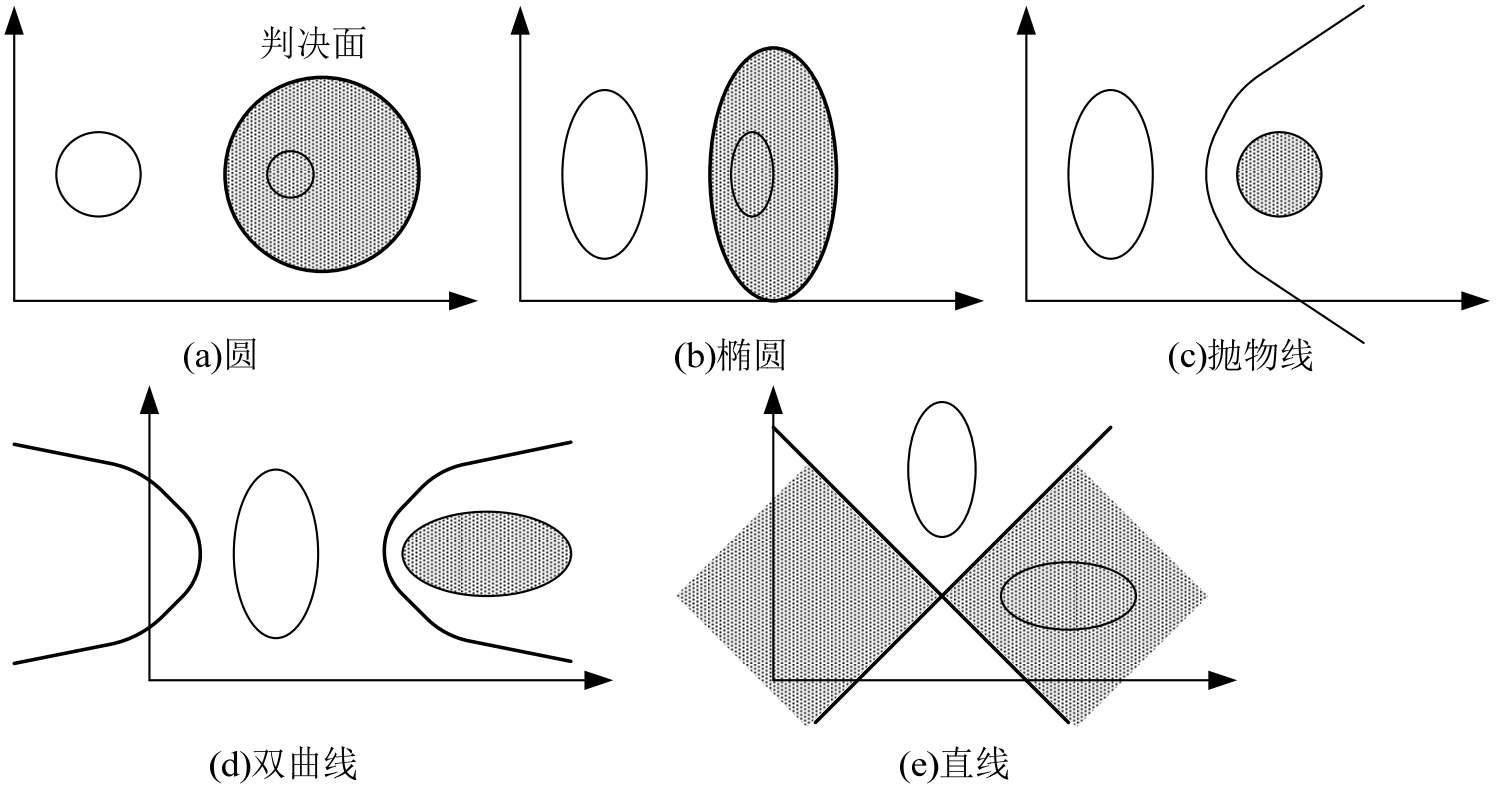
  - 若所有类的的协方差相等， $\Sigma_i = \Sigma$
  - 则判决函数 $d_i(\mathbf{X}) = -\frac{1}{2}(\mathbf{X}-\mu_i)^T\Sigma^{-1}(\mathbf{X}-\mu_i) + \log P(\omega_i)$
  - 展开上式，并去掉所有与类别*i* 无关的项，有：
  - $$d_i(\mathbf{X}) = \log P(\omega_i) + \mathbf{X}^T\Sigma^{-1}\mu_i - \frac{1}{2}\mu_i^T\Sigma^{-1}\mu_i$$

- 进一步，如果 $\Sigma = I$，而且各个类等概率出现，则有：

$$d_j(\mathbf{x}) = \mathbf{x}^T\mathbf{m}_j - \frac{1}{2}\mathbf{m}_j^T\mathbf{m}_j, \quad j = 1,2,\cdots W$$

- 为最小距离分类器的决策函数。
- 因此，如果（1）所有模式都是高斯的；(2)所有协方差阵都是单位阵；（3）所有类出现的概率相等，则：在贝叶斯意义上，最小距离分类器是最佳的。满足这些条件的高斯模式类是n维空间中外形相同的球状云团（超球面）。
- 最小距离分类器在每对类之间建立一个超平面，它是连接超球面中心的垂直平分线。

几种不同的判决面类型示意

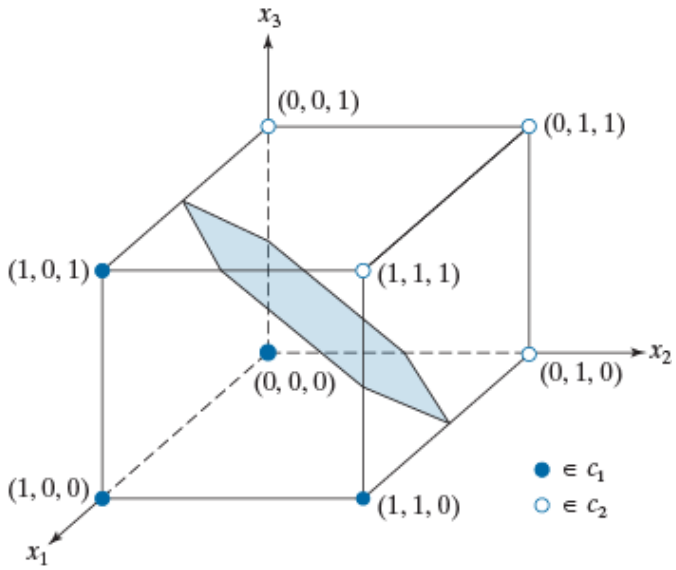


三维模式的分类器示例

均值：$\mathbf{m}_1 = \frac{1}{4}\begin{bmatrix}3\\1\\1\end{bmatrix}$ and $\mathbf{m}_2 = \frac{1}{4}\begin{bmatrix}1\\3\\3\end{bmatrix}$

方差：$\mathbf{C}_1 = \mathbf{C}_2 = \frac{1}{16}\begin{bmatrix}3 & 1 & 1\\1 & 3 & -1\\1 & -1 & 3\end{bmatrix}$

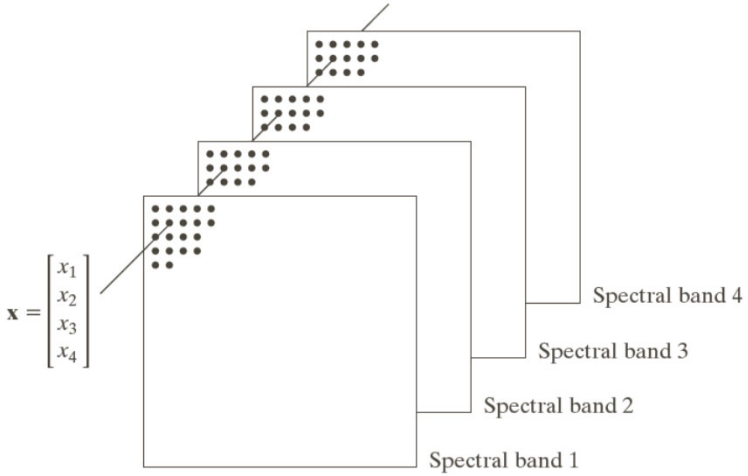
判决函数：$d_j(\mathbf{x}) = \mathbf{x}^T\mathbf{C}^{-1}\mathbf{m}_j - \frac{1}{2}\mathbf{m}_j^T\mathbf{C}^{-1}\mathbf{m}_j$

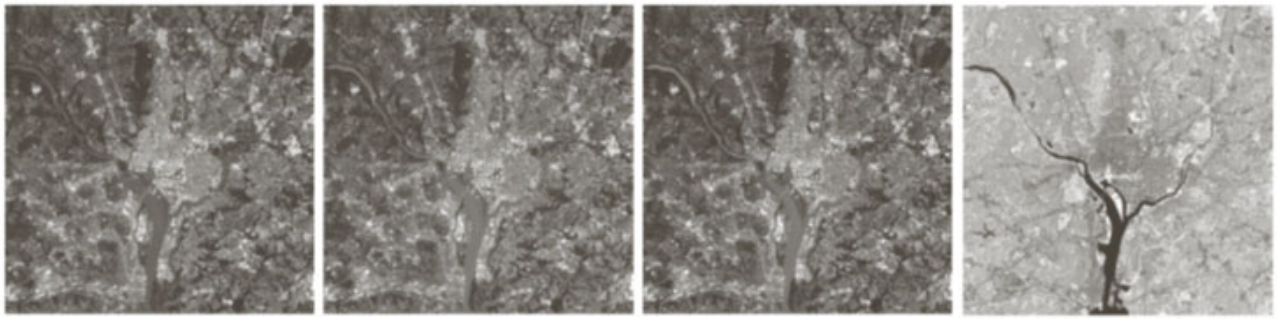


分类决策面：$d_1(\mathbf{x}) - d_2(\mathbf{x}) = 8x_1 - 8x_2 - 8x_3 + 4 = 0$

多光谱图像分类示例

- 四个波段，构造四维模式向量后，用贝叶斯分类器分类





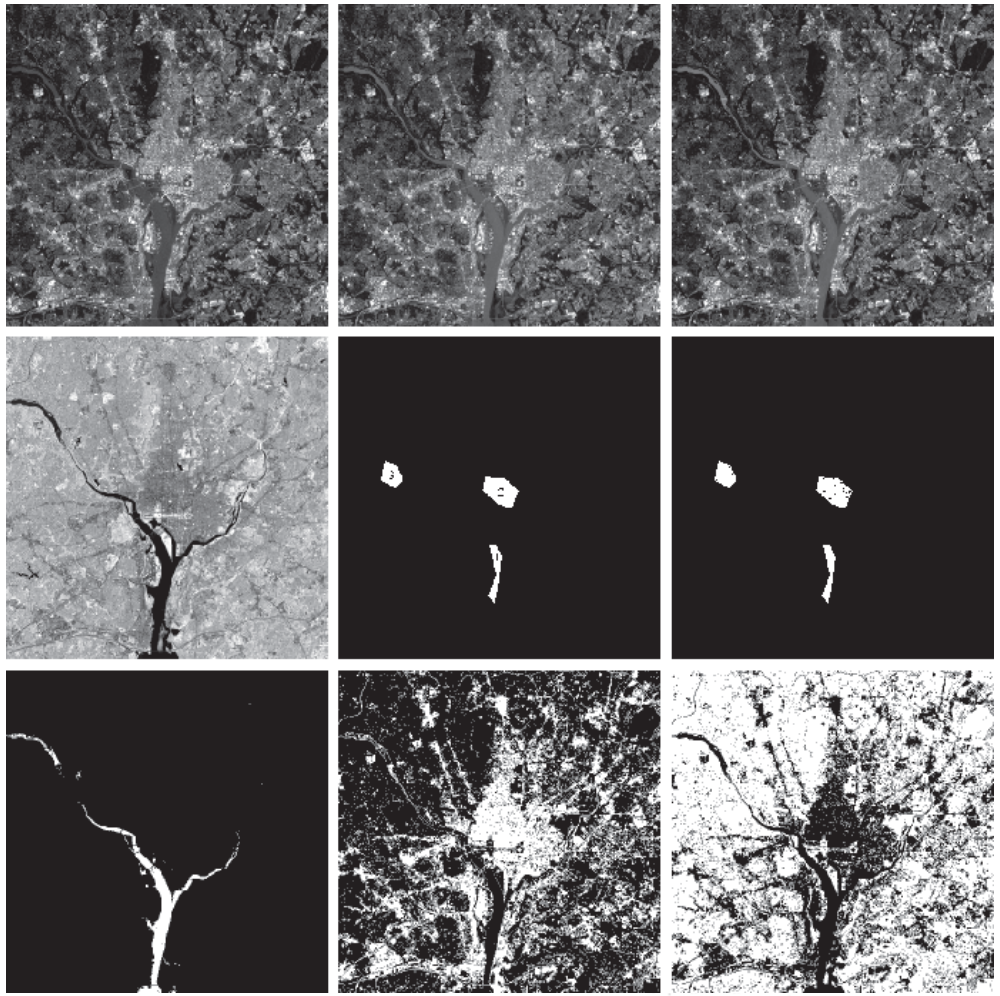
多光谱图像分类示例

TABLE 12.1
Bayes classification of multispectral image data. Classes 1, 2, and 3 are water, urban, and vegetation, respectively.

Training Patterns						Test Patterns					
Class	No. of Samples	Classified into Class			% Correct	Class	No. of Samples	Classified into Class			% Correct
		1	2	3				1	2	3	
1	484	482	2	0	99.6	1	483	478	3	2	98.9
2	933	0	885	48	94.9	2	932	0	880	52	94.4
3	483	0	19	464	96.1	3	482	0	16	466	96.7

$$d_i(X) = -\frac{1}{2}\log|\Sigma_i| - \frac{1}{2}(X-\mu_i)^T\Sigma_i^{-1}(X-\mu_i) + \log P(\omega_i)$$

- 小样本，生成式模型



a b c
d e f
g h i

图 12.13 多光谱数据的贝叶斯分类：(a)~(d)可见蓝光、可见绿光、可见红光和近红外波长图像；(e)显示(1)水体、(2)市区和(3)植被的样本区域的模板；(f)分类结果。黑点表示未正确分类的点，其他(白)点是正确分类的点；(g)分类为水体的所有图像像素(白色)；(h)分类为市区的所有图像像素(白色)；(i)分类为植被的所有图像像素(白色)

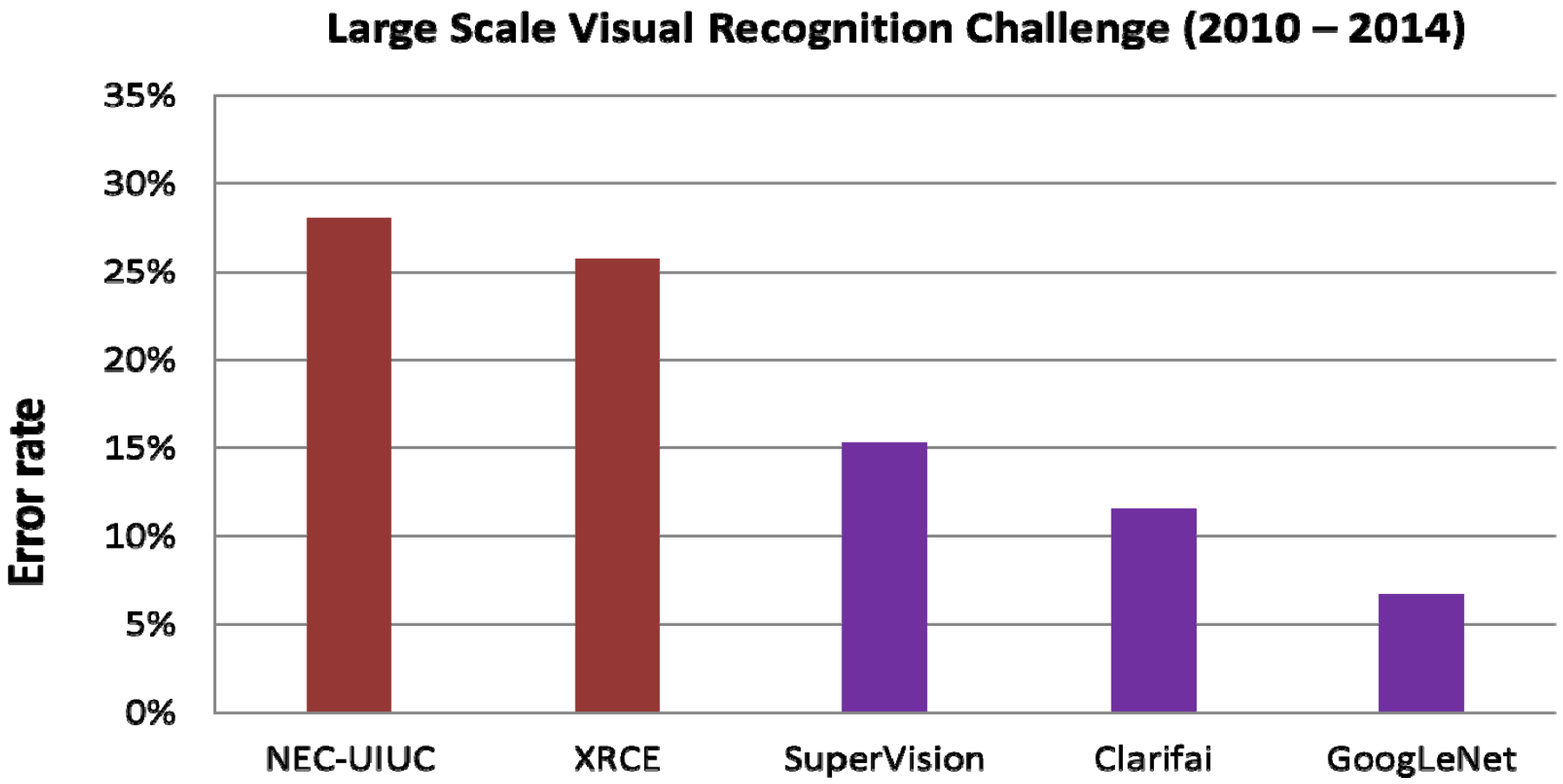
图像处理中逐像素分类与分割的关系

- 逐像素的分类实际上就是一个分割问题，把一幅图像分成两个或者多个可能的区域；
- 在图像分割中的阈值处理，可以视为一个贝叶斯分类问题。它最佳地将模式赋给两个或多个类；
- 但最佳性要求已知每个类的概率密度函数和先验概率。但在实际中不容易获得（通常假设为高斯密度）。则在分割中所能实现的最佳程度，与假设接近真实情况的程度成正比。
- 事实上，多数情况下几乎无法获得真实的概率分布。对抗生成网络（GAN）的一大作用就是依靠对抗学习，来学习获得样本的分布。

传统模式分类方式的缺陷

- 需要人工设计特征表达方式，带来很多问题：
  - 需要领域知识；
  - 设计要求很高（尺度、光照等不变性）；
  - 泛化能力不足（很难表达所有情况）；
  - 很难利用高层特征（很多时候人类自己也说不清楚靠什么特征分类）
  - 非常耗时费力
  - ……
- 神经网络能学习特征表达，带来的计算机视觉的性能飞跃。

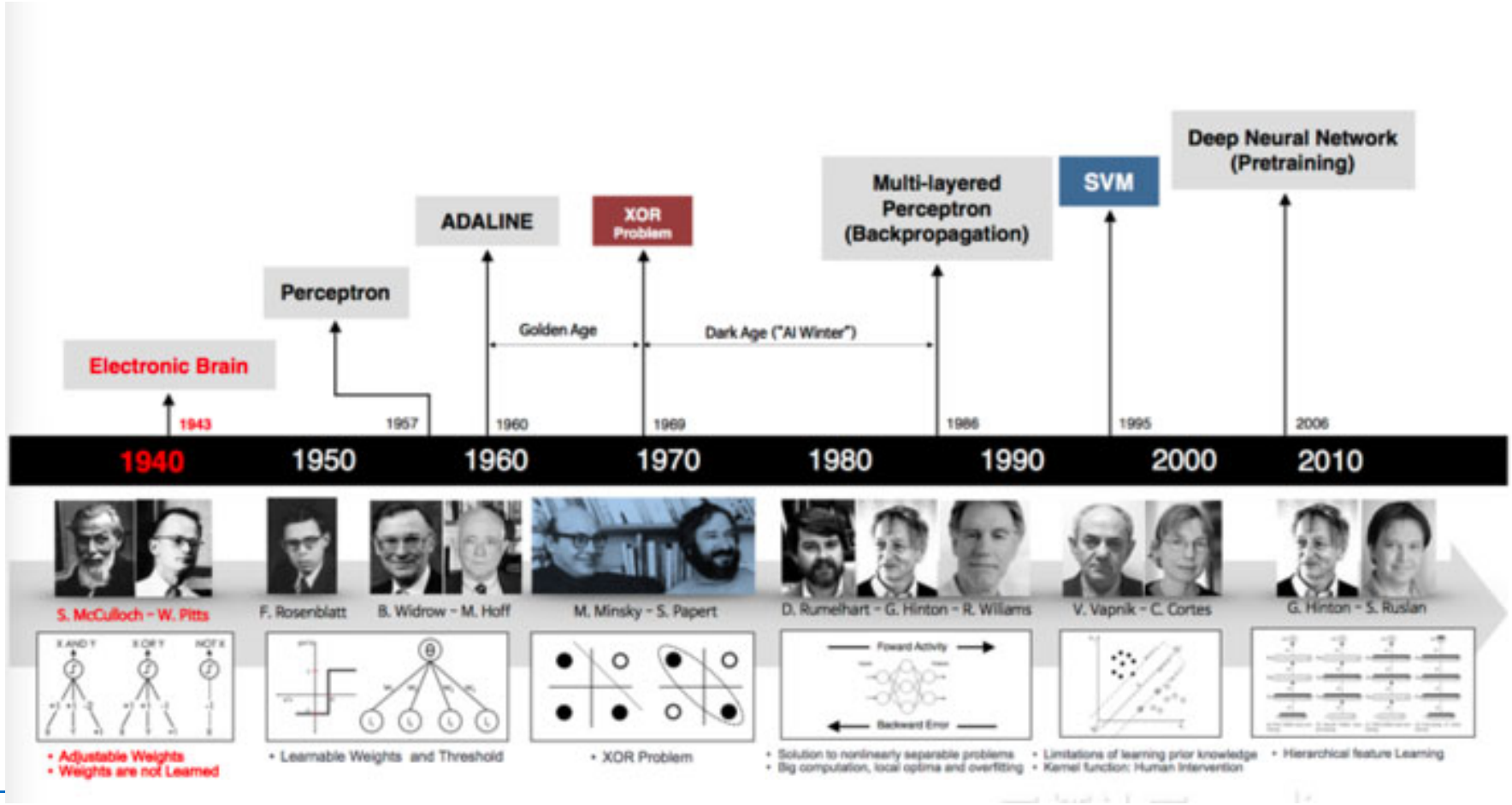
神经网络


Large Scale Visual Recognition Challenge (2010 – 2014)
Error rate
35%
30%
25%
20%
15%
10%
5%
0%
NEC-UIUC
XRCE
SuperVision
Clarifai
GoogLeNet

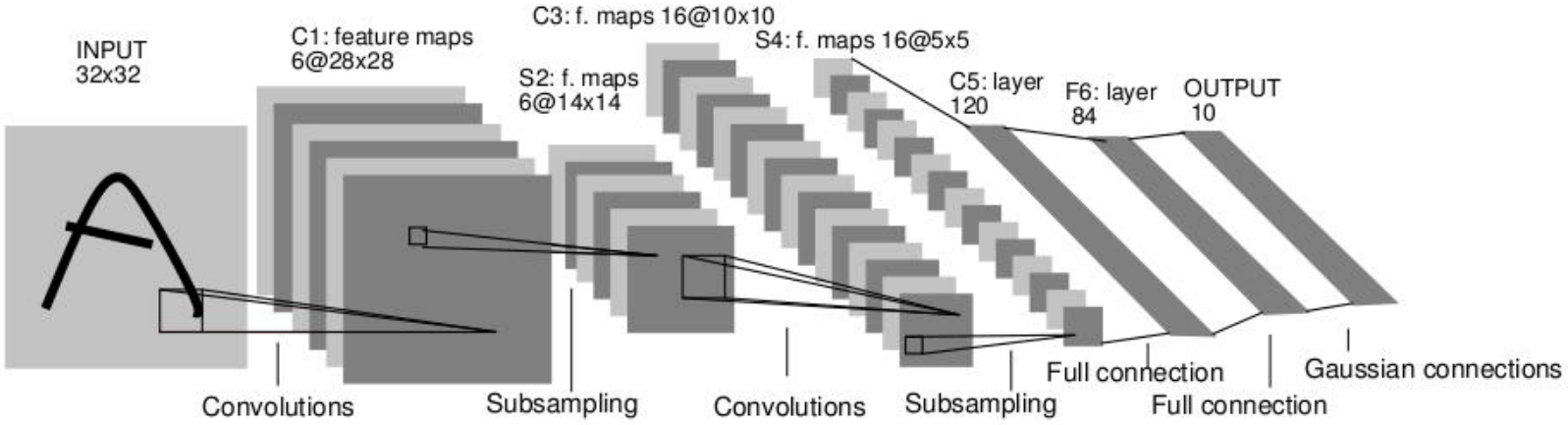
人工神经网络

- 人工神经网络（Artificial Neural Network，ANN）简称神经网络(NN)，是基于生物学中神经网络的基本原理，在理解和抽象了人脑结构和外界刺激响应机制后，以网络拓扑知识为理论基础，模拟人脑的神经系统对复杂信息的处理机制的一种数学模型。
- 该模型以并行分布的处理能力、高容错性、智能化和自学习等能力为特征，将信息的加工和存储结合在一起，以其独特的知识表示方式和智能化的自适应学习能力，引起各学科领域的关注。
- 它实际上是一个有大量简单元件相互连接而成的复杂网络，具有高度的非线性，能够进行复杂的逻辑操作和非线性关系实现的系统。

神经网络的发展历史


Deep Neural Network
(Pretraining)
SVM
Multi-layered
Perceptron
(Backpropagation)
XOR
Problem
ADALINE
Perceptron
Electronic Brain
Golden Age
Dark Age ("AI Winter")
1943
1957
1960
1969
1986
1995
2006
1940
1950
1960
1970
1980
1990
2000
2010
S. McCulloch - W. Pitts
F. Rosenblatt
B. Widrow - M. Hoff
M. Minsky - S. Papert
D. Rumelhart - G. Hinton - R. Williams
V. Vapnik - C. Cortes
G. Hinton - S. Ruslan
• Adjustable Weights
• Weights are not Learned
• Learnable Weights and Threshold
• XOR Problem
Foward Activity
Backward Error
• Solution to nonlinearly separable problems
• Big computation, local optima and overfitting
• Limitations of learning prior knowledge
• Kernel function: Human Intervention
• Hierarchical feature Learning

卷积网络例子：LeNet

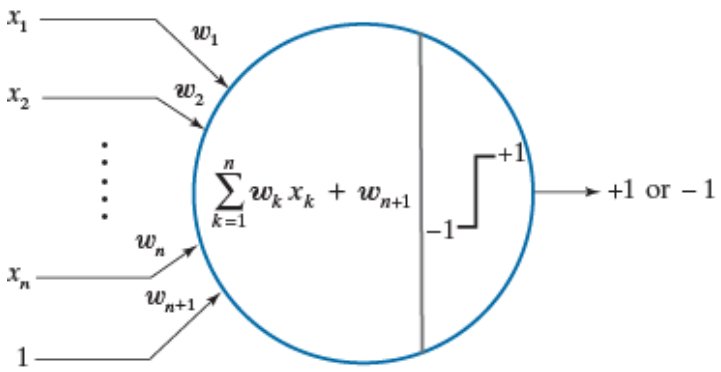


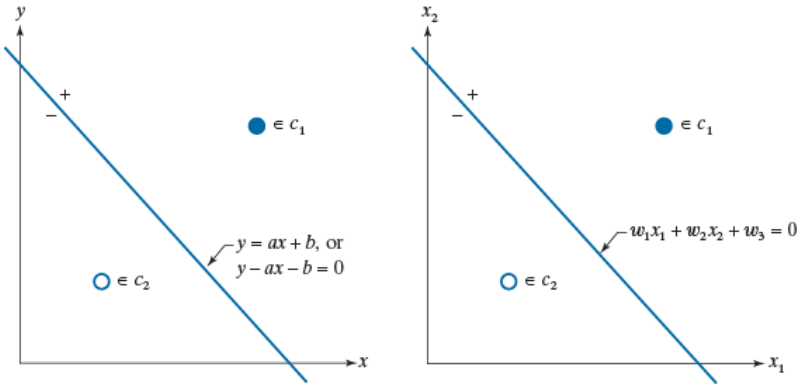
The first convolutional neural network
applied on hand written digits recognition

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, *Gradient-based learning applied to document recognition*, Proceedings of the IEEE 86(11): 2278–2324, 1998.

感知机

- 单个感知机学习两个线性可分离模式类之间的线性边界



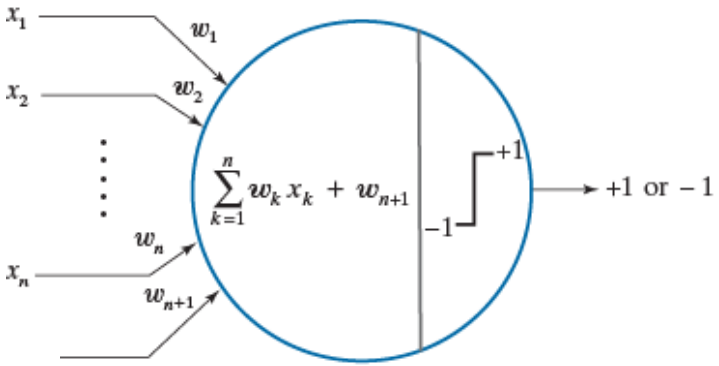


$$\mathbf{w}^T\mathbf{x} + w_{n+1} = \begin{cases} > 0 & \text{if } \mathbf{x} \in c_1 \\ < 0 & \text{if } \mathbf{x} \in c_2 \end{cases}$$

单个感知机训练算法

define $\mathbf{x} \triangleq [x_1, x_2, \ldots, x_n, 1]^T$ and $\boldsymbol{w} \triangleq [w_1, w_2, \ldots, w_n, w_{n+1}]^T$

则有：
$$\boldsymbol{w}^T\mathbf{x} = \begin{cases} > 0 & \text{if } \mathbf{x} \in c_1 \\ < 0 & \text{if } \mathbf{x} \in c_2 \end{cases}$$



训练算法如下：

1′) If $\mathbf{x}(k) \in c_1$ and $\boldsymbol{w}^T(k)\mathbf{x}(k) \le 0$, let

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) + \alpha\mathbf{x}(k)$$

2′) If $\mathbf{x}(k) \in c_2$ and $\boldsymbol{w}^T(k)\mathbf{x}(k) \ge 0$, let

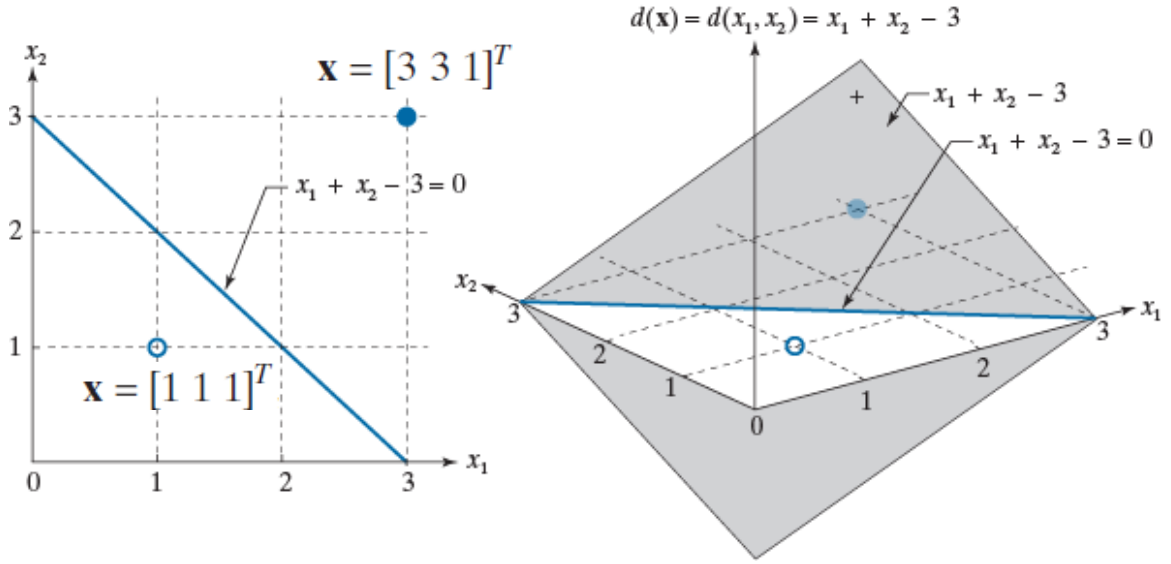
$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) - \alpha\mathbf{x}(k)$$

3′) Otherwise, let

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k)$$

对两个样本分类的感知机例子

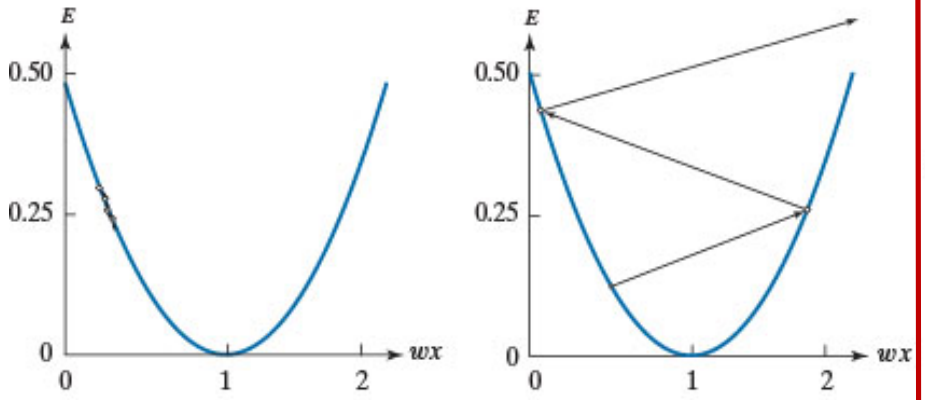
用单个感知机训练
算法得到的决策面

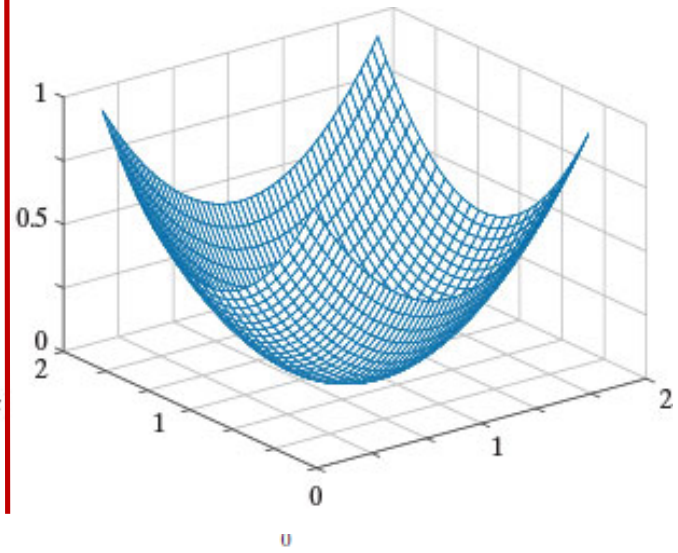


最小均方误差（LMSE）算法

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) - \alpha\left[\frac{\partial E(\boldsymbol{w})}{\partial \boldsymbol{w}}\right]_{\boldsymbol{w}=\boldsymbol{w}(k)}$$

$$E(\boldsymbol{w}) = \frac{1}{2}\left(r - \boldsymbol{w}^T\mathbf{x}\right)^2$$



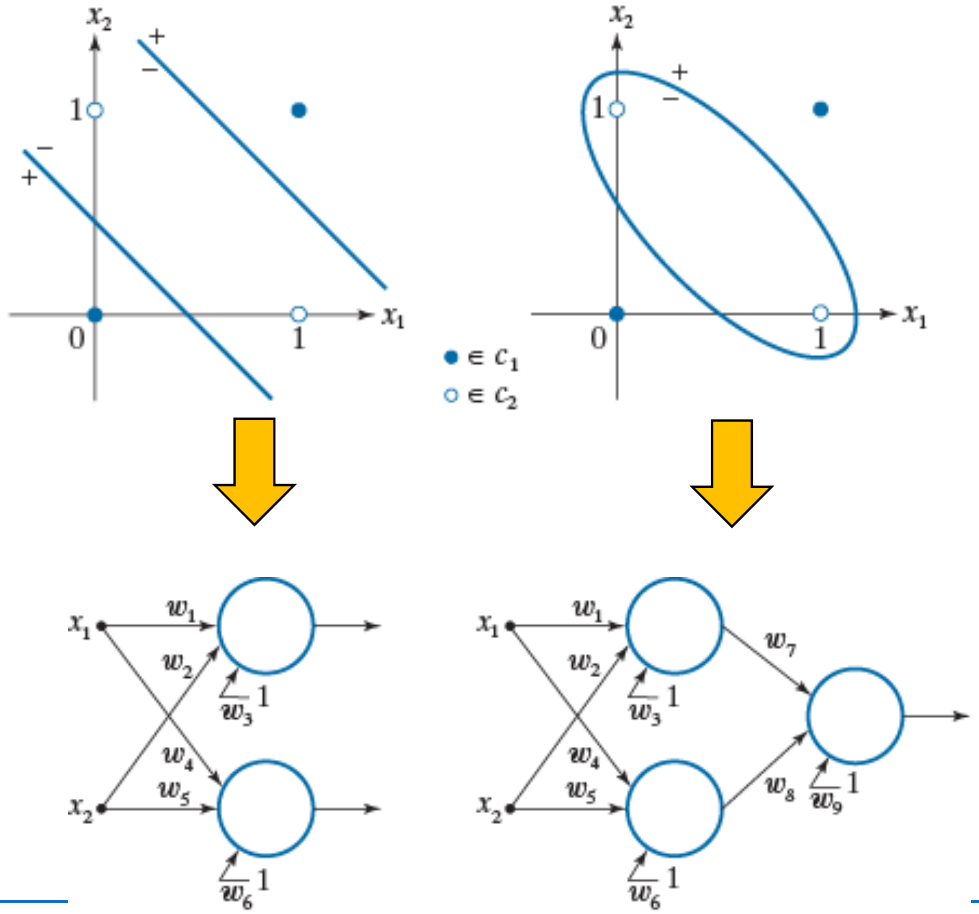


a b c

FIGURE 12.25 Plots of E as a function of wx for $r = 1$. (a) A value of α that is too small can slow down convergence. (b) If α is too large, large oscillations or divergence may occur. (c) Shape of the error function in 2-D.

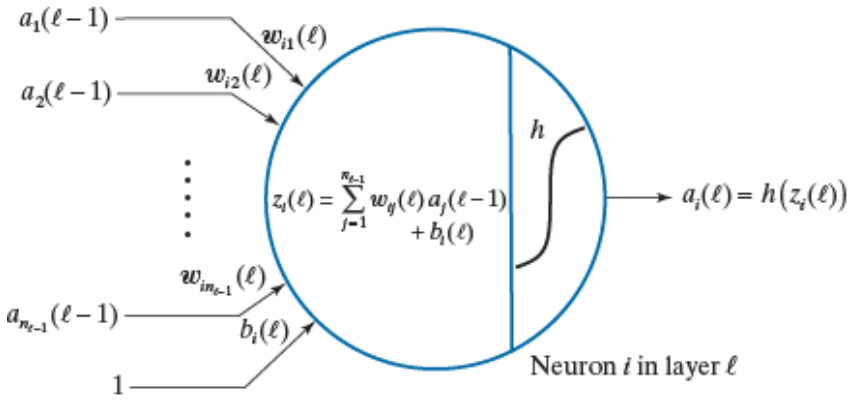
单个感知机的局限（XOR问题）

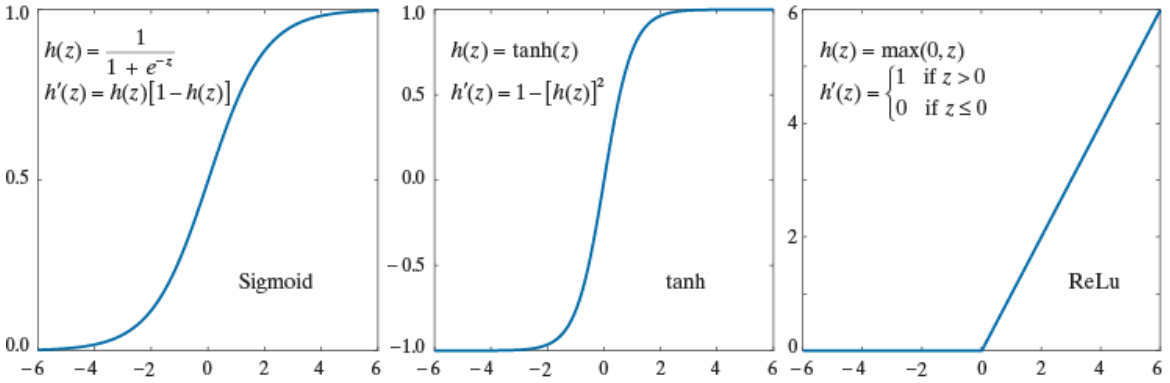
A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0



多层前馈神经网络

- 感知机用硬阈值处理函数来执行分类，在0附近波动很大。
- 这种对小信号的敏感性会在这些单元的互联系统中导致严重的稳定性问题，使其不适合于分层的结构。
- 解决方案是采用一个平滑函数(如sigmoid)代替硬限制器。





a b c
FIGURE 12.30 Various activation functions. (a) Sigmoid. (b) Hyperbolic tangent (also has a sigmoid shape, but it is centered about 0 in both dimensions). (c) Rectifier linear unit (ReLU).

全连接神经网络

层 l 中神经元 i 的总输入：

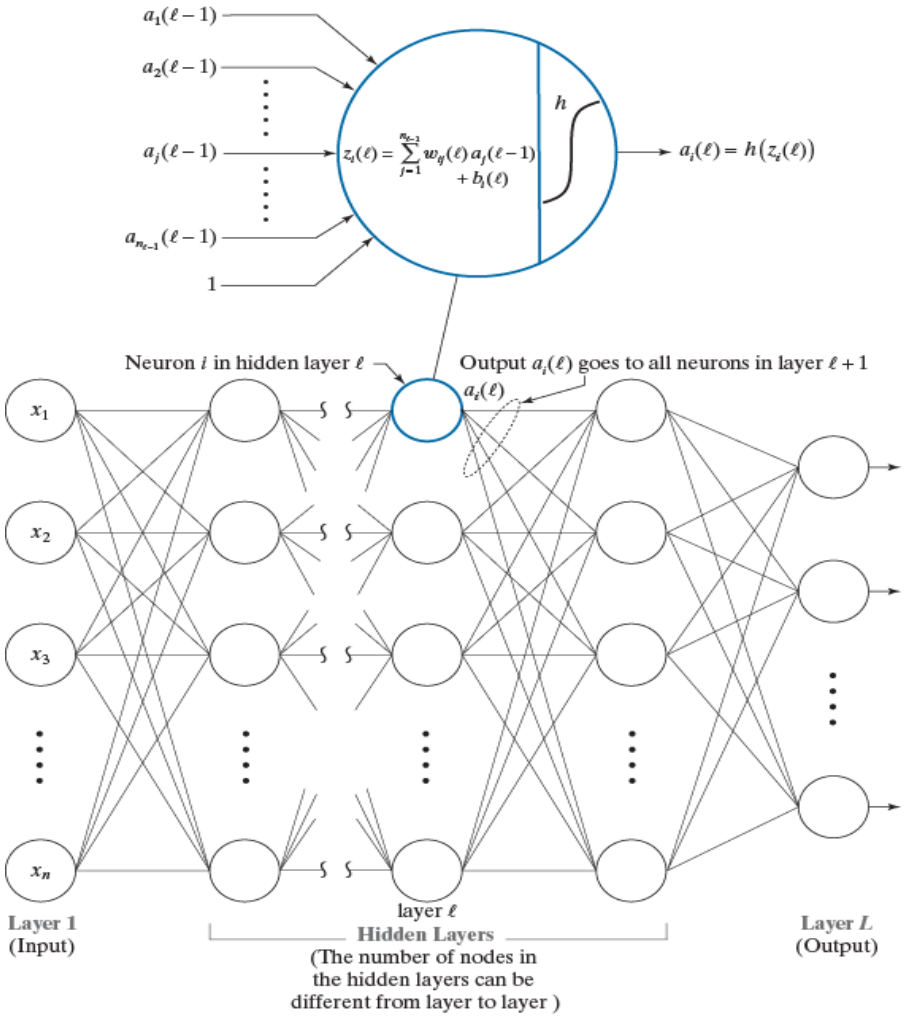
$$z_i(\ell) = \sum_{j=1}^{n_{\ell-1}} w_{ij}(\ell)\, a_j(\ell-1) + b_i(\ell)$$

层 l 中神经元 i 的输出（激活值）为：

$$a_i(\ell) = h\big(z_i(\ell)\big) \qquad i = 1, 2, \ldots, n_\ell$$

网络输出节点神经元 i 的值为：

$$a_i(L) = h\big(z_i(L)\big) \qquad i = 1, 2, \ldots, n_L$$



全连接网络示例

$$z_1(2) = \sum_{j=1}^{3} w_{1j}(2)a_j(1) + b_1(2) = (0.1)(3) + (0.2)(0) + (0.6)(1) + 0.4 = 1.3$$

$$a_1(2) = h\big(z_1(2)\big) = \frac{1}{1+e^{-1.3}} = 0.7858$$

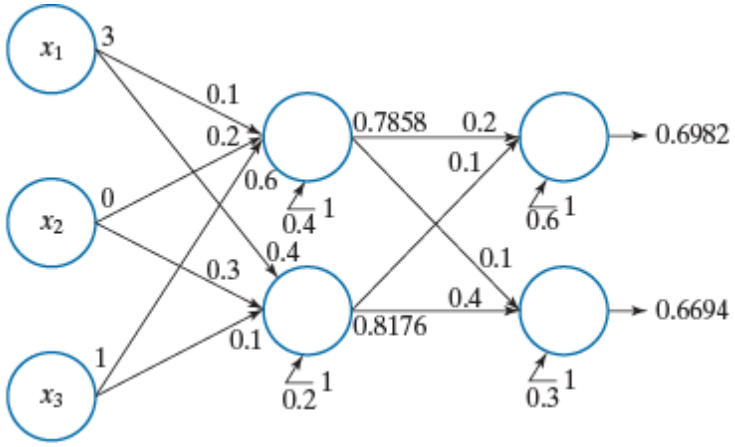
$$z_2(2) = \sum_{j=1}^{3} w_{2j}(2)a_j(1) + b_2(2) = (0.4)(3) + (0.3)(0) + (0.1)(1) + 0.2 = 1.5$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 3 \\ 0 \\ 1 \end{bmatrix}$$

$$a_2(2) = h\big(z_2(2)\big) = \frac{1}{1+e^{-1.5}} = 0.8176$$

$$z_2(3) = \sum_{j=1}^{2} w_{2j}(3)a_j(2) + b_2(3) = (0.1)(0.7858) + (0.4)(0.8176) + 0.3 = 0.7056$$

$$a_2(3) = h\big(z_2(2)\big) = \frac{1}{1+e^{-0.7056}} = 0.6694$$



矩阵公式表达

■ 层 l 的权重：$\mathbf{W}(\ell)=\begin{bmatrix} w_{11}(\ell) & w_{12}(\ell) & \cdots & w_{1n_{\ell-1}}(\ell) \\ w_{21}(\ell) & w_{22}(\ell) & \cdots & w_{2n_{\ell-1}}(\ell) \\ \vdots & \vdots & \cdots & \\ w_{n_\ell 1}(\ell) & w_{n_\ell 2}(\ell) & \cdots & w_{n_\ell n_{\ell-1}}(\ell) \end{bmatrix}$

层 l 的总输入：$\mathbf{z}(\ell)=\mathbf{W}(\ell)\mathbf{a}(\ell-1)+\mathbf{b}(\ell) \qquad \ell=2,3,\ldots,L$

层 l 的输出：$\mathbf{a}(\ell)=h[\mathbf{z}(\ell)]=\begin{bmatrix} h(z_1(\ell)) \\ h(z_2(\ell)) \\ \vdots \\ h(z_{n_\ell}(\ell)) \end{bmatrix}$

■ 同时输入多个模式向量时的正向传播计算步骤（X为 $n\times n_p$ 维，共n_p个模式向量）：

Step	Description	Equations
Step 1	Input patterns	$\mathbf{A}(1)=\mathbf{X}$
Step 2	Feedforward	For $\ell=2,\ldots,L$, compute $\mathbf{Z}(\ell)=\mathbf{W}(\ell)\mathbf{A}(\ell-1)+\mathbf{B}(\ell)$ and $\mathbf{A}(\ell)=h(\mathbf{Z}(\ell))$
Step 3	Output	$\mathbf{A}(L)=h(\mathbf{Z}(L))$

矩阵计算，提高效率！！

反向传播训练（BP）

- 反向传播训练的4个步骤：

1. 输入模式向量；
2. 正向传播网络，对训练集所有模式进行分类并确定分类误差；
3. 反向传播，向网络反向传播误差，计算更新参数所需的变化；
4. 更新网络中的权重和偏置。重复1-4，直到误差达到可接受的水平。

对于输出层，有：

$$E=\sum_{j=1}^{n_L}E_j=\frac{1}{2}\sum_{j=1}^{n_L}\left(r_j-a_j(L)\right)^2$$
$$=\frac{1}{2}\|\mathbf{r}-\mathbf{a}(L)\|^2$$

$$\delta_j(L)=\frac{\partial E}{\partial z_j(L)}=\frac{\partial E}{\partial a_j(L)}\frac{\partial a_j(L)}{\partial z_j(L)}=\frac{\partial E}{\partial a_j(L)}\frac{\partial h\left(z_j(L)\right)}{\partial z_j(L)}$$
$$=\frac{\partial E}{\partial a_j(L)}h'\left(z_j(L)\right)$$

$$\delta_j(L)=h\left(z_j(L)\right)\left[1-h\left(z_j(L)\right)\right]\left[a_j(L)-r_j\right]$$

反向传播训练

对于隐含层，有：

$$\begin{aligned}\delta_j(\ell) &= \frac{\partial E}{\partial z_j(\ell)} = \sum_i \frac{\partial E}{\partial z_i(\ell+1)} \frac{\partial z_i(\ell+1)}{\partial a_j(\ell)} \frac{\partial a_j(\ell)}{\partial z_j(\ell)} \\ &= \sum_i \delta_i(\ell+1) \frac{\partial z_i(\ell+1)}{\partial a_j(\ell)} h'\left(z_j(\ell)\right) \\ &= h'\left(z_j(\ell)\right) \sum_i w_{ij}(\ell+1)\delta_i(\ell+1)\end{aligned}$$

根据链式规则，得到权重的梯度为：

$$\begin{aligned}\frac{\partial E}{\partial w_{ij}(\ell)} &= \frac{\partial E}{\partial z_i(\ell)} \frac{\partial z_i(\ell)}{\partial w_{ij}(\ell)} \\ &= \delta_i(\ell) \frac{\partial z_i(\ell)}{\partial w_{ij}(\ell)} \\ &= a_j(\ell-1)\boldsymbol{\delta}_i(\ell)\end{aligned} \qquad \frac{\partial E}{\partial b_i(\ell)} = \delta_i(\ell)$$

根据梯度下降法更新参数：

$$\begin{aligned}w_{ij}(\ell) &= w_{ij}(\ell) - \alpha \frac{\partial E(\ell)}{\partial w_{ij}(\ell)} \\ &= w_{ij}(\ell) - \alpha \delta_i(\ell) a_j(\ell-1)\end{aligned} \qquad \begin{aligned}b_i(\ell) &= b_i(\ell) - \alpha \frac{\partial E}{\partial b_i(\ell)} \\ &= b_i(\ell) - \alpha \delta_i(\ell)\end{aligned}$$

反向传播矩阵公式

对于输出层的所有神经元，有：

$$\boldsymbol{\delta}(L)=\begin{bmatrix}\delta_1(L)\\ \delta_2(L)\\ \vdots\\ \delta_{n_L}(L)\end{bmatrix}=\begin{bmatrix}\frac{\partial E}{\partial a_1(L)}h'(z_1(L))\\ \frac{\partial E}{\partial a_2(L)}h'(z_2(L))\\ \vdots\\ \frac{\partial E}{\partial a_{n_L}(L)}h'(z_{n_L}(L))\end{bmatrix}=\begin{bmatrix}\frac{\partial \boldsymbol{E}}{\partial a_1(L)}\\ \frac{\partial E}{\partial a_2(L)}\\ \vdots\\ \frac{\partial E}{\partial a_{n_L}(L)}\end{bmatrix}\odot\begin{bmatrix}h'(z_1(L))\\ h'(z_2(L))\\ \vdots\\ h'(z_{n_L}(L))\end{bmatrix}=\frac{\partial E}{\partial \mathbf{a}(L)}\odot h'(\mathbf{z}(L))=(\mathbf{a}(L)-\mathbf{r})\odot h'(\mathbf{z}(L))$$

考虑所有输入模式（样本），对于输出神经元有：　$\mathbf{D}(L)=(\mathbf{A}(L)-\mathbf{R})\odot h'(\mathbf{Z}(L))$

考虑所有输入模式（样本），对于隐含层神经元有：$\mathbf{D}(\ell)=\left(\mathbf{W}^T(\ell+1)\mathbf{D}(\ell+1)\right)\odot h'(\mathbf{Z}(\ell))$

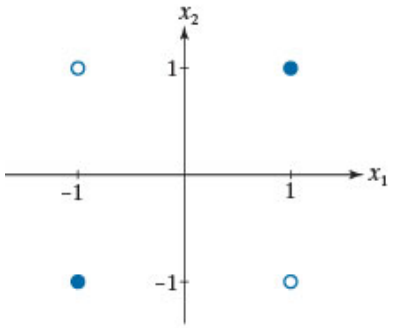
反向传播矩阵公式

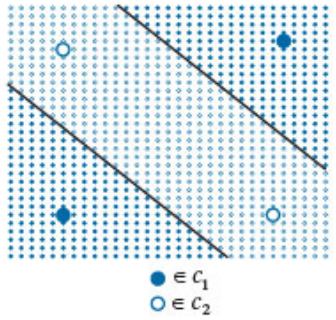
TABLE 13.3
Matrix formulation for training a feedforward, fully connected multilayer neural network using backpropagation. Steps 1–4 are for one epoch of training. $\mathbf{X}$, $\mathbf{R}$, and the learning rate parameter α, are provided to the network for training. The network is initialized by specifying weights, $\mathbf{W}(1)$, and biases, $\mathbf{B}(1)$, as small random numbers.

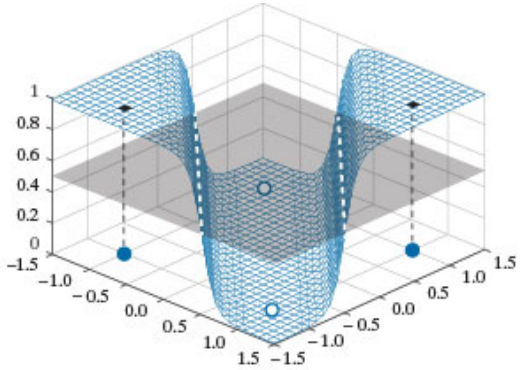
Step	Description	Equations
Step 1	Input patterns	$\mathbf{A}(1) = \mathbf{X}$
Step 2	Forward pass	For $\ell = 2, \ldots, L$, compute: $\mathbf{Z}(\ell) = \mathbf{W}(\ell)\mathbf{A}(\ell-1) + \mathbf{B}(\ell)$; $\mathbf{A}(\ell) = h(\mathbf{Z}(\ell))$; $h'(\mathbf{Z}(\ell))$; and $\mathbf{D}(L) = (\mathbf{A}(L) - \mathbf{R}) \odot h'(\mathbf{Z}(L))$
Step 3	Backpropagation	For $\ell = L-1, L-2, \ldots, 2$, compute $\mathbf{D}(\ell) = \left(\mathbf{W}^T(\ell+1)\mathbf{D}(\ell+1)\right) \odot h'(\mathbf{Z}(\ell))$
Step 4	Update weights and biases	For $\ell = 2, \ldots, L$, let $\mathbf{W}(\ell) = \mathbf{W}(\ell) - \alpha\mathbf{D}(\ell)\mathbf{A}^T(\ell-1)$, $\mathbf{b}(\ell) = \mathbf{b}(\ell) - \alpha\sum_{k=1}^{n_p}\boldsymbol{\delta}_k(\ell)$, and $\mathbf{B}(\ell) = \underset{n_p \text{ times}}{\text{concatenate}}\{\mathbf{b}(\ell)\}$, where the $\boldsymbol{\delta}_k(\ell)$ are the columns of $\mathbf{D}(\ell)$

全连接神经网络求解异或问题

$$\mathbf{X}=\begin{bmatrix}1 & -1 & -1 & 1\\ 1 & -1 & 1 & -1\end{bmatrix};\ \mathbf{R}=\begin{bmatrix}1 & 1 & 0 & 0\\ 0 & 0 & 1 & 1\end{bmatrix}$$

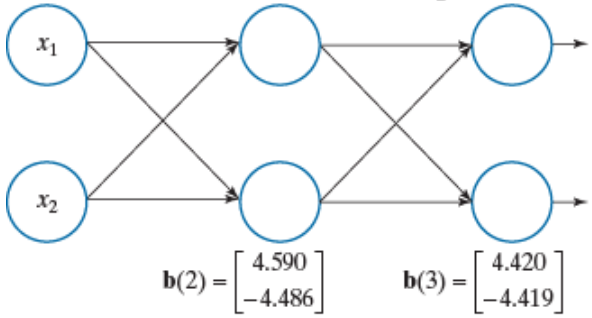




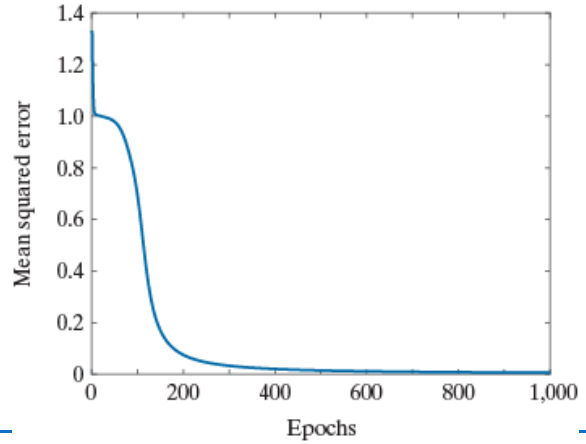


使用 $\alpha = 1.0$，一个均值为0、标准差为0.5的高斯随机权重值初始集以及sigmoid激活函数，进行1000个训练代后，得：

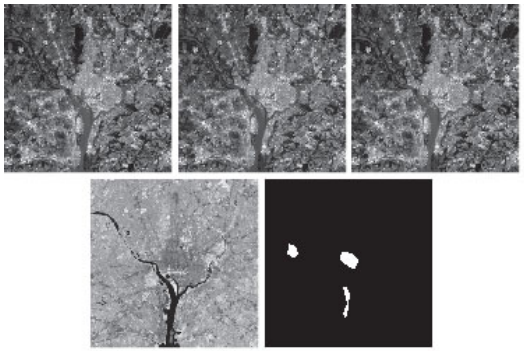
$$\mathbf{W}(2)=\begin{bmatrix}4.792 & 4.792\\ 4.486 & 4.486\end{bmatrix}\quad \mathbf{W}(3)=\begin{bmatrix}-9.180 & 9.429\\ 9.178 & -9.427\end{bmatrix}$$



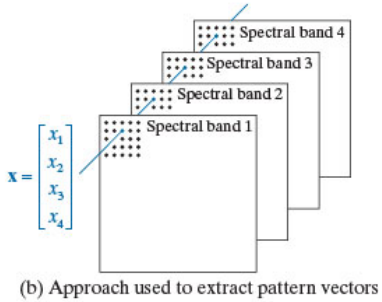
$$\mathbf{b}(2)=\begin{bmatrix}4.590\\ -4.486\end{bmatrix}\quad \mathbf{b}(3)=\begin{bmatrix}4.420\\ -4.419\end{bmatrix}$$



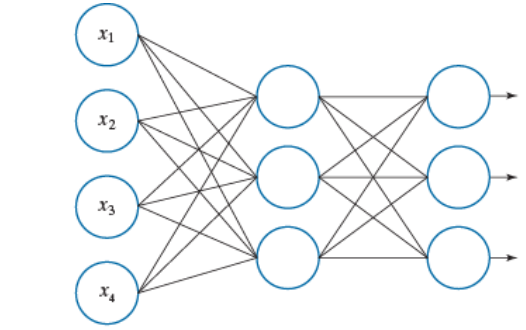
多光谱图像分类示例



(a) Images in spectral bands 1–4 and binary mask used to extract training samples



(b) Approach used to extract pattern vectors



$$\mathbf{W}(2) = \begin{bmatrix} 2.393 & 1.020 & 1.249 & -15.965 \\ 6.599 & -2.705 & -0.912 & 14.928 \\ 8.745 & 0.270 & 3.358 & 1.249 \end{bmatrix} \quad \mathbf{W}(3) = \begin{bmatrix} 4.093 & -10.563 & -3.245 \\ 7.045 & 9.662 & 6.436 \\ -7.447 & 3.931 & -6.619 \end{bmatrix}$$

$$\mathbf{b}(2) = [4.920 \quad -2.002 \quad -3.485]^T \qquad \mathbf{b}(3) = [3.277 \quad -14.982 \quad 1.582]^T$$

TABLE 13.4

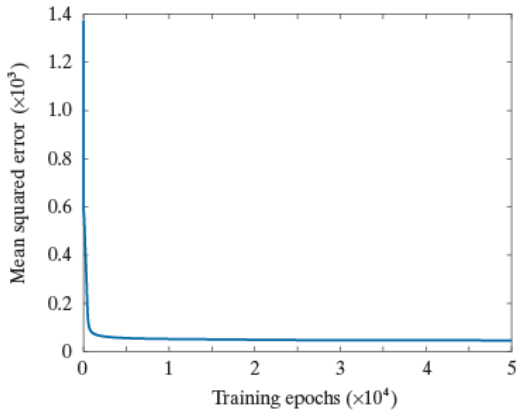
Recognition rate as a function of neural net architecture for $\alpha = 0.001$ and 1,000 training epochs. The network architecture is defined by the numbers in brackets. The first and last number inside each bracket refer to the number of input and output nodes, respectively. The inner entries give the number of nodes in each hidden layer.

Network Architecture	[4 2 3]	[4 3 3]	[4 4 3]	[4 5 3]	[4 2 2 3]	[4 4 3 3]	[4 4 4 3]	[4 10 3 3]	[4 10 10 3]
Recognition Rate	95.8%	96.2%	95.9%	96.1%	74.6%	90.8%	87.1%	84.9%	89.7%

TABLE 13.5

Recognition performance on the training set as a function of training epochs. The learning rate constant was $\alpha = 0.001$ in all cases.

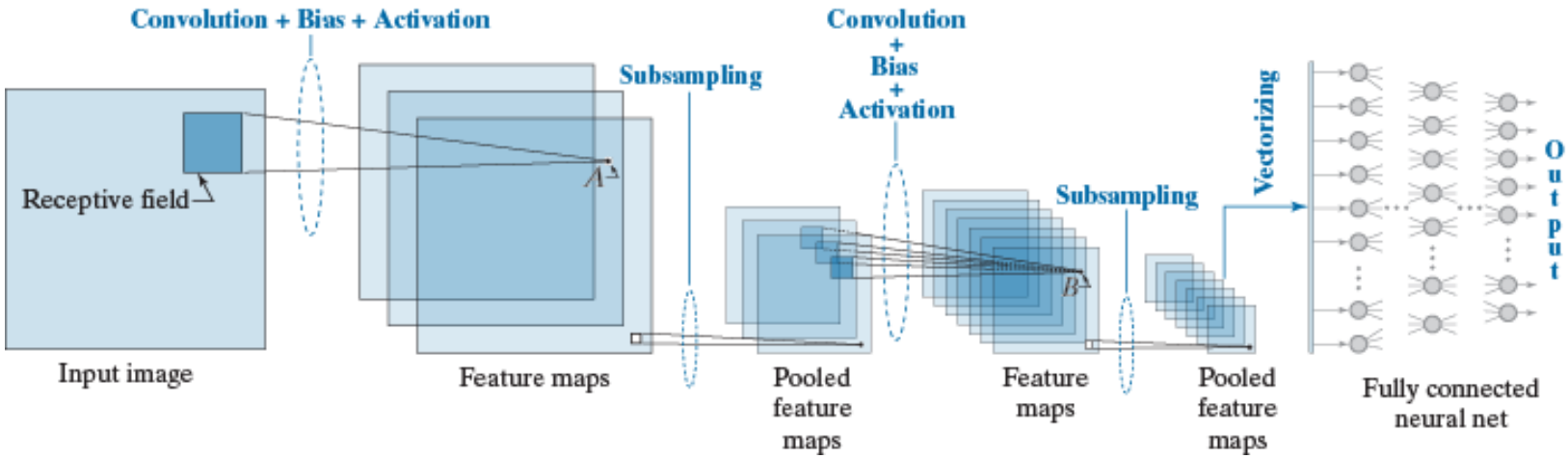
Training Epochs	1,000	10,000	20,000	30,000	40,000	50,000	60,000	70,000	80,000
Recognition Rate	95.3%	96.6%	96.7%	96.8%	96.9%	97.0%	97.0%	97.0%	97.0%



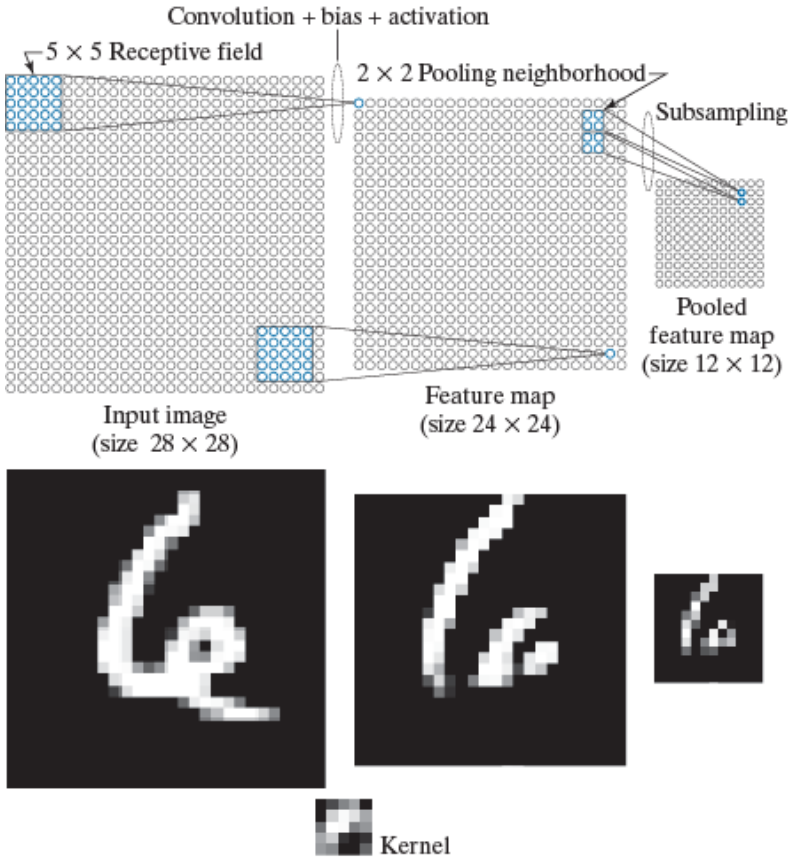
深度卷积神经网络（LeNet为例）

相比全连接网络，CNN的特点是：

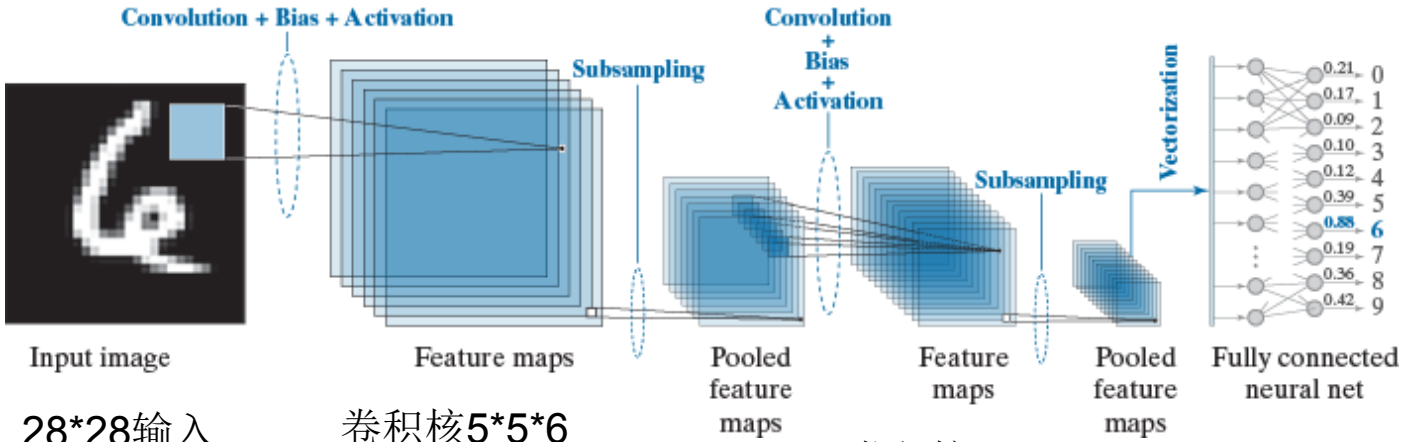
1. 能直接由原始图像数据学习二维特征；
2. 各层的连接方式是卷积（感受野，权重共享），而非全连接；
3. 层间有降采样（池化），降低输入中对平移变化的敏感度。



深度卷积神经网络（LeNet为例）



深度卷积神经网络（LeNet为例）



28*28输入

卷积核5*5*6

参数量：
5*5*6+6=156

2*2

卷积核5*5*12

参数量：
6*5*5*12+12=1812

全连接参数量：192*1

深度卷积神经网络（LeNet为例）

第一层的6个卷积核：

第2层的12个卷积核（12列）：

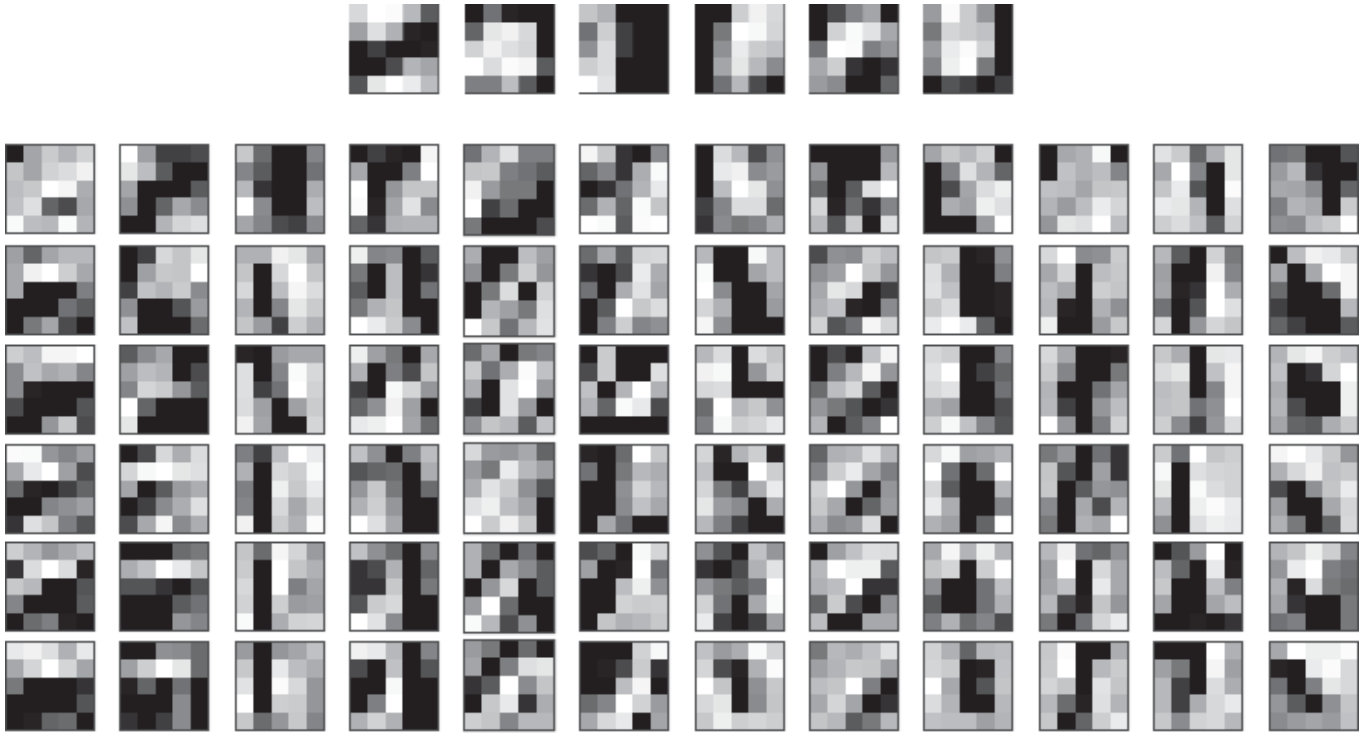
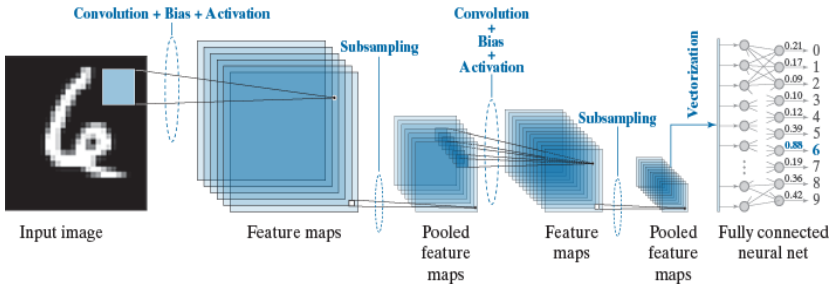
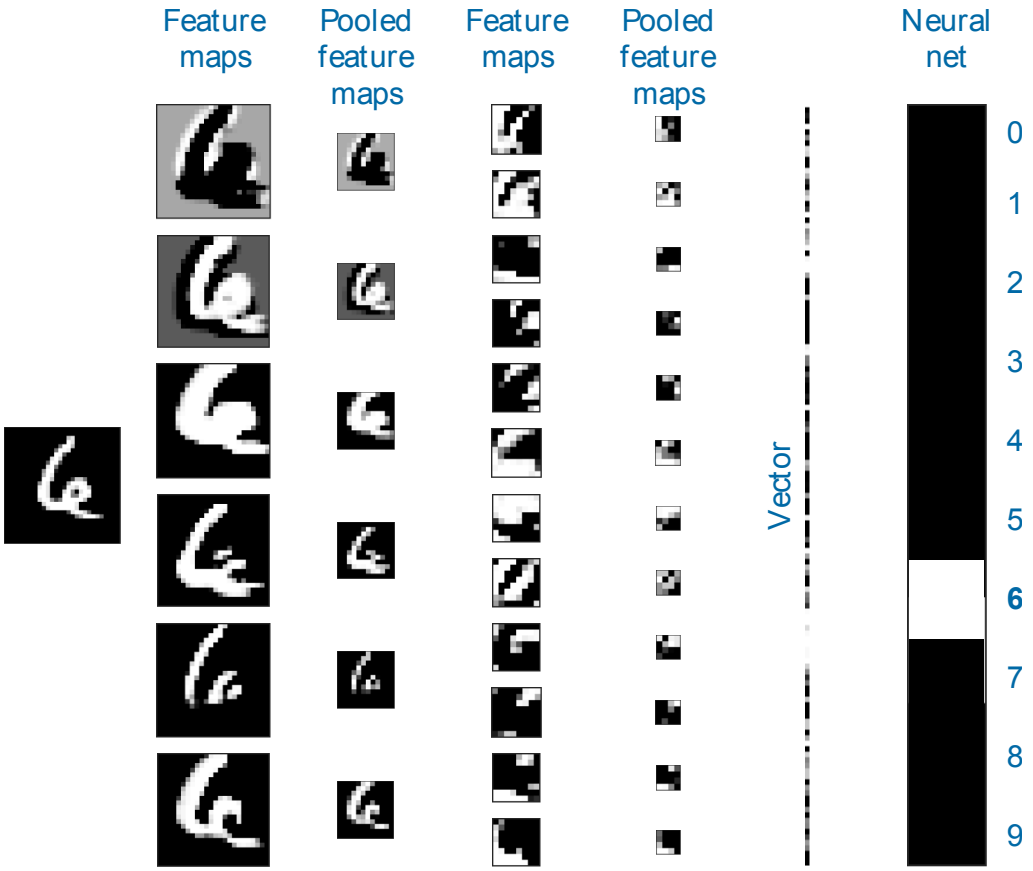


FIGURE 12.43 Top: The weights (shown as images of size 5×5) corresponding to the six feature maps in the first layer of the CNN in Fig. 12.42. Bottom: The weights corresponding to the twelve feature maps in the second layer.

深度卷积神经网络（LeNet为例）

对于特定输入图像，获得的特征图、通道数以及对应的输出。





深度卷积神经网络的模型及训练（LeNet为例）

$$z_{x,y}(\ell) = \sum_l \sum_k w_{l,k}(\ell) a_{x-l,y-k}(\ell-1) + b(\ell)$$
$$= w(\ell) \star a_{x,y}(\ell-1) + b(\ell)$$

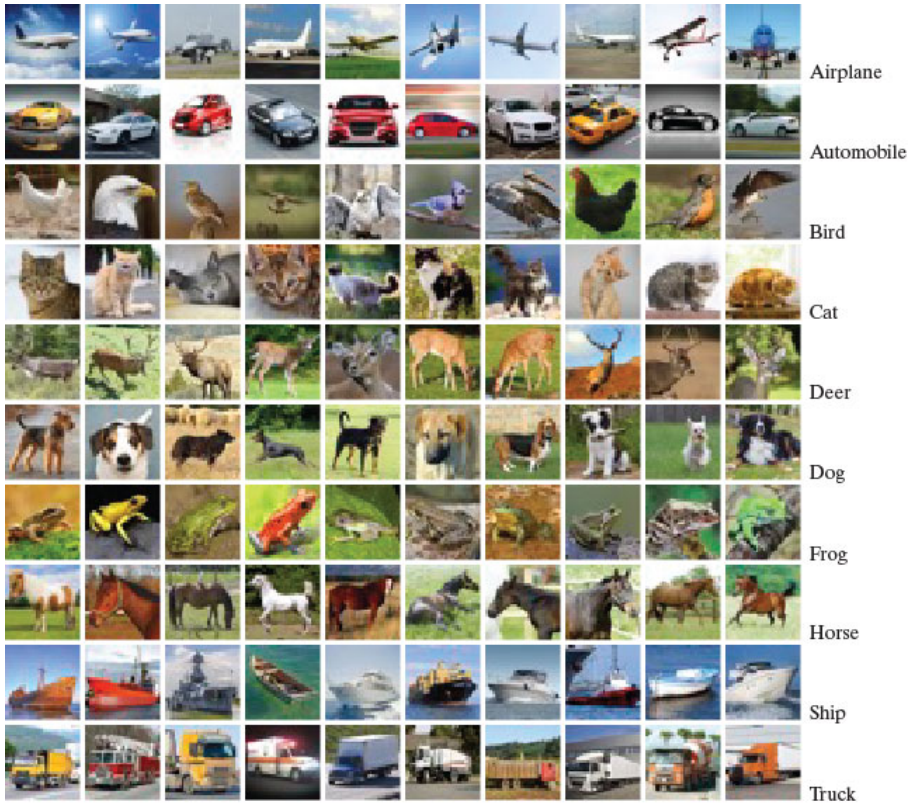
其中：$w \star a_{x,y} = \sum_l \sum_k w_{l,k} a_{x-l,y-k}$

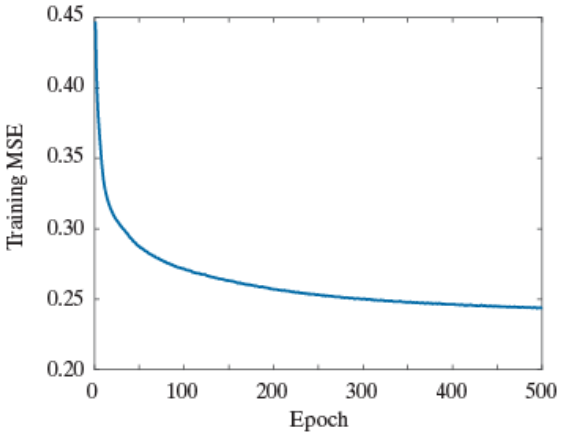
$$a_{x,y}(\ell) = h\left(z_{x,y}(\ell)\right)$$

Step	Description	Equations
Step 1	Input images	$a(0)$ = the set of image pixels in the input to layer 1
Step 2	Forward pass	For each neuron corresponding to location (x,y) in each feature map in layer ℓ compute: $z_{x,y}(\ell) = w(\ell) \star a_{x,y}(\ell-1) + b(\ell)$ and $a_{x,y}(\ell) = h\left(z_{x,y}(\ell)\right)$; $\ell = 1, 2, \ldots, L_c$
Step 3	Backpropagation	For each neuron in each feature map in layer ℓ compute: $\delta_{x,y}(\ell) = h'\left(z_{x,y}(\ell)\right)\left[\delta_{x,y}(\ell+1) \star \text{rot180}\left(w(\ell+1)\right)\right]$; $\ell = L_c - 1, L_c - 2, \ldots, 1$
Step 4	Update parameters	Update the weights and bias for each feature map using $w_{l,k}(\ell) = w_{l,k}(\ell) - \alpha \delta_{l,k}(\ell) \star \text{rot180}\left(a(\ell-1)\right)$ and $b(\ell) = b(\ell) - \alpha \sum_x \sum_y \delta_{x,y}(\ell)$; $\ell = 1, 2, \ldots, L_c$

识别自然图像例子

FIGURE 12.56
Mini images of size 32×32 pixels, representative of the 50,000 training and 10,000 test images in the CIFAR-10 database (the 10 stands for ten classes). The class names are shown on the right. (Images courtesy of Pearson Education.)





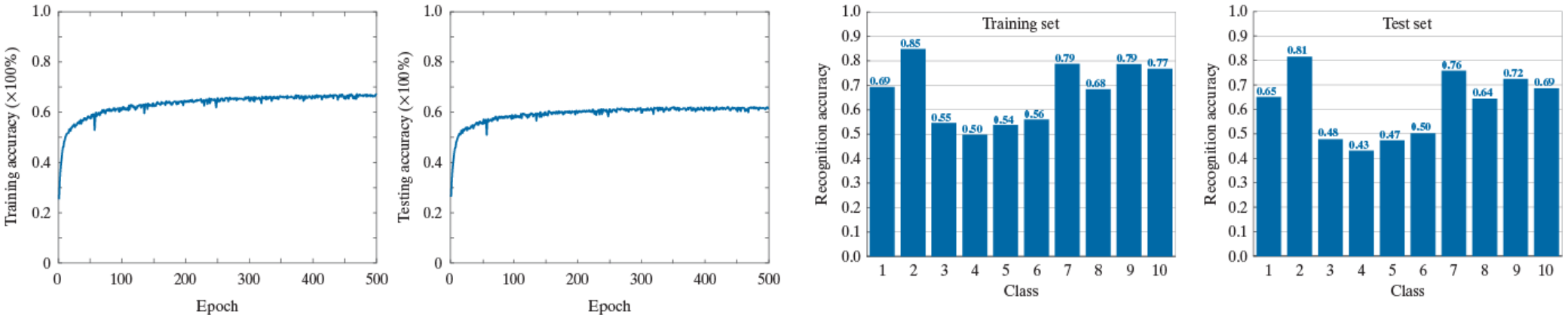
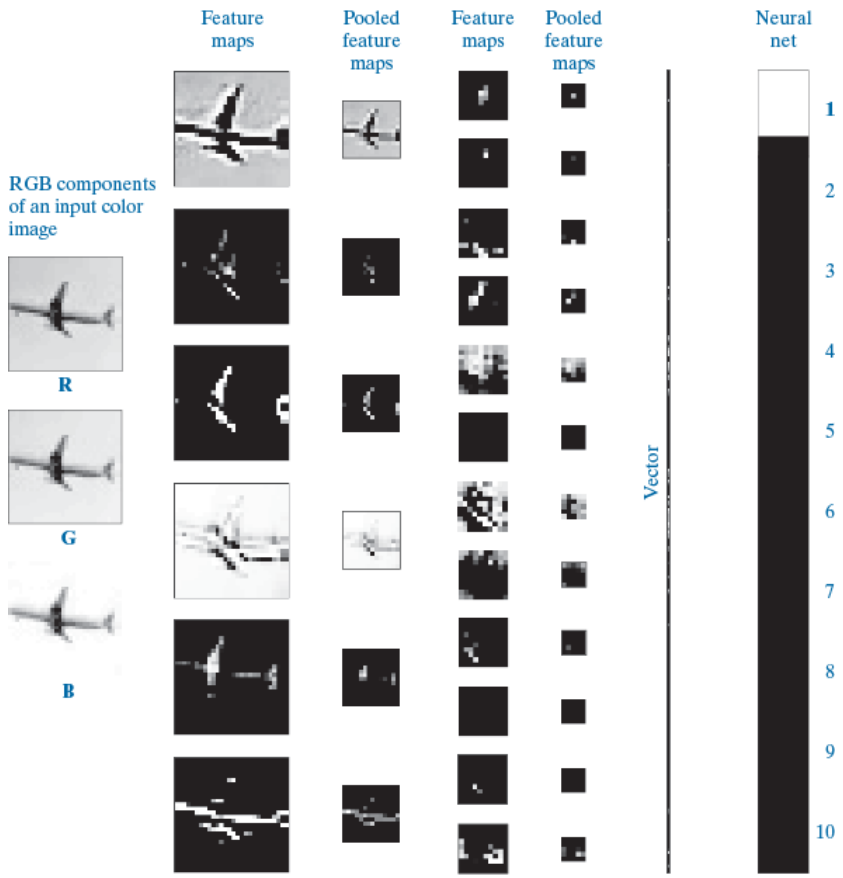


FIGURE 12.58 (a) Training accuracy (percent correct recognition of the training set) as a function of epoch for the 50,000 training images in the CIFAR-10 database. (b) Accuracy as a function of epoch for the 10,000 CIFAR-10 test images.


Feature maps
Pooled feature maps
Feature maps
Pooled feature maps
Neural net
RGB components of an input color image
R
G
B
Vector
1
2
3
4
5
6
7
8
9
10

全卷积网络（FCN）

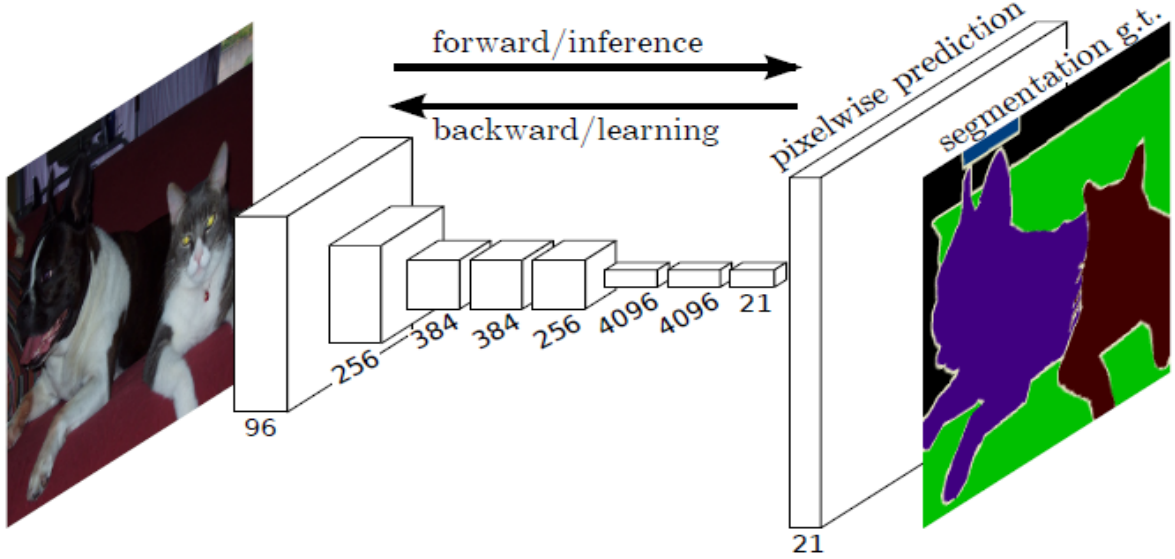
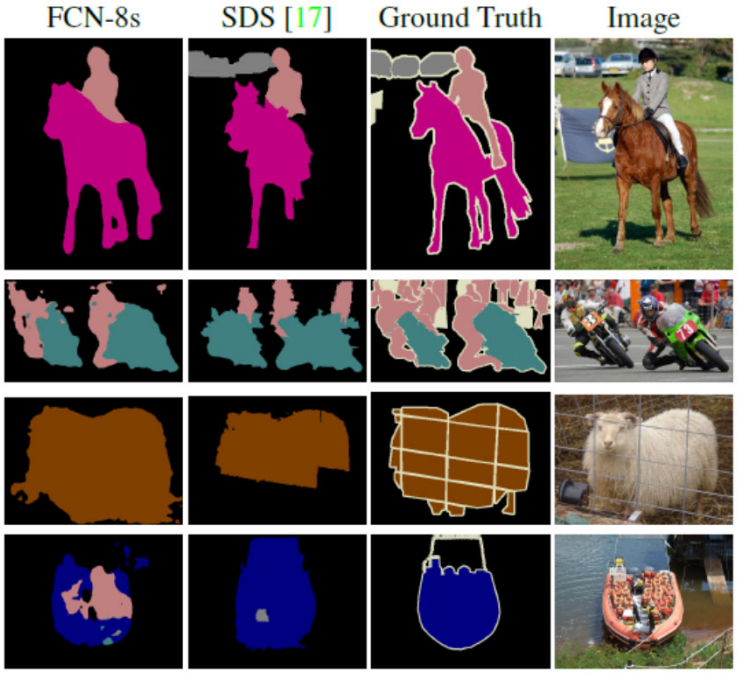


Figure 1. Fully convolutional networks can efficiently learn to make dense predictions for per-pixel tasks like semantic segmentation.



本章作业：
习题12.2， 12.9，12.16，12.30

1. 本章无编程作业。请大家考虑大作业选题（自愿），在5月25日前先上报题目。
2. 选题可以选与图像处理相关的任何主题。除了算法软件实现外，鼓励在瑞星微板子上实现嵌入式应用（总分加1分）。