



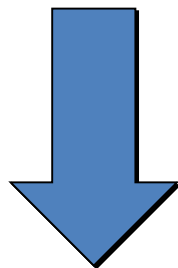
分类与预测

2021/11/15

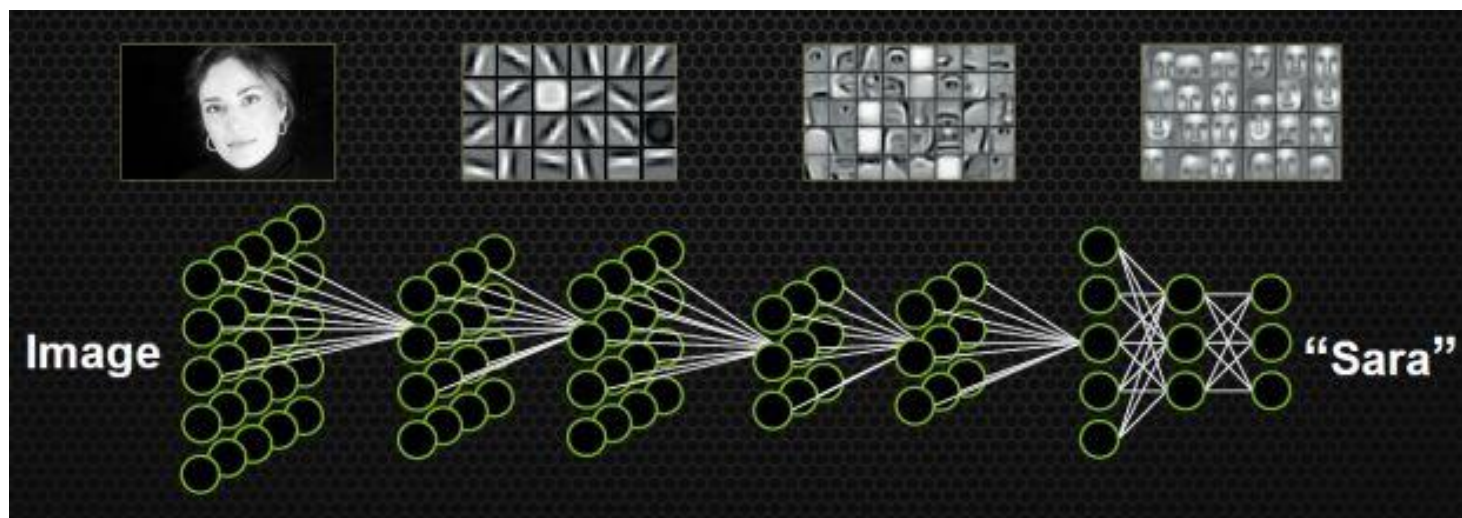
目录

1	背景
2	决策树
3	回归
4	集成学习
5	神经网络
6	深度学习

深度学习-手工设计特征到自动化构造特征

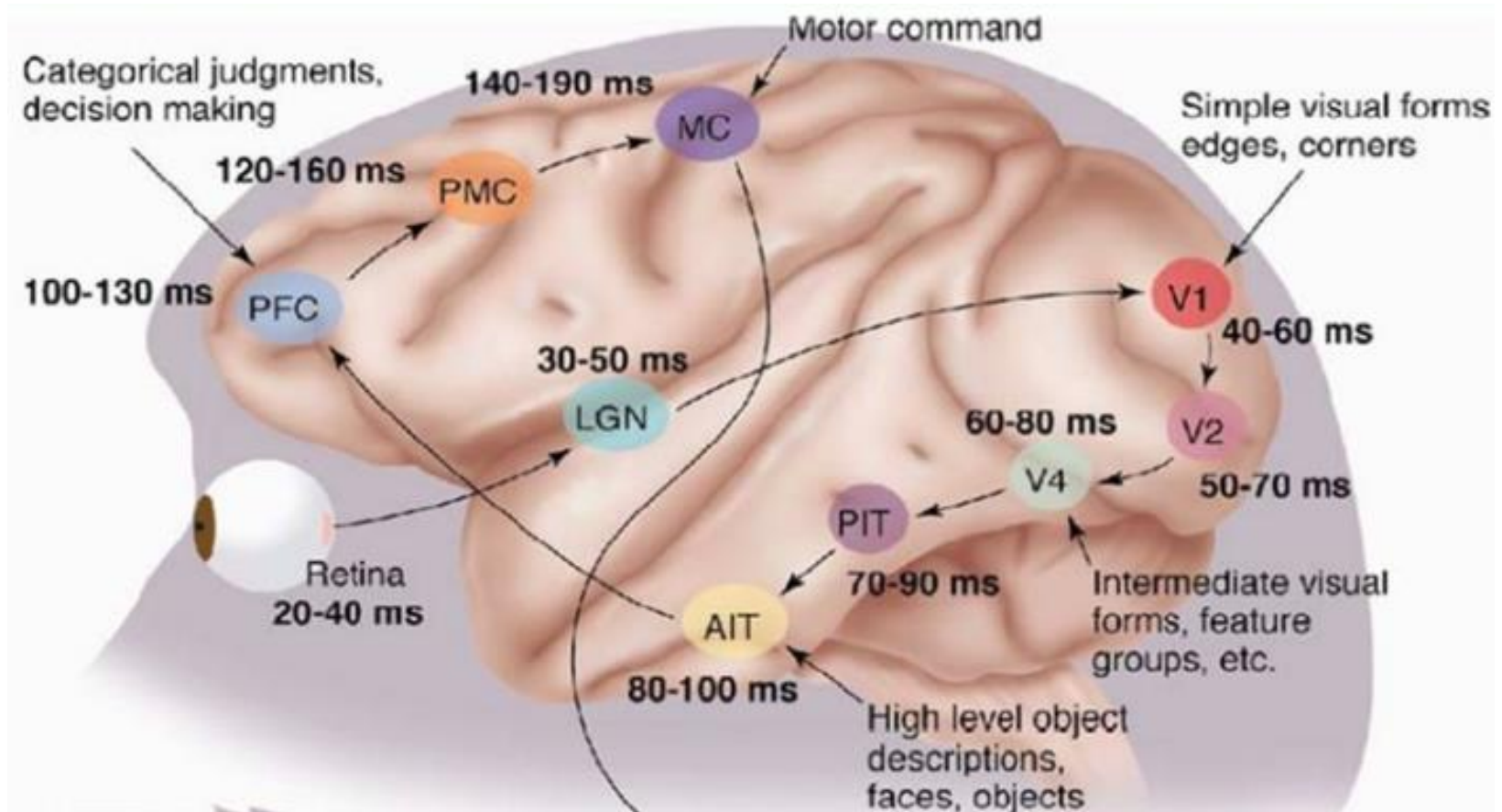


良好的特征表达，对最终算法的准确性起了非常关键的作用，而且系统主要的计算和测试工作都耗在这一大部分。



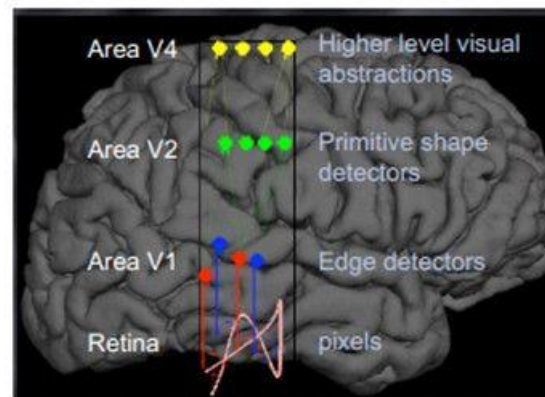
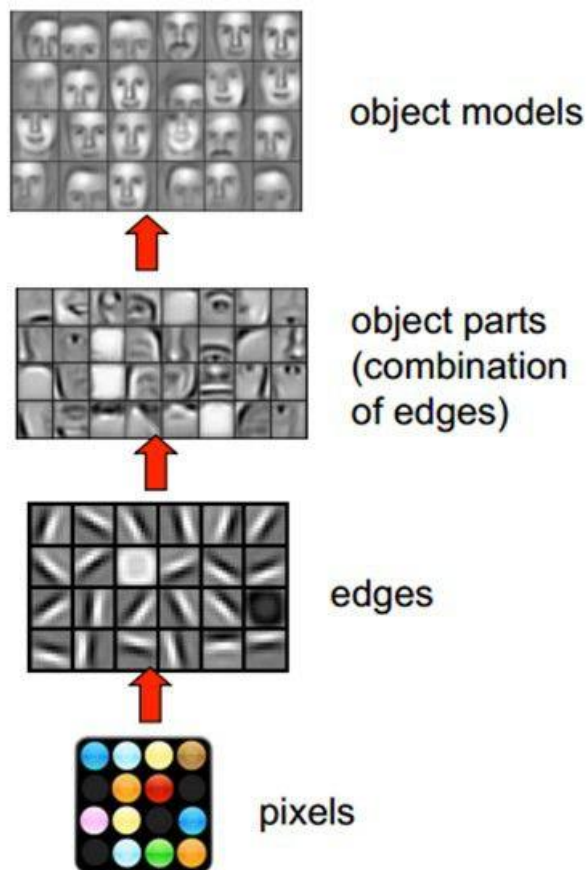
深度学习-大脑视觉处理机制

1981年的诺贝尔医学奖获得者 David Hubel和Torsten Wiesel发现了视觉系统的信息处理机制，他们发现了一种被称为“**方向选择性细胞的神经元细胞**”，当瞳孔发现了眼前的物体的边缘，而且这个边缘指向某个方向时，这种神经元细胞就会活跃。



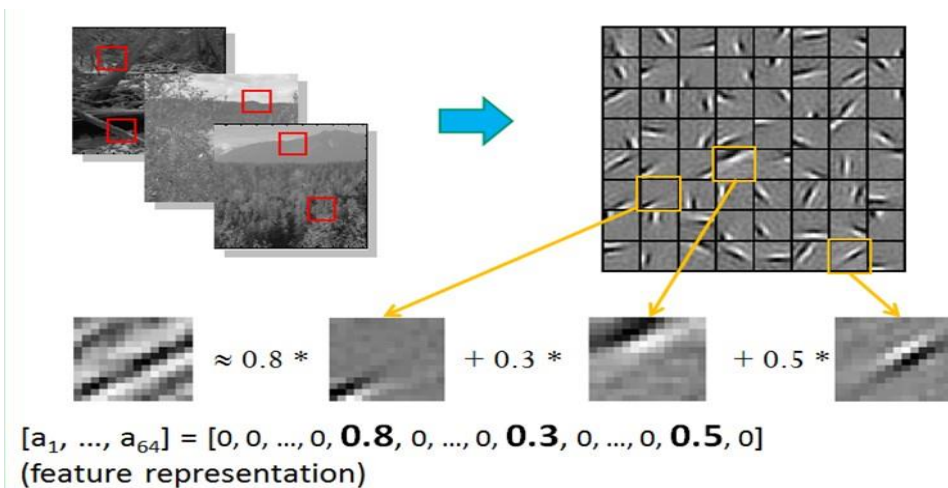
深度学习-大脑视觉处理机制

人的视觉功能一个是抽象，一个是迭代。抽象就是把非常具体的形象的元素，即原始的光线像素等信息，抽象出来形成有意义的概念。这些有意义的概念又会往上迭代，变成更加抽象，人可以感知到的抽象概念。



深度学习-原理

- 1995年前后, Bruno Olshausen和David Field两位学者任职Cornell University, 他们试图同时用生理学和计算机的手段, 双管齐下, 研究视觉问题。
- 收集很多黑白照片, 从中提取出400个小碎片, 每个照片碎片的尺寸均为16x16像素, 把这400个碎片标记为 $S[i], i=0, \dots, 399$ 。再从这些黑白风景照片中, 随机提取另一个碎片, 尺寸也是16x16像素, 把这个碎片标记为 T 。提出问题, 如何从这400个碎片中, 选取一组碎片, $S[k]$, 通过叠加的办法, 合成出一个新的碎片, 而这个新的碎片, 与随机选择的目标碎片 T , 尽可能相似, 同时, $S[k]$ 的数量尽可能少。



$\text{Sum}_k(a[k]*S[k]) \rightarrow T$, 其中 $a[k]$ 是在叠加碎片 $S[k]$ 时的权重系数。

稀疏编码:

- 1) 选择一组 $S[k]$, 然后调整 $a[k]$, 使得 $\text{Sum}_k(a[k]*S[k])$ 最接近 T 。
- 2) 固定住 $a[k]$, 在400个碎片中, 选择其它更合适的碎片 $S'[k]$, 替代原先的 $S[k]$, 使得 $\text{Sum}_k(a[k]*S'[k])$ 最接近 T 。

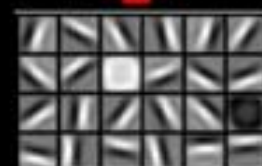
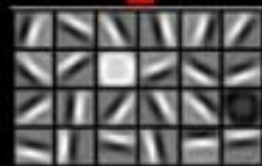
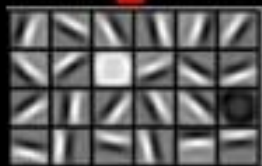
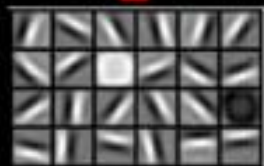
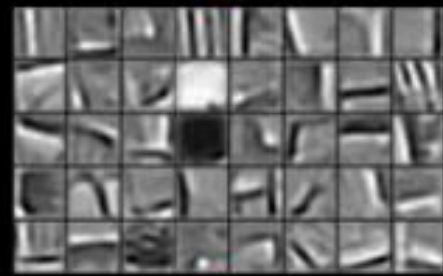
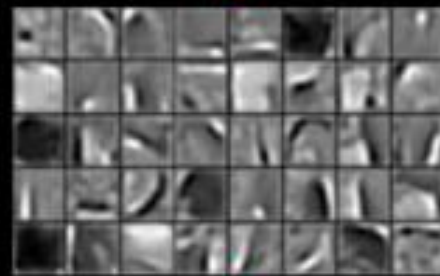
Features learned from training on different object classes.

Faces

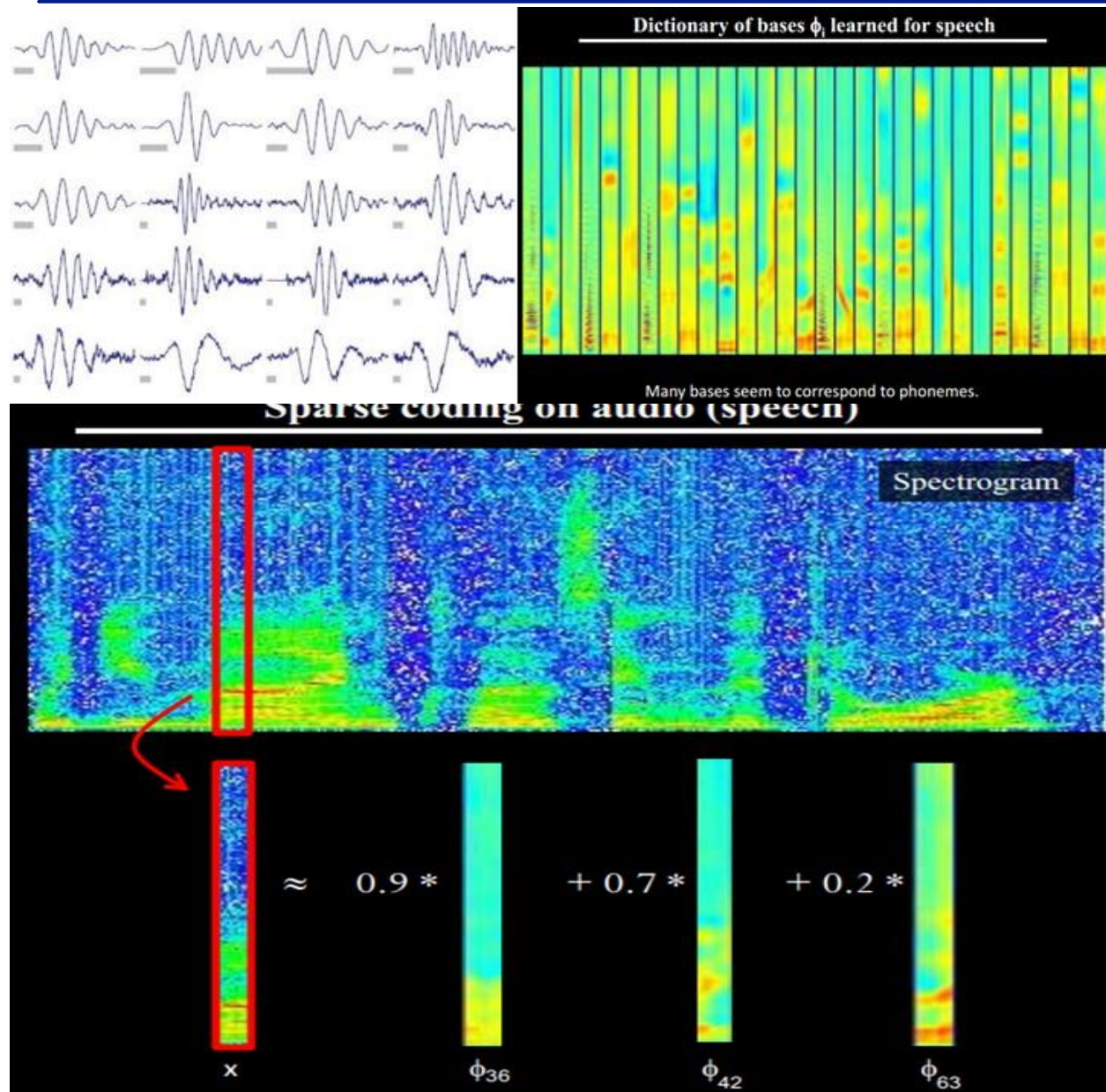
Cars

Elephants

Chairs

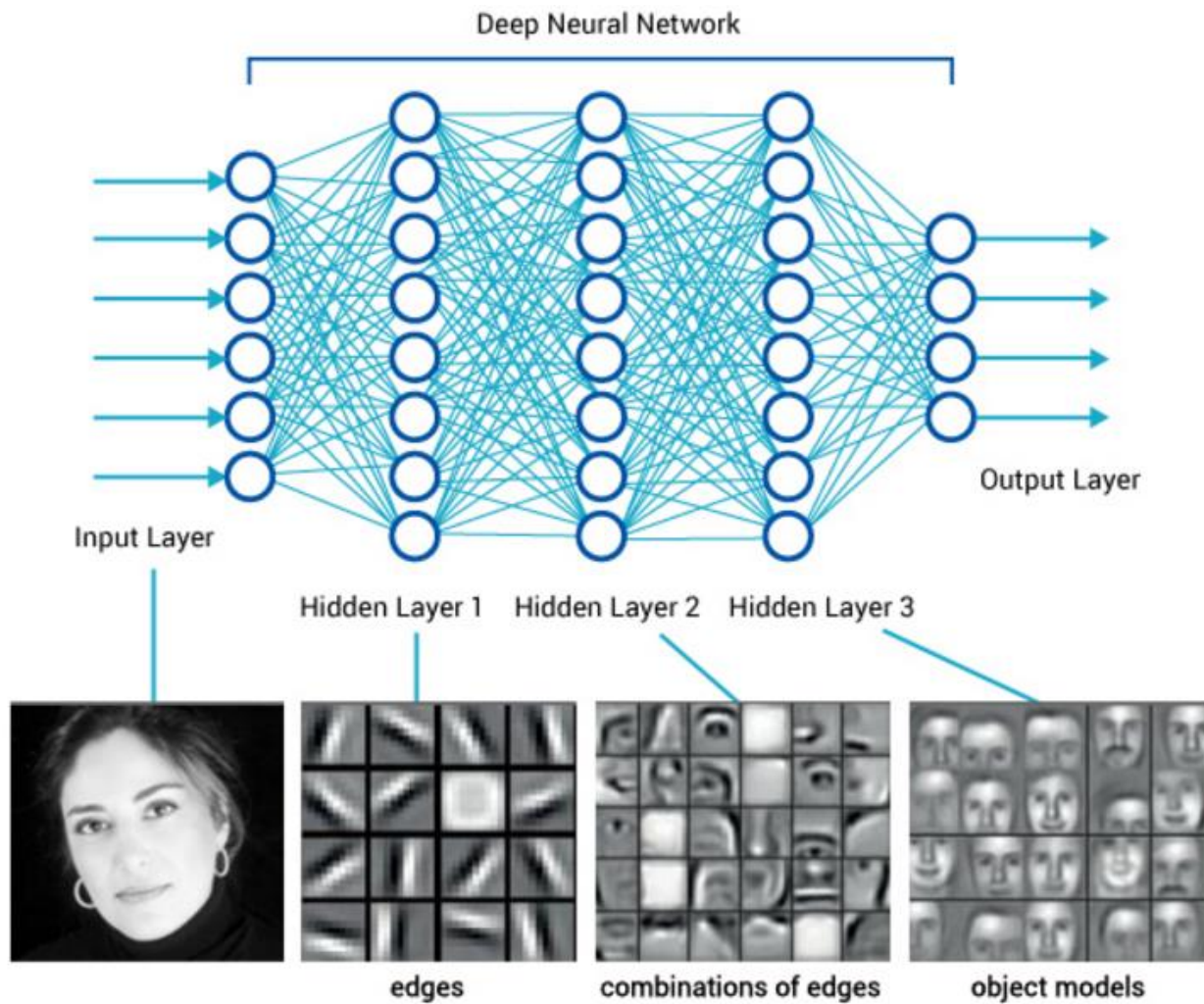


深度学习-原理



科学家们还发现，不仅图像存在这个规律，声音也存在。他们从未标注的声音中发现了20种基本的声音结构，其余的声音可以由这20种基本结构合成。

深度学习-深层前馈神经网络



定理 4.1 – 通用近似定理 (Universal Approximation Theorem)

[Cybenko, 1989, Hornik et al., 1989]: 令 $\varphi(\cdot)$ 是一个非常数、有界、单调递增的连续函数, \mathcal{I}_d 是一个 d 维的单位超立方体 $[0, 1]^d$, $C(\mathcal{I}_d)$ 是定义在 \mathcal{I}_d 上的连续函数集合。对于任何一个函数 $f \in C(\mathcal{I}_d)$, 存在一个整数 m , 和一组实数 $v_i, b_i \in \mathbb{R}$ 以及实数向量 $\mathbf{w}_i \in \mathbb{R}^d$, $i = 1, \dots, m$, 以至于我们可以定义函数

$$F(\mathbf{x}) = \sum_{i=1}^m v_i \varphi(\mathbf{w}_i^T \mathbf{x} + b_i), \quad (4.33)$$

作为函数 f 的近似实现, 即

$$|F(\mathbf{x}) - f(\mathbf{x})| < \epsilon, \forall \mathbf{x} \in \mathcal{I}_d. \quad (4.34)$$

其中 $\epsilon > 0$ 是一个很小的正数。

根据通用近似定理, 对于具有线性输出层和至少一个使用“挤压”性质的激活函数的隐藏层组成的前馈神经网络, 只要其隐藏层神经元的数量足够, 它可以以任意的精度来近似任何从一个定义在实数空间中的有界闭集函数。

- ▶ 神经网络可以作为一个“万能”函数来使用，可以用来进行复杂的特征转换，或逼近一个复杂的条件分布。

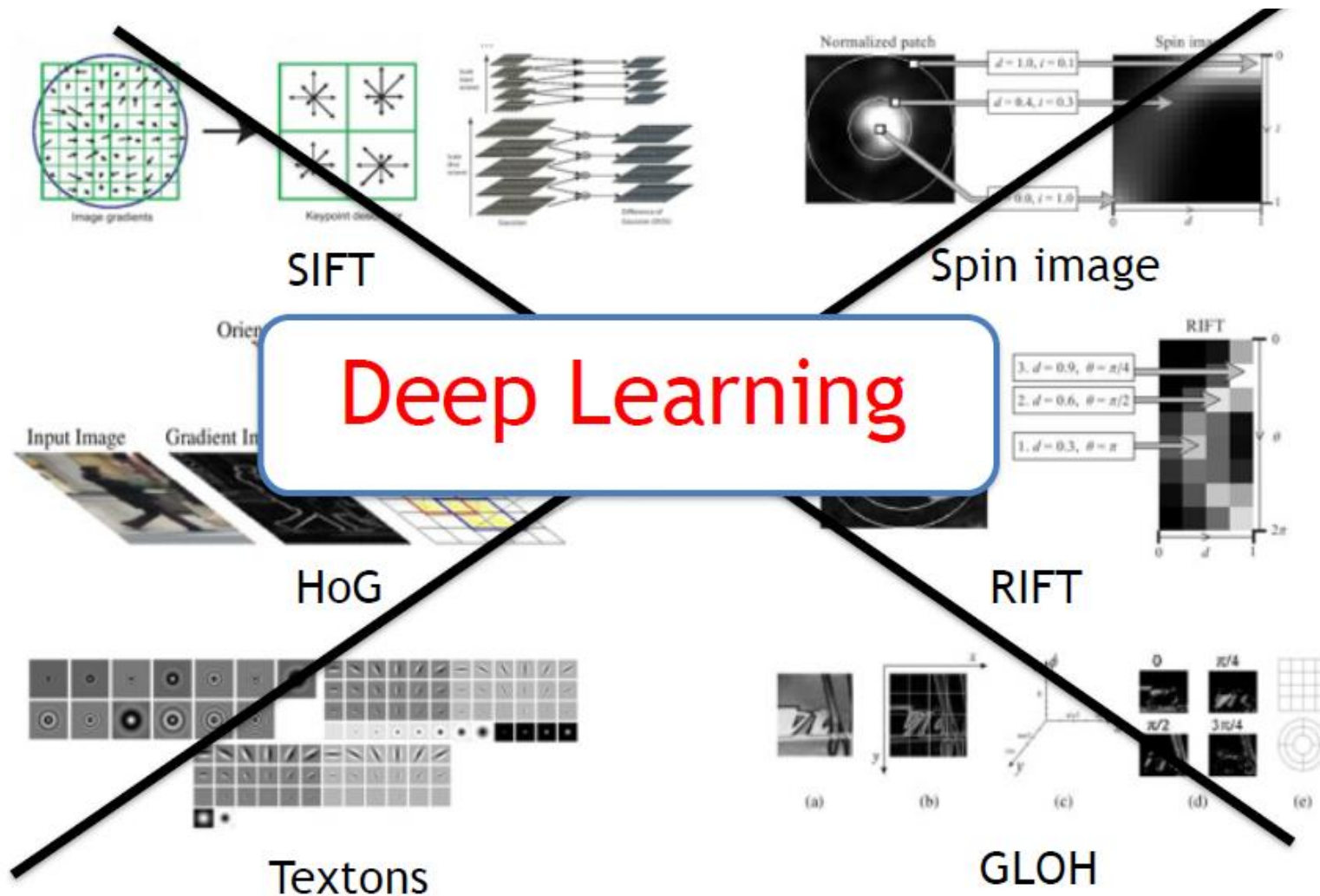
$$\hat{y} = g(\underline{\varphi(\mathbf{x})}, \theta)$$

分类器

神经网络

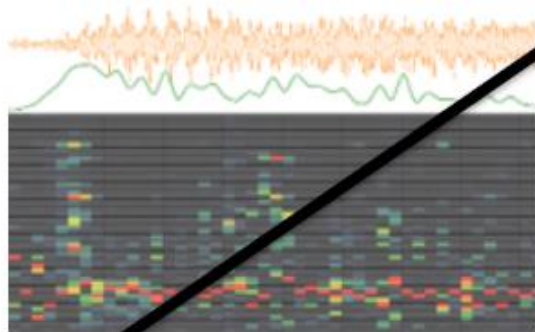
- ▶ 如果 $g(\cdot)$ 为Logistic回归，那么Logistic回归分类器可以看成神经网络的最后一层。

深度学习-Computer Vision Features

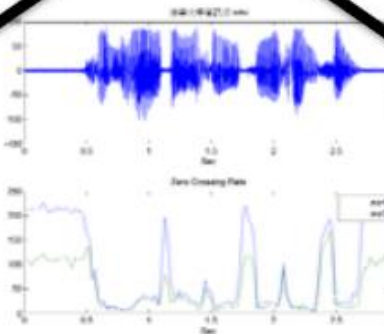


深度学习-Audio Features

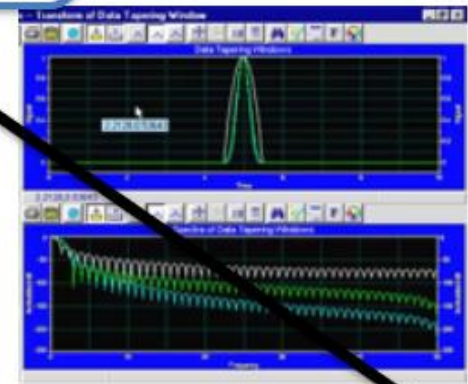
Deep Learning



Flux



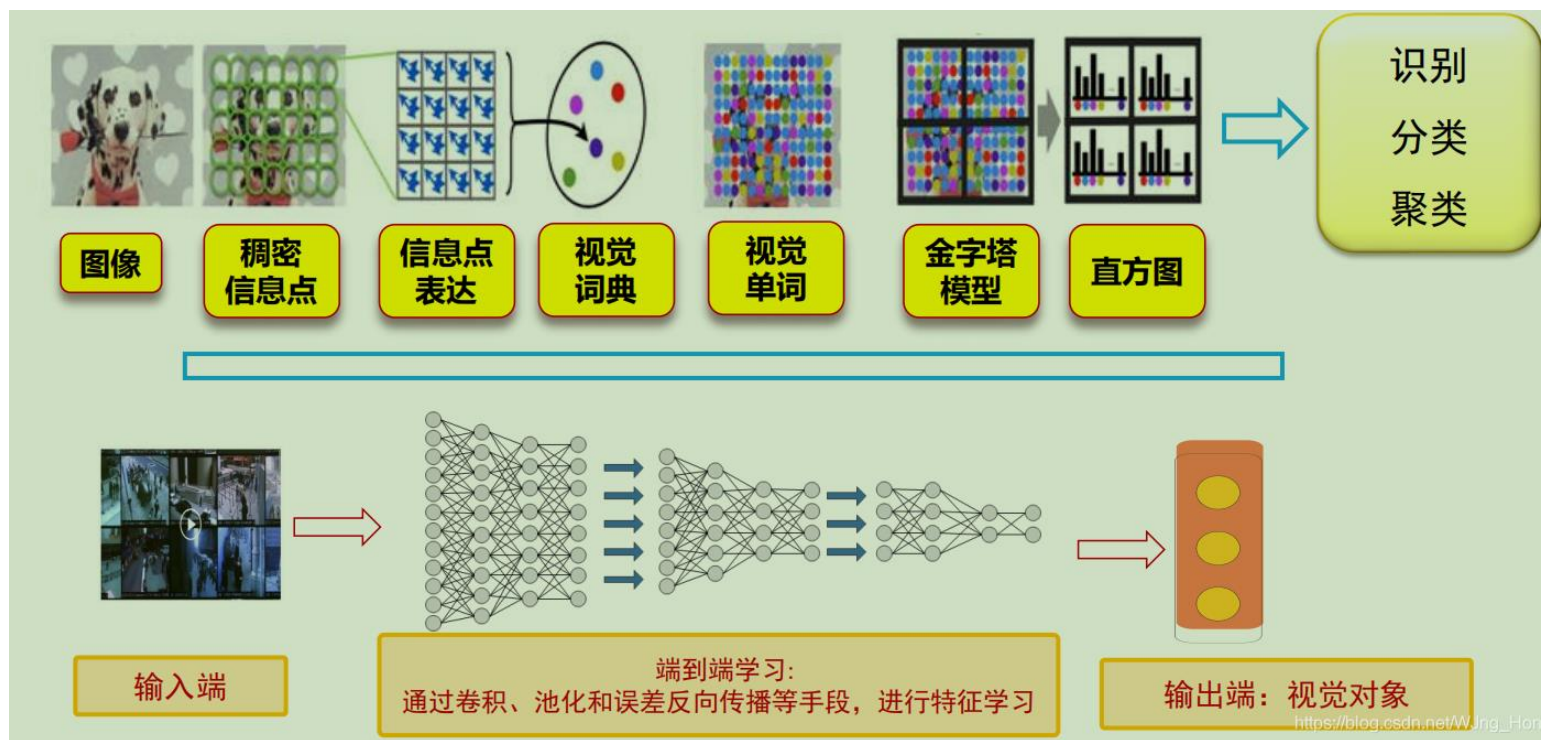
ZCR



Rolloff

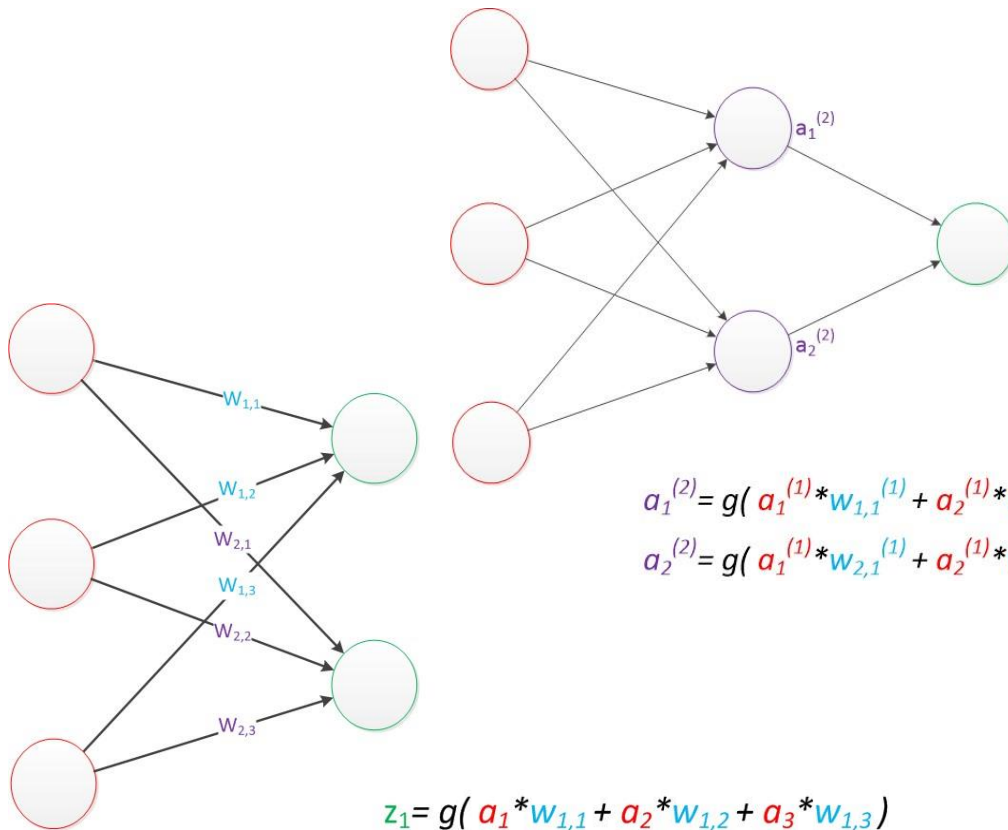
深度学习-从浅层学习到深度学习

典型浅层学习流程：首先从图像中检测出其关键的信息点；然后对这些信息点进行表达，如著名的Sift算法，就形成了每个像素点128维的向量；对信息点的表达向量进行聚类，得到信息点所组成的视觉词典；在视觉词典的基础之上，用一个一个的视觉单词来表达整幅图像，由于图像具有空间近邻关系，因此可以通过金字塔模型来捕获视觉单词之间所形成的空间结构；在金字塔模型的基础之上，可以形成图像的视觉直方图，图像从表达成了视觉图像直方图的向量形式，基于此可以对图像进行聚类、分类。



深度学习流程：从输入端输入经过卷积、池化、反向传播等操作后可得到原始数据的特征表达，基于此特征表达，能够对原始数据进行识别与分类。这里深度学习主要体现是通过层层抽象的方法直接得到输入数据的特征表达。

人工神经网络-浅层学习到深度学习

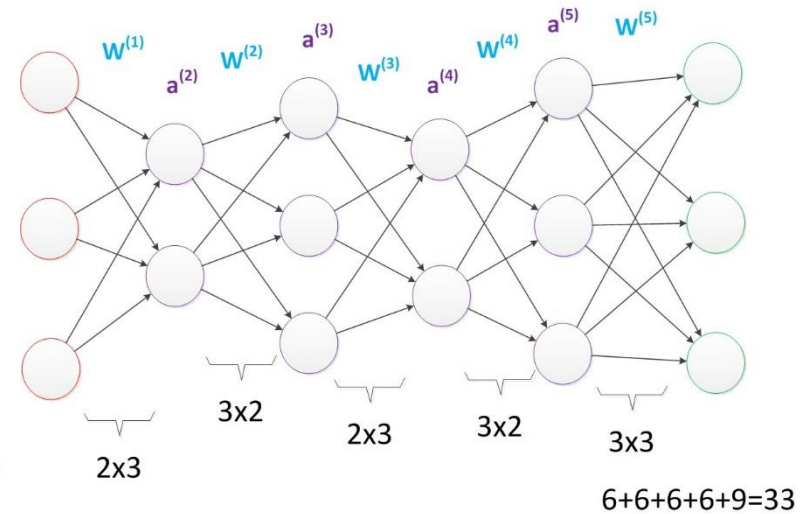


$$a_1^{(2)} = g(a_1^{(1)} * w_{1,1}^{(1)} + a_2^{(1)} * w_{1,2}^{(1)} + a_3^{(1)} * w_{1,3}^{(1)})$$

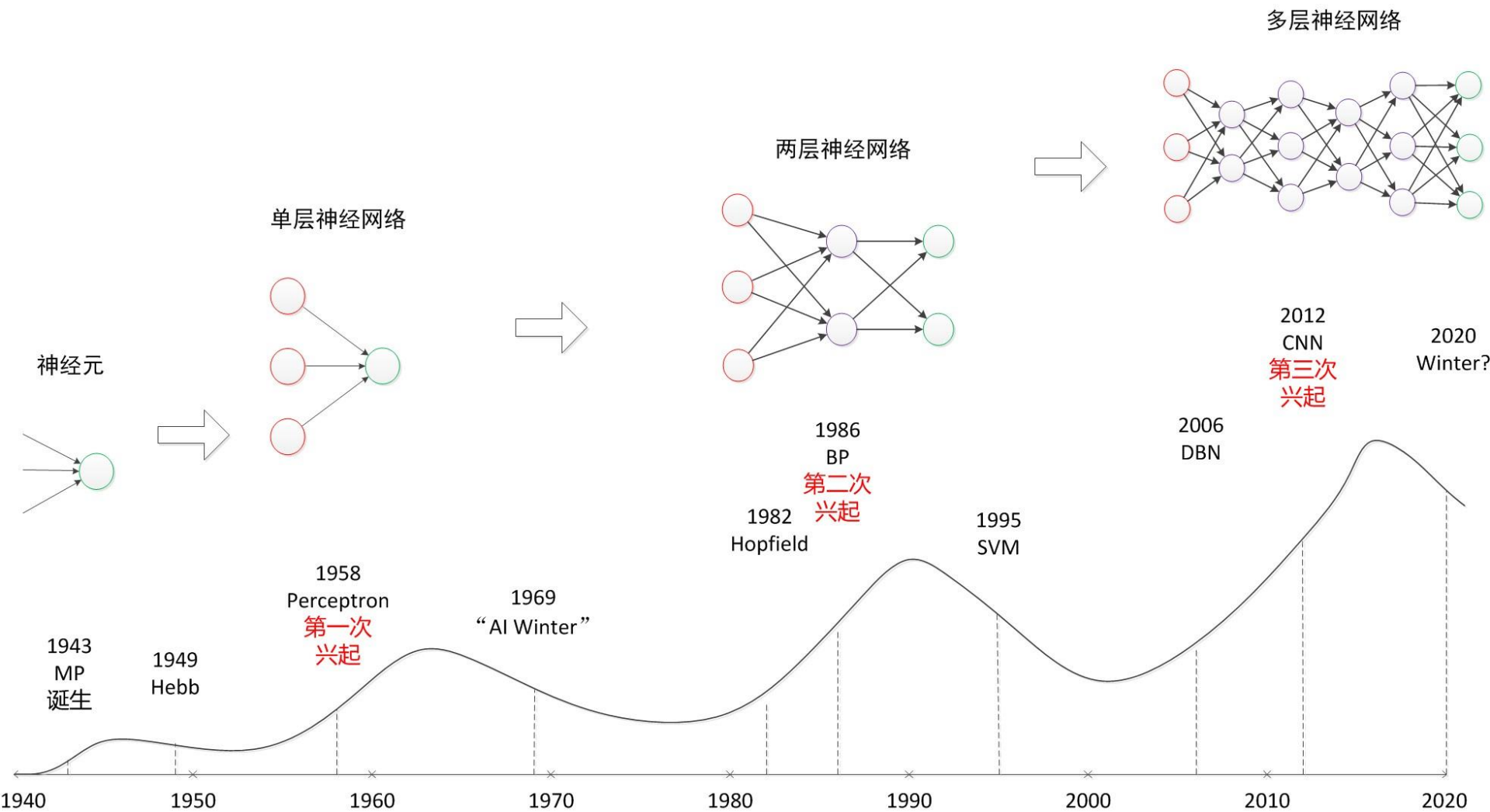
$$a_2^{(2)} = g(a_1^{(1)} * w_{2,1}^{(1)} + a_2^{(1)} * w_{2,2}^{(1)} + a_3^{(1)} * w_{2,3}^{(1)})$$

$$z_1 = g(a_1 * w_{1,1} + a_2 * w_{1,2} + a_3 * w_{1,3})$$

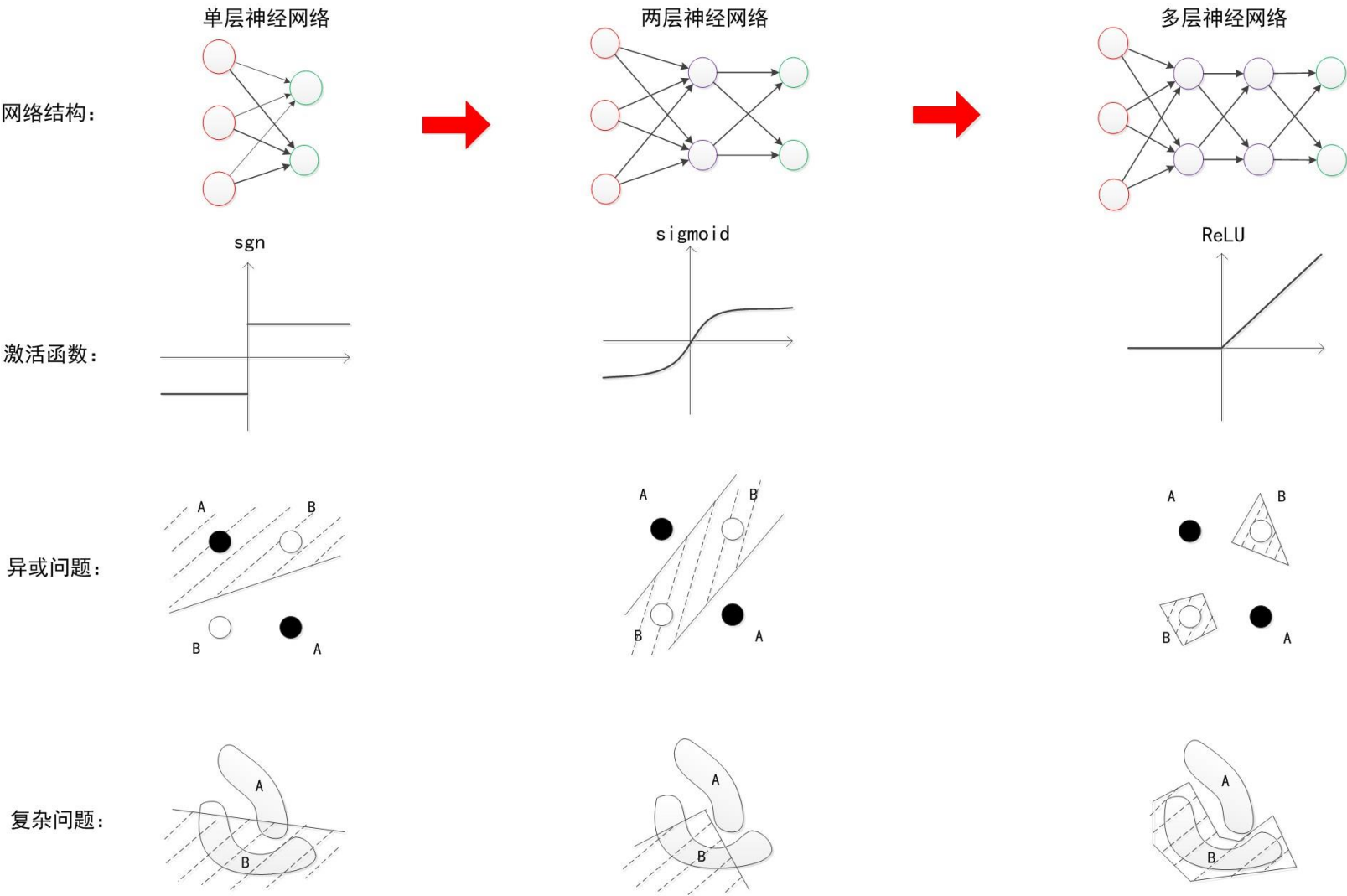
$$z_2 = g(a_1 * w_{2,1} + a_2 * w_{2,2} + a_3 * w_{2,3})$$



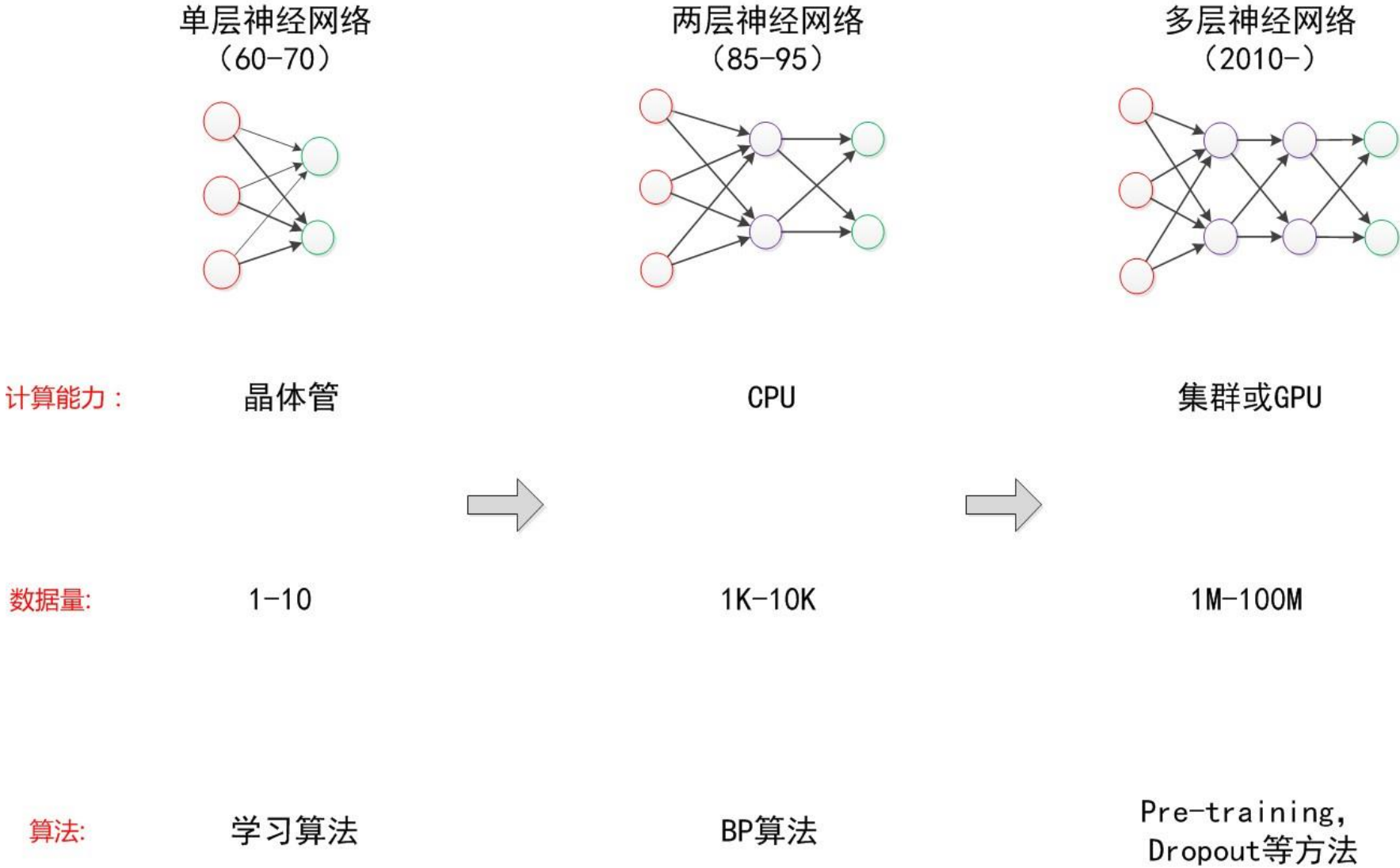
三起三落的神经网络



表示能力不断增强



神经网络发展的外在原因

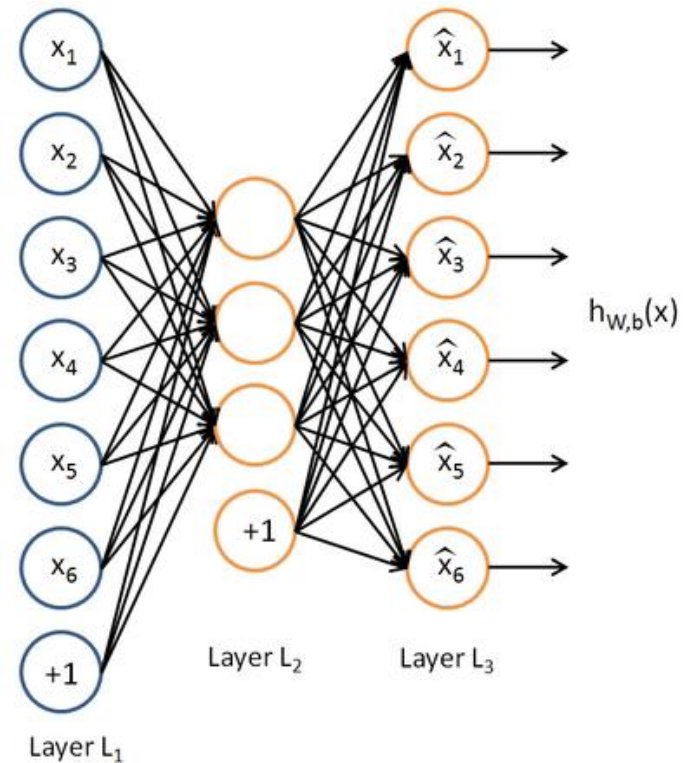


深度学习常用模型

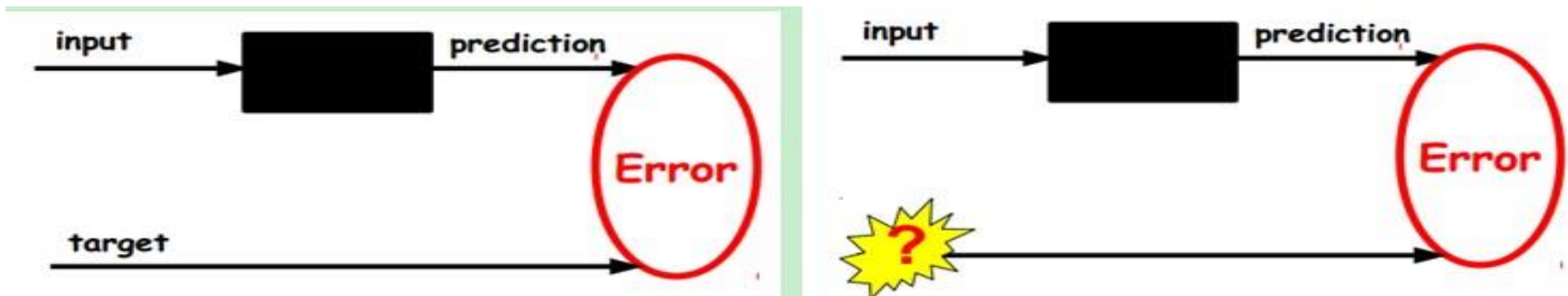
- Auto Encoder (自动编码器)
- Convolutional Neural Networks (卷积神经网络)
- Transformer

深度学习-AutoEncoder

- Deep Learning最简单的一种方法
- 利用人工神经网络本身的层次结构特点
 - 如果给定一个神经网络，假设其输出与输入是相同的，然后训练调整其参数，得到每一层中的权重。
 - 自然地，就得到了输入 I 的几种不同表示（每一层代表一种表示），这些表示就是特征。
- 自动编码器就是一种尽可能复现输入信号的神经网络。
 - 为了实现这种复现，自动编码器就必须捕捉可以代表输入数据的最重要的因素，就像PCA那样，找到可以代表原信息的主要成分



深度学习-AutoEncoder

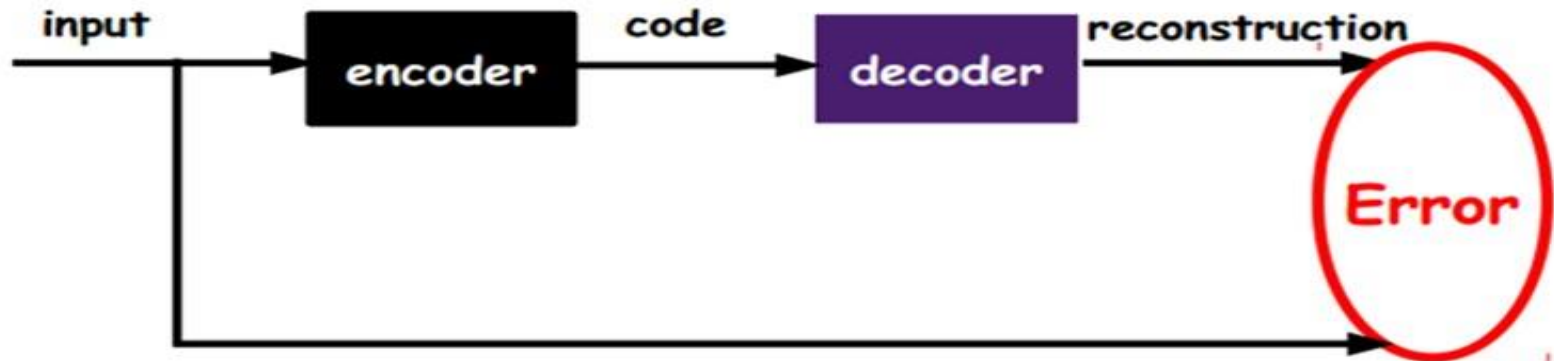


在如左图所示的前向神经网络中，输入的样本是有标签的，即（input, target），这样可以**根据当前输出和target（label）之间的差去更新前面各层的参数，直到收敛。**

但现在我们只有无标签数据，也就是右边的图。那么这个误差怎么得到？

深度学习-AutoEncoder

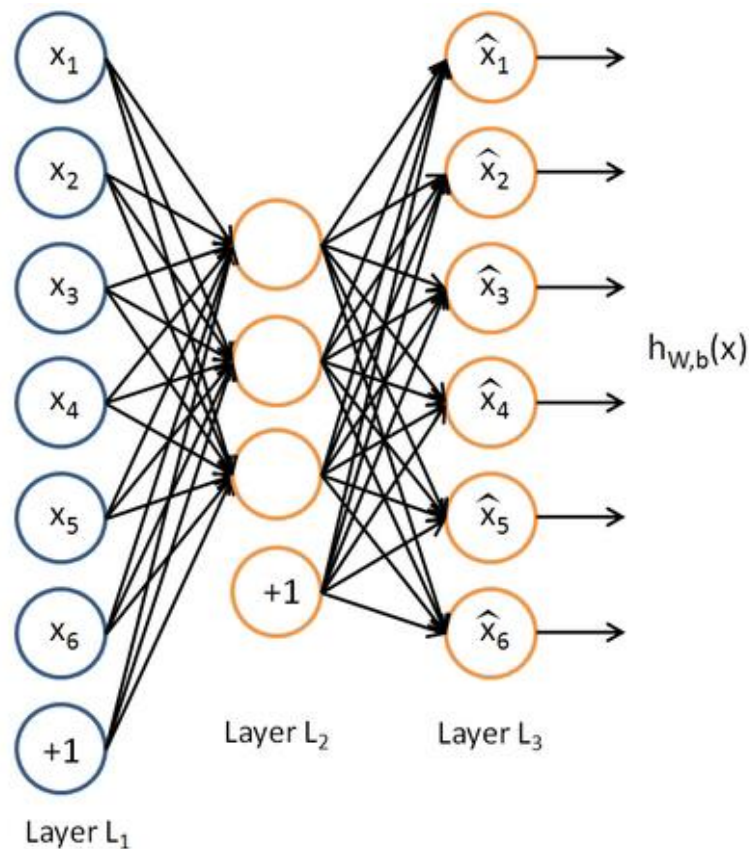
- 将input输入一个encoder编码器，就会得到一个code，这个code也就是输入的一个表示
 - 那么我们怎么知道这个code表示的就是input呢？
- 增加一个decoder解码器
 - decoder输出的信息 vs 开始的输入信号input
- 通过调整encoder和decoder的参数，使得重构误差最小，这样就得到输入input信号的一个表示了，也就是编码code。
- 因为是无标签数据，所以误差的来源就是直接重构后与原输入相比得到。



■ 网络结构

- 三层结构
 - 输入层，隐藏层，输出层
- 神经元模型
- 限定神经元的数量
 - 输入层神经元数=输出层神经元数
 - 隐层神经元数量<输入层神经元数量

意义：迫使隐藏层节点学习得到输入数据的压缩表示方法



深度学习-AutoEncoder

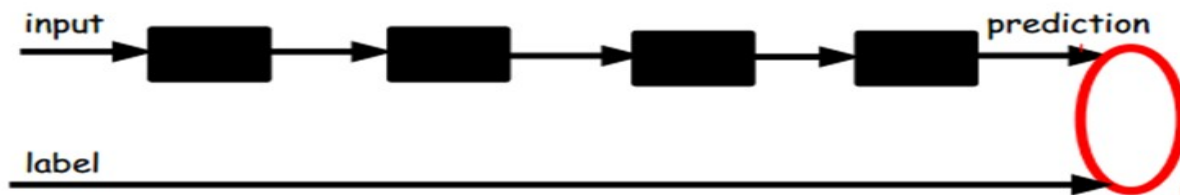
■ 监督学习

- 最后层的特征code输入到分类器中，基于有标签样本，通过监督学习对网络进行微调

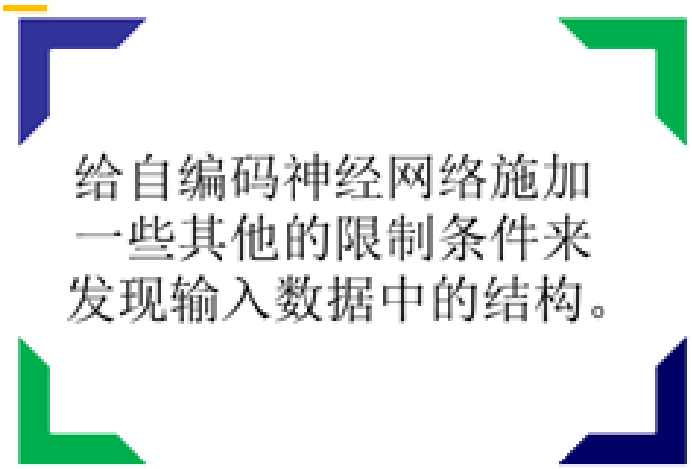
1、只调整分类器



2、通过有标签样本， 微调整个系统：（如果有足够多的数据， end-to-end learning端 对端学习）



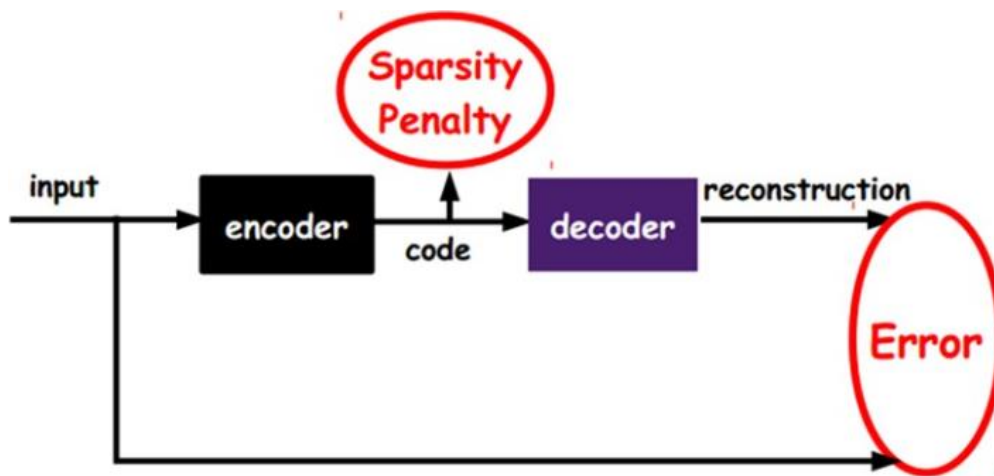
- Sparse AutoEncoder稀疏自动编码器
 - 限制得到的表达code尽量稀疏
- Denoising AutoEncoders降噪自动编码器
 - 数据存在噪声



给自编码神经网络施加一些其他的限制条件来发现输入数据中的结构。

- 限制得到的表达code尽量稀疏
 - 在AutoEncoder的基础上加上L1的Regularity限制

人脑好像也是这样的，某个输入只是刺激某些神经元，其他的大部分神经元是受到抑制的

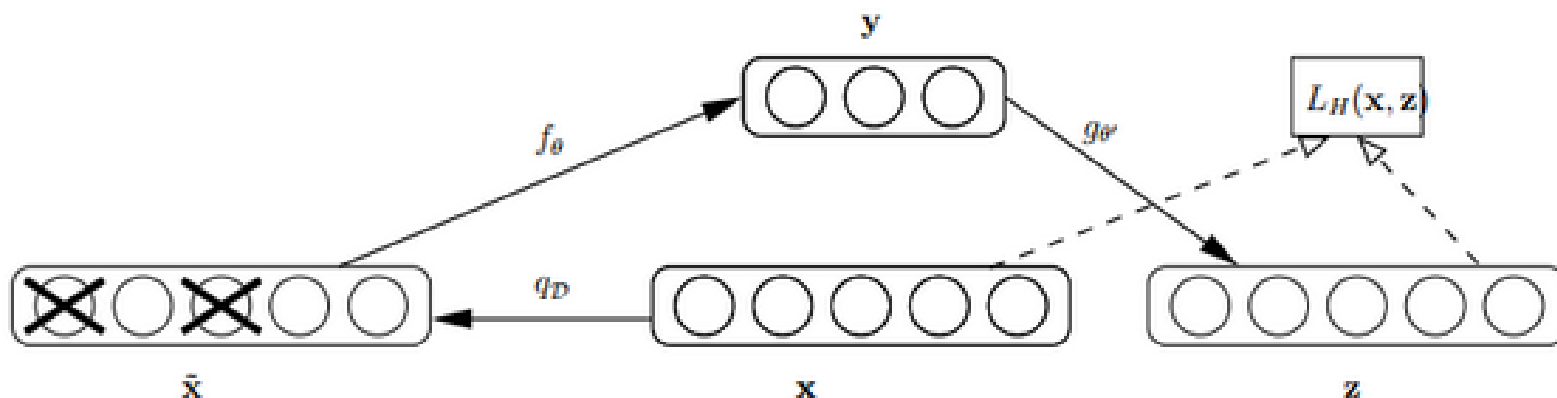


- input: X code: $h = W^T X$

- loss: $L(X; W) = \|Wh - X\|^2 + \lambda \sum_j |h_j|$

深度学习-AutoEncoder扩展

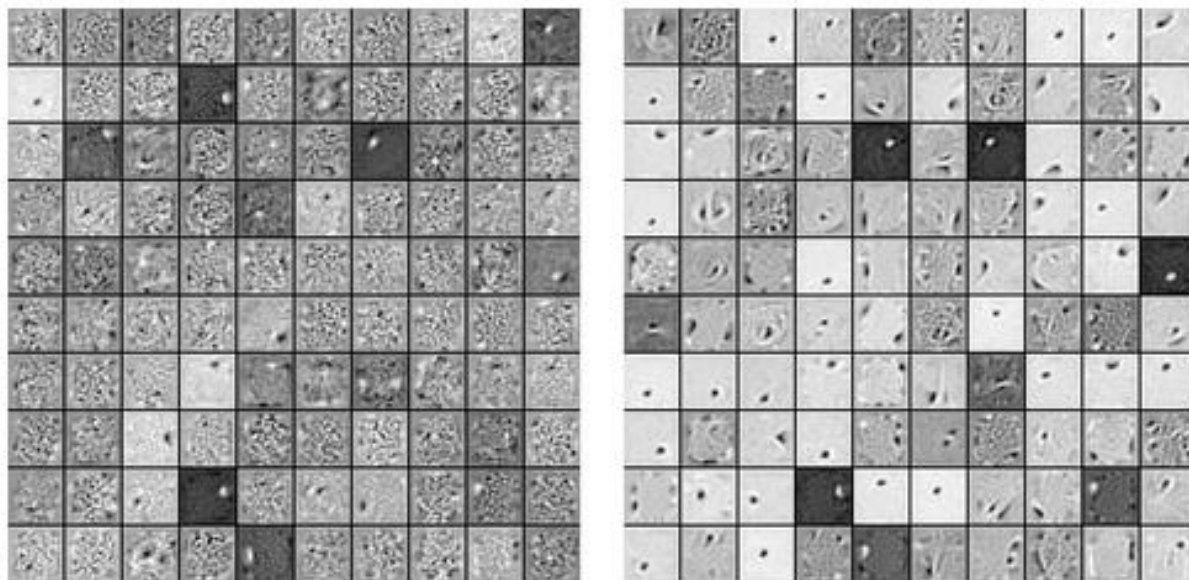
- 若训练数据中存在噪声，自动编码器必须学习去除这种噪声而获得真正的没有被噪声污染过的输入。
- 迫使编码器去学习输入信号的更加鲁棒的表达。
 - 就是以一定概率分布（通常使用二项分布）去擦除原始input矩阵，即每个值都随机置0，这样看起来部分数据的部分特征是丢失了。
 - 以这丢失的数据 \tilde{x} 去计算 y ，计算 z ，并将 z 与原始 x 做误差迭代，这样，网络就学习了这个破损（Corrupted）的数据。



■ 破损数据的作用

- 通过与非破损数据训练的对比，破损数据训练出来的Weight噪声比较小。
- 破损数据一定程度上减轻了训练数据与测试数据的代沟。

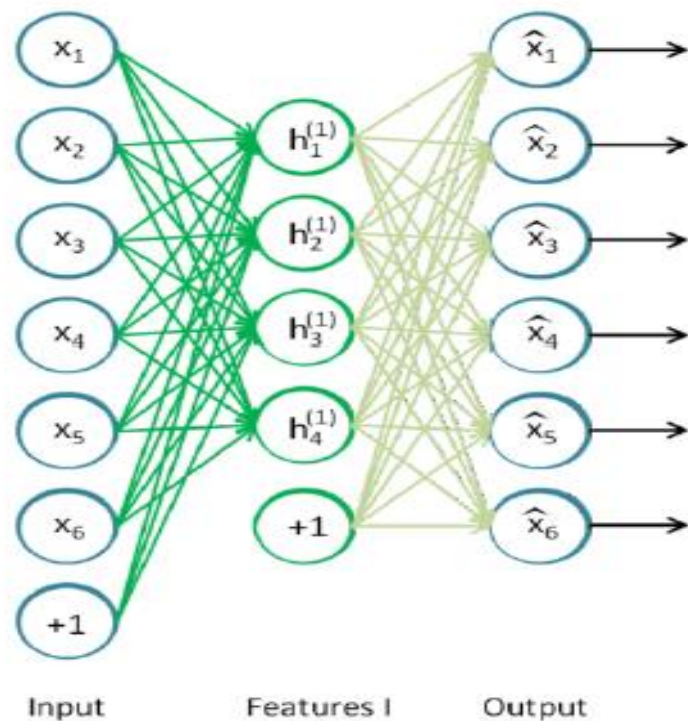
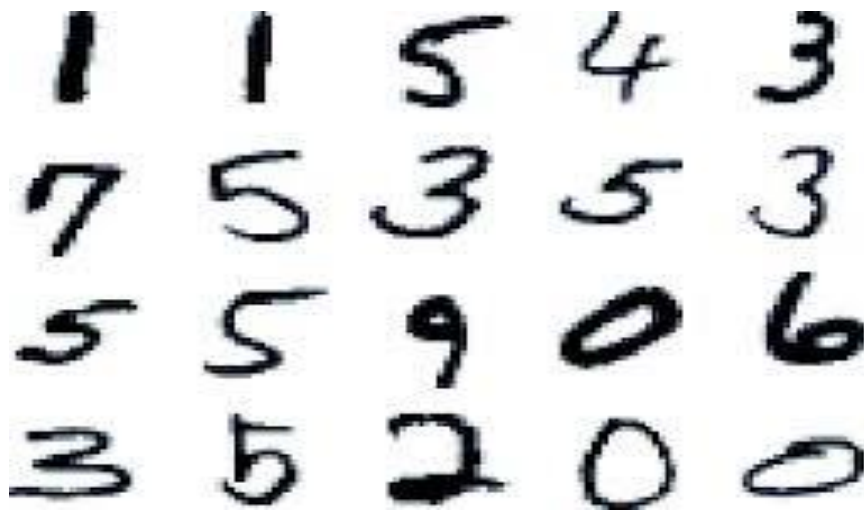
■ 这样胡乱擦除原始input真的很科学？真的没问题？



Vincent又从大脑认知角度给了解释：人类具有认知被阻挡的破损图像能力，此源于我们高等的联想记忆感受机能。我们能以多种形式去记忆（比如图像、声音），所以即便是数据破损丢失，我们也能回想起来。

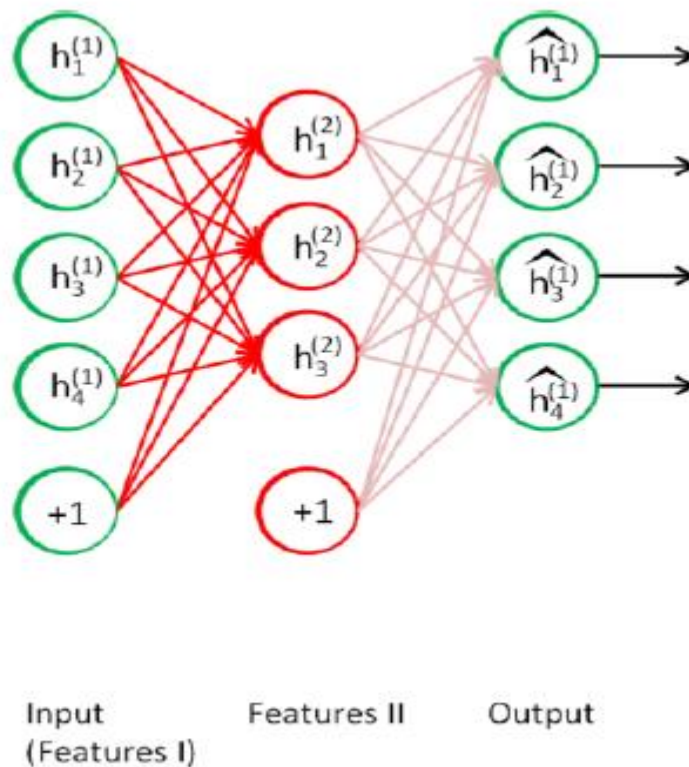
深度学习-AutoEncoder

- 训练一个包含两个隐含层的栈式自编码网络，用来进行MNIST手写数字分类
 1. 用原始输入 $x(k)$ 训练第一个自编码器，学习得到原始输入的一阶特征表示 $h^{(1)}(k)$

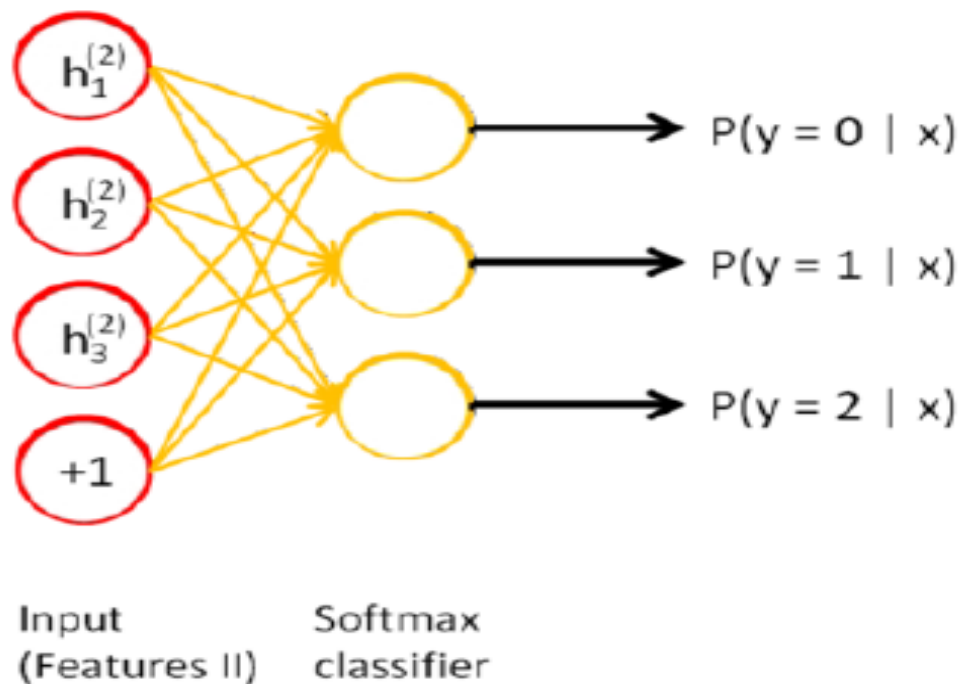


深度学习-AutoEncoder

2. 把上一层的一阶特征作为另一个稀疏自编码器的输入，使用它们来学习二阶特征 $h^{(2)}(k)$

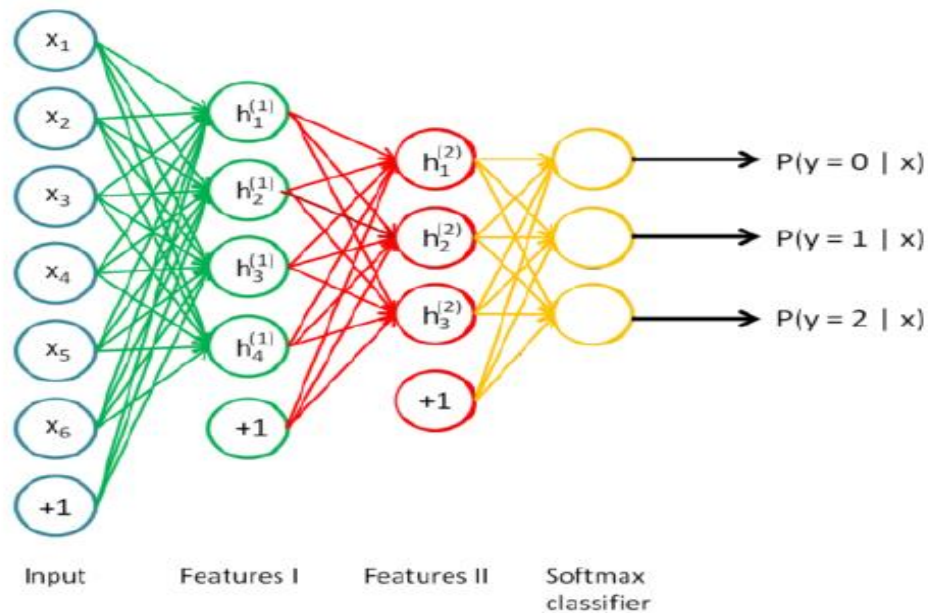
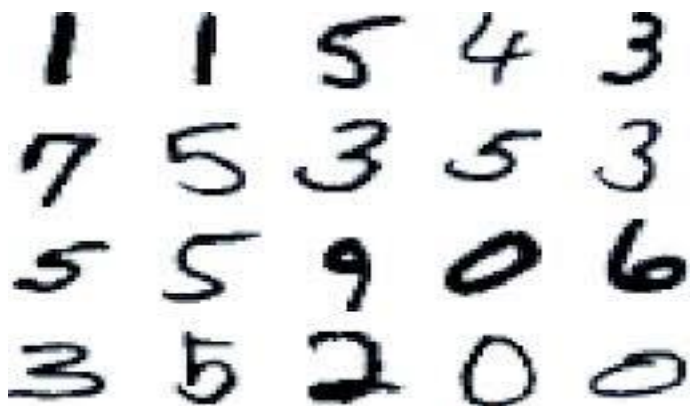


3. 将二阶特征作为softmax分类器的输入，训练得到一个能将二阶特征映射到数字标签的模型



深度学习-AutoEncoder

4. 将这三层结合起来构成一个栈式自编码网络，通过反向传播算法 (BP) 同时调整所有层的参数以改善学习结果(称为整体细调fine-tuning)



- 栈式自编码神经网络具有强大的表达能力及深度神经网络的所有优点。
- 通常能够获取到输入的“层次型分组”或者“部分-整体分解”结构。
 - 学习方式：前层的输出作为下一层输入的方式依次训练。
 - 如果网络的输入数据是图像，网络的第一层会学习如何去识别边，第二层一般会学习如何去组合边，从而构成轮廓、角等。更高层会学习如何去组合更形象且有意义的特征。
 - 如果输入数据集包含人脸图像，更高层会学习如何识别或组合眼睛、鼻子、嘴等人脸器官。

深度学习-卷积神经网络 (CNN)

■ 20世纪60年代，Hubel和Wiesel研究猫脑皮层

- 用于局部敏感和方向选择的神经元，其独特的网络结构可以有效地降低反馈神经网络的复杂性

■ 卷积神经网络是一种特殊的深层神经网络模型

- 它的神经元间的连接是非全连接的
- 同一层中某些神经元之间的连接的权重是共享的（即相同的）。

Hubel-Wiesel结构

■ 基于猫的初级视皮层(VI区)的研究。

- 简单细胞
- 复杂细胞

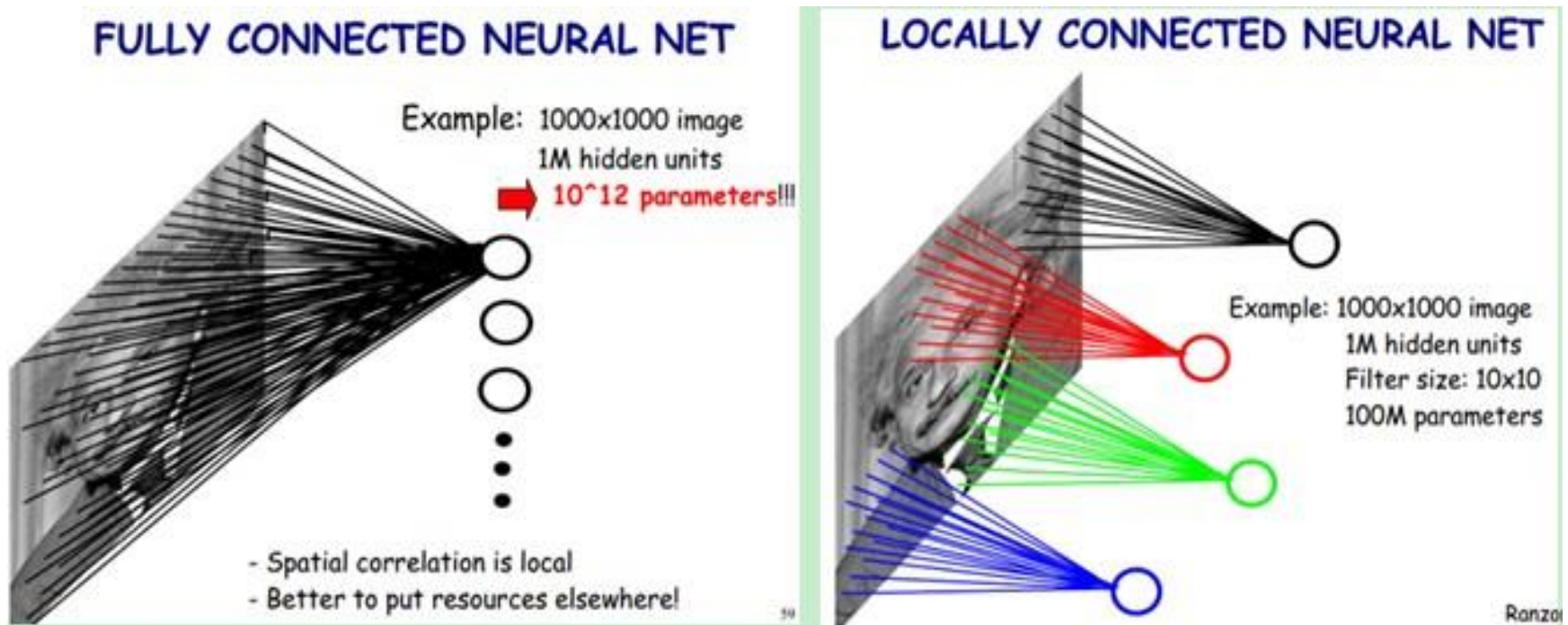
■ 两层神经网络模拟初级视皮层中的简单细胞和复杂细胞

- 每层的神经元被组织成二维平面
- “简单细胞”层提取其输入中的局部特征
- “复杂细胞”层组合“简单细胞”层中相应的子区域，使得整个网络对局部变换具有一定的不变性。

深度学习-卷积神经网络 (CNN)

■ 局部感知野

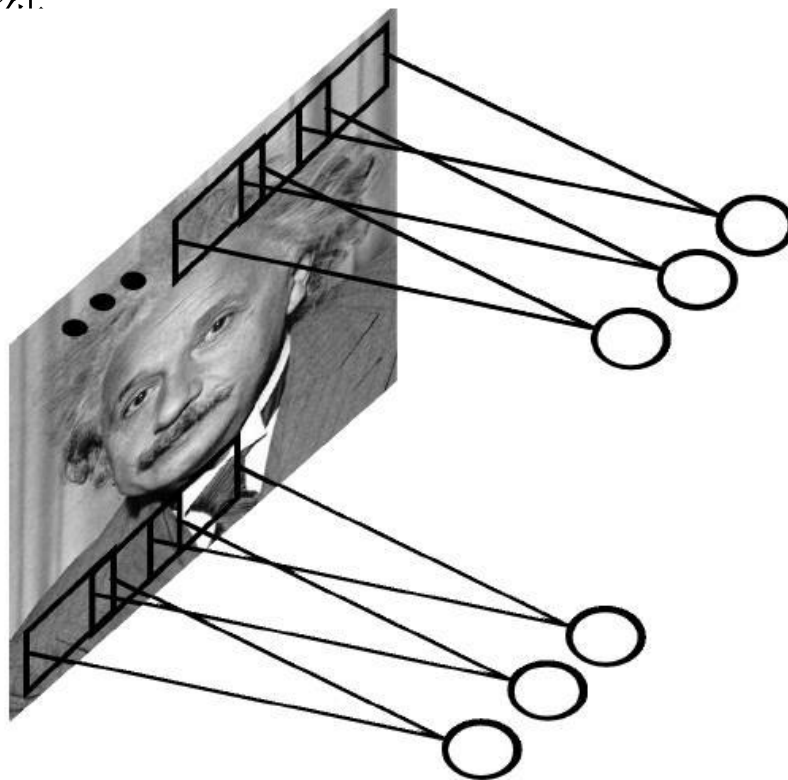
- 图像的空间联系也是局部的像素联系较为紧密，而距离较远的像素相关性则较弱。
- 减少了需要训练的权值数目



深度学习-卷积神经网络 (CNN)

■ 参数共享

- 图像的一部分的统计特性与其他部分是一样的。
- 在输入的不同位置检测同一种特征
- 平移不变性



深度学习-卷积神经网络 (CNN)

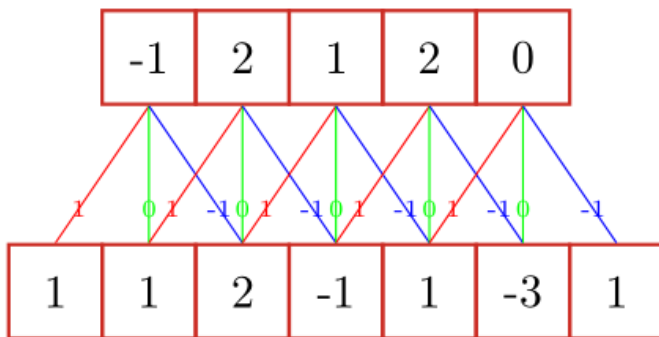
- 卷积经常用在信号处理中，用于计算信号的延迟累积。
- 假设一个信号发生器每个时刻 t 产生一个信号 x_t ，其信息的衰减率为 w_k ，即在 $k-1$ 个时间步长后，信息为原来的 w_k 倍
 - 假设 $w_1 = 1, w_2 = 1/2, w_3 = 1/4$
- 时刻 t 收到的信号 y_t 为当前时刻产生的信息和以前时刻延迟信息的叠加

$$y_t = 1 \times x_t + 1/2 \times x_{t-1} + 1/4 \times x_{t-2}$$

$$= w_1 \times x_t + w_2 \times x_{t-1} + w_3 \times x_{t-2}$$

$$= \sum_{k=1}^3 w_k \cdot x_{t-k+1}$$

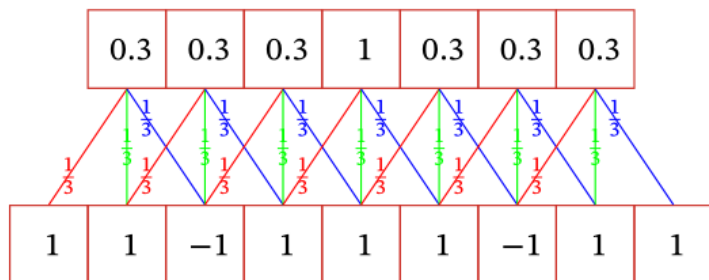
Filter: [1,0,-1]



滤波器 (filter) 或卷积核
(convolution kernel)

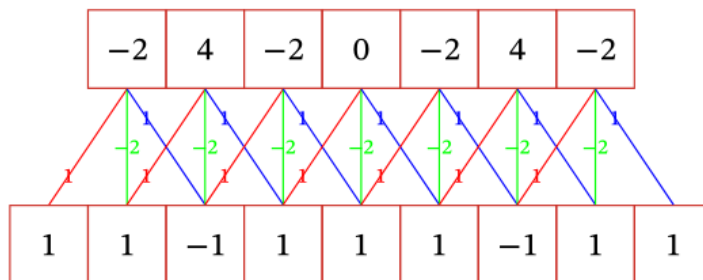
深度学习-卷积神经网络 (CNN)

- 不同的滤波器来提取信号序列中的不同特征



(a) 滤波器 $[1/3, 1/3, 1/3]$

低频信息

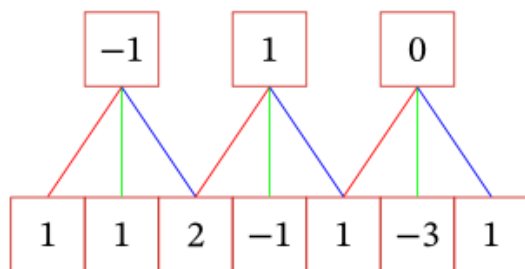


(b) 滤波器 $[1, -2, 1]$

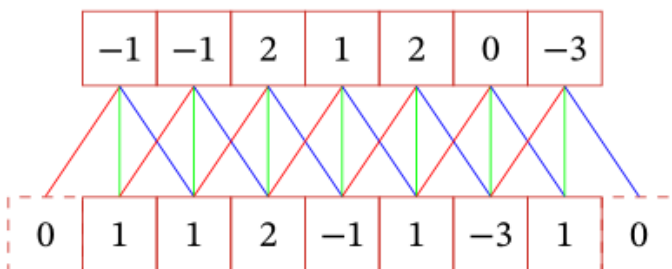
高频信息

深度学习-卷积神经网络 (CNN)

▶ 引入滤波器的滑动步长 S 和零填充 P



(a) 步长 $S = 2$



(b) 零填充 $P = 1$

▶ 卷积的结果按输出长度不同可以分为三类 (d 为神经元数, m 为卷积大小)：

- ▶ **窄卷积**：步长 $T=1$ ，两端不补零 $P=0$ ，卷积后输出长度为 $d - m + 1$
 - ▶ **宽卷积**：步长 $T=1$ ，两端补零 $P=m-1$ ，卷积后输出长度 $d + m - 1$
 - ▶ **等宽卷积**：步长 $T=1$ ，两端补零 $P=(m-1)/2$ ，卷积后输出长度 d
-
- ▶ 在早期的文献中，卷积一般默认为**窄卷积**。
 - ▶ 而目前的文献中，卷积一般默认为**等宽卷积**。

深度学习-卷积神经网络 (CNN)

- 在图像处理中，图像是以二维矩阵的形式输入到神经网络中，因此我们需要二维卷积。

一个输入信息 \mathbf{X} 和滤波器 \mathbf{W} 的二维卷积定义为

$$\mathbf{Y} = \mathbf{W} * \mathbf{X},$$

$$y_{ij} = \sum_{u=1}^U \sum_{v=1}^V w_{uv} x_{i-u+1, j-v+1}.$$

1	1	1	1	1
-1	0	-3	0	1
2	1	1	-1	0
0	-1	1	2	1
1	2	1	1	1

x-1

x0

x0

x0

x0

x0

x0

x0

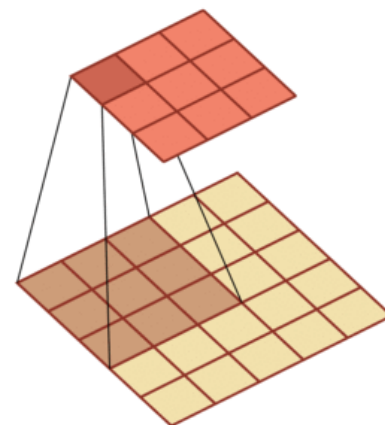
x0

x1

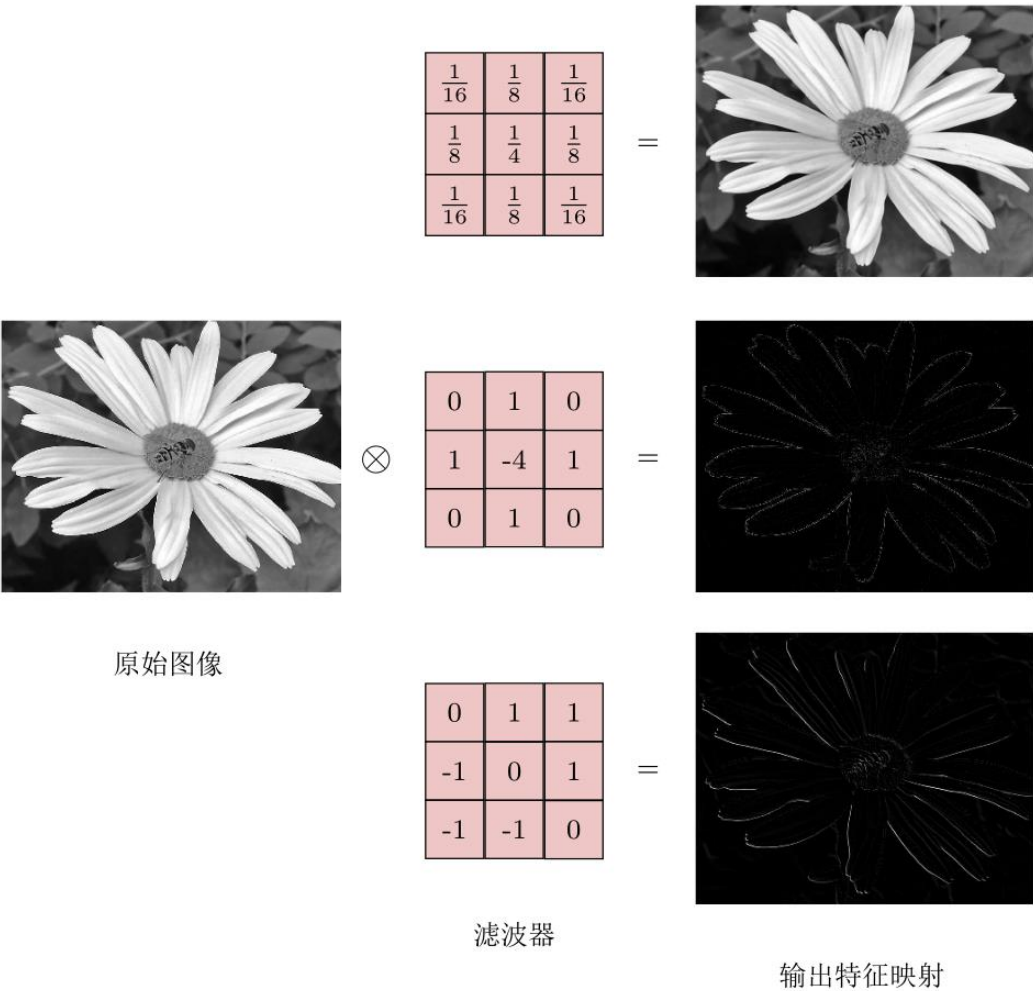
1	0	0
0	0	0
0	0	-1

=

0	-2	-1
2	2	4
-1	0	0

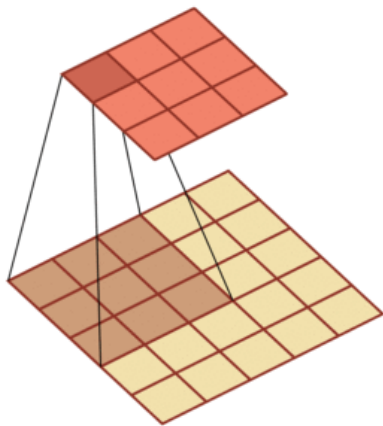


深度学习-卷积神经网络 (CNN)

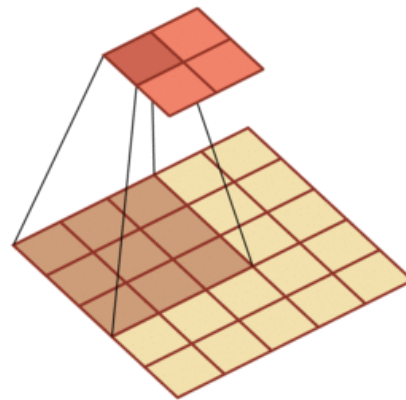


深度学习-卷积神经网络 (CNN)

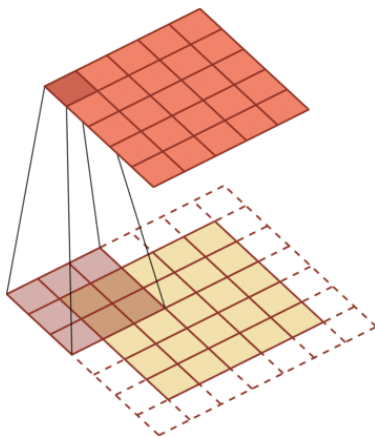
二维卷积



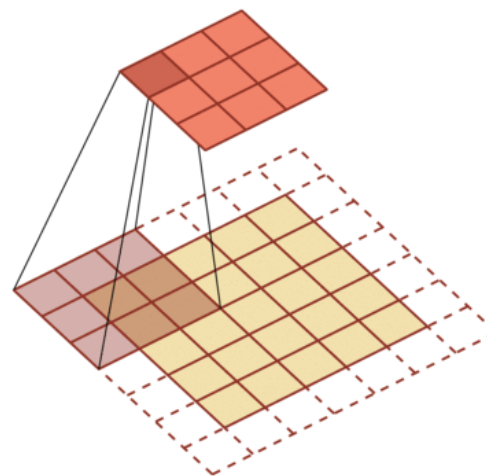
步长1，零填充0



步长2，零填充0



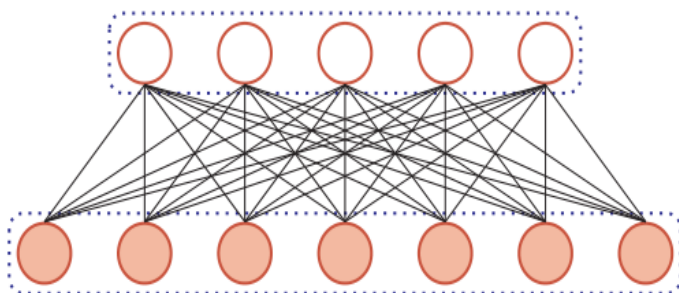
步长1，零填充1



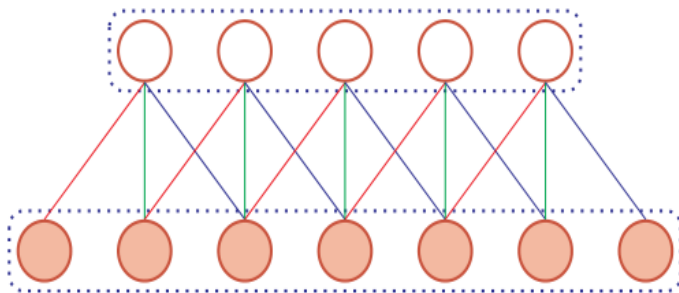
步长2，零填充1

深度学习-卷积神经网络 (CNN)

► 用卷积层代替全连接层



(a) 全连接层




(b) 卷积层

深度学习-卷积神经网络 (CNN)

- ▶ 计算卷积需要进行卷积核翻转。
- ▶ 卷积操作的目标：提取特征。

翻转是不必要的！

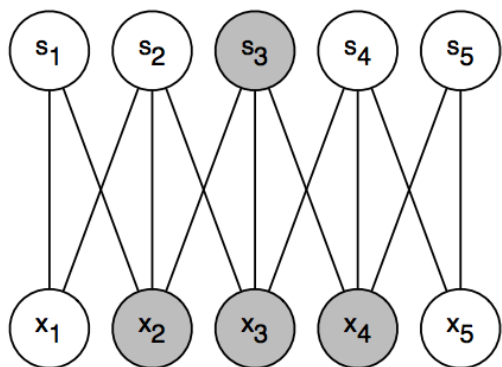
- ▶ 互相关

$$y_{ij} = \sum_{u=1}^m \sum_{v=1}^n w_{uv} \cdot x_{i+u-1, j+v-1}$$
$$v_{ij} = \sum_{u=1}^U \sum_{v=1}^V w_{uv} x_{i-u+1, j-v+1}$$


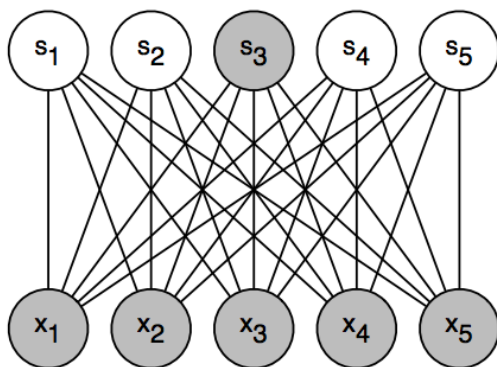
除非特别声明，卷积一般指“互相关”。

深度学习-卷积神经网络 (CNN)

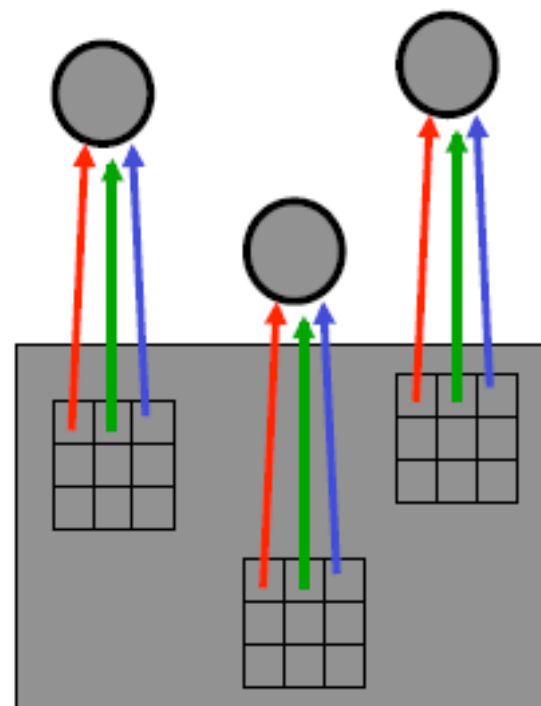
- 稀疏连接
- 参数共享



sparse

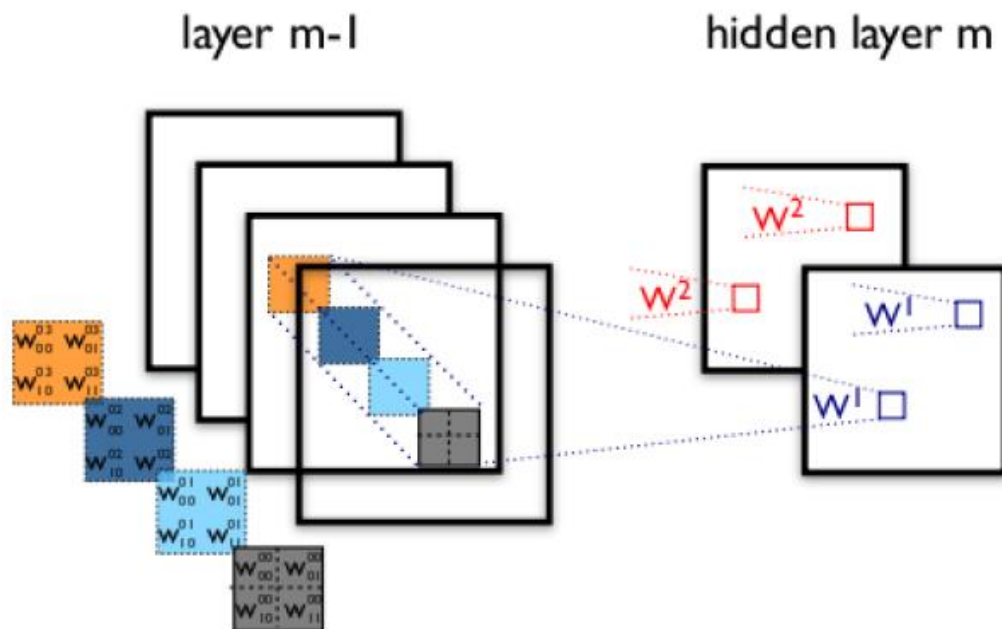
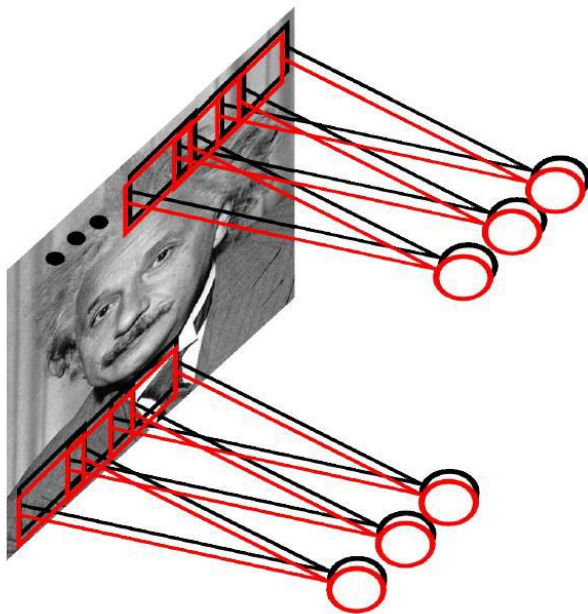


dense



卷积神经网络 (CNN)

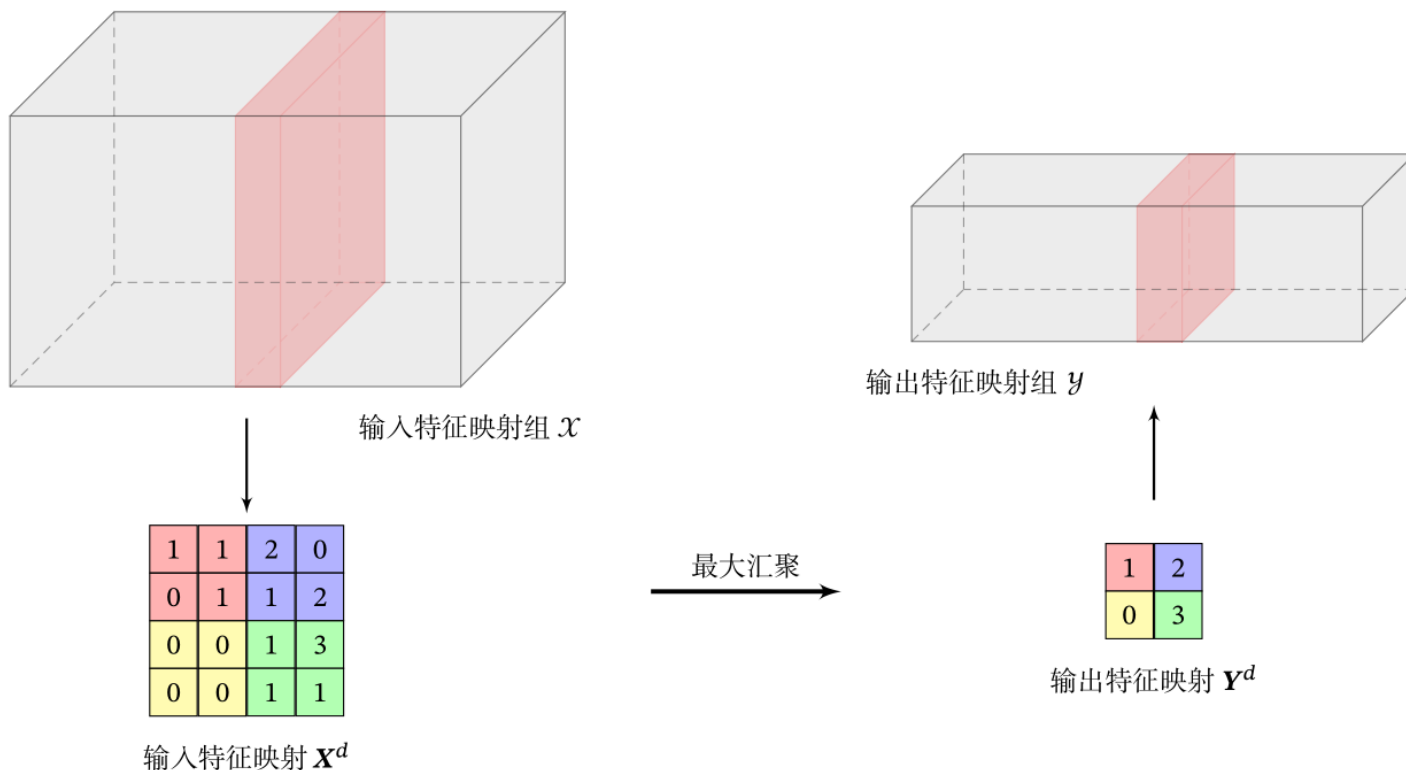
- 每个卷积核都会将图像生成为另一幅图像。
 - 两个卷积核就可以将生成两幅图像，这两幅图像可以看做是一张图像的不同通道。



由4个通道卷积得到2个通道的过程

卷积神经网络 (CNN)

- 卷积层虽然可以显著减少连接的个数，但是每一个特征映射的神经元个数并没有显著减少。



卷积神经网络 (CNN)

- 通过卷积获得了特征之后，下一步利用这些特征去做分类。

- 使用卷积时是利用了图像的“静态”特征
- Pooling, 对不同位置的特征进行聚合统计

- 子采样

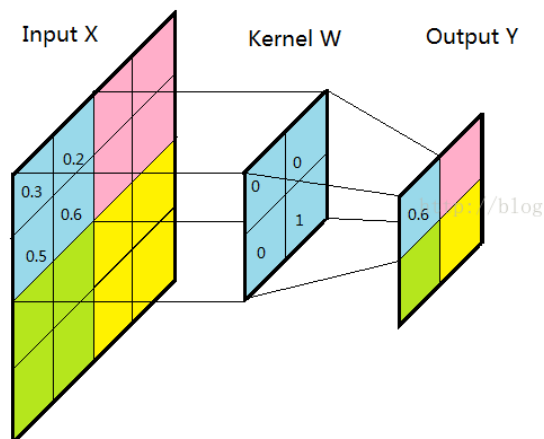
- Mean pooling

$$\frac{1}{s^2} \sum x_i$$

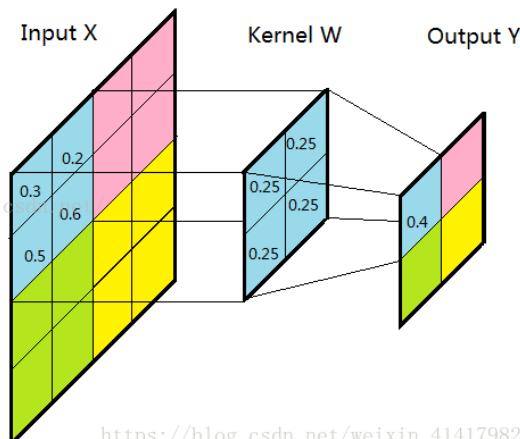
- Max pooling $\max\{x_i\}$

- L2 pooling $\sqrt{\frac{1}{s^2} \sum x_i^2}$

max-pooling

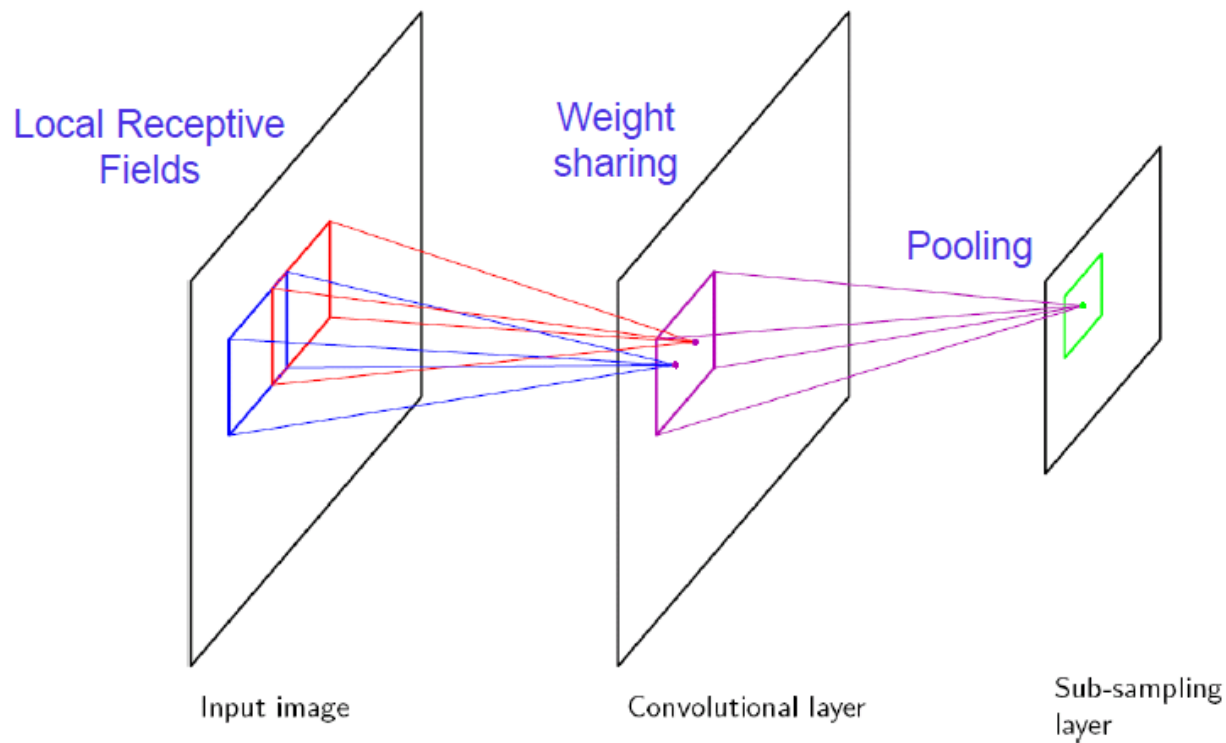


mean-polling



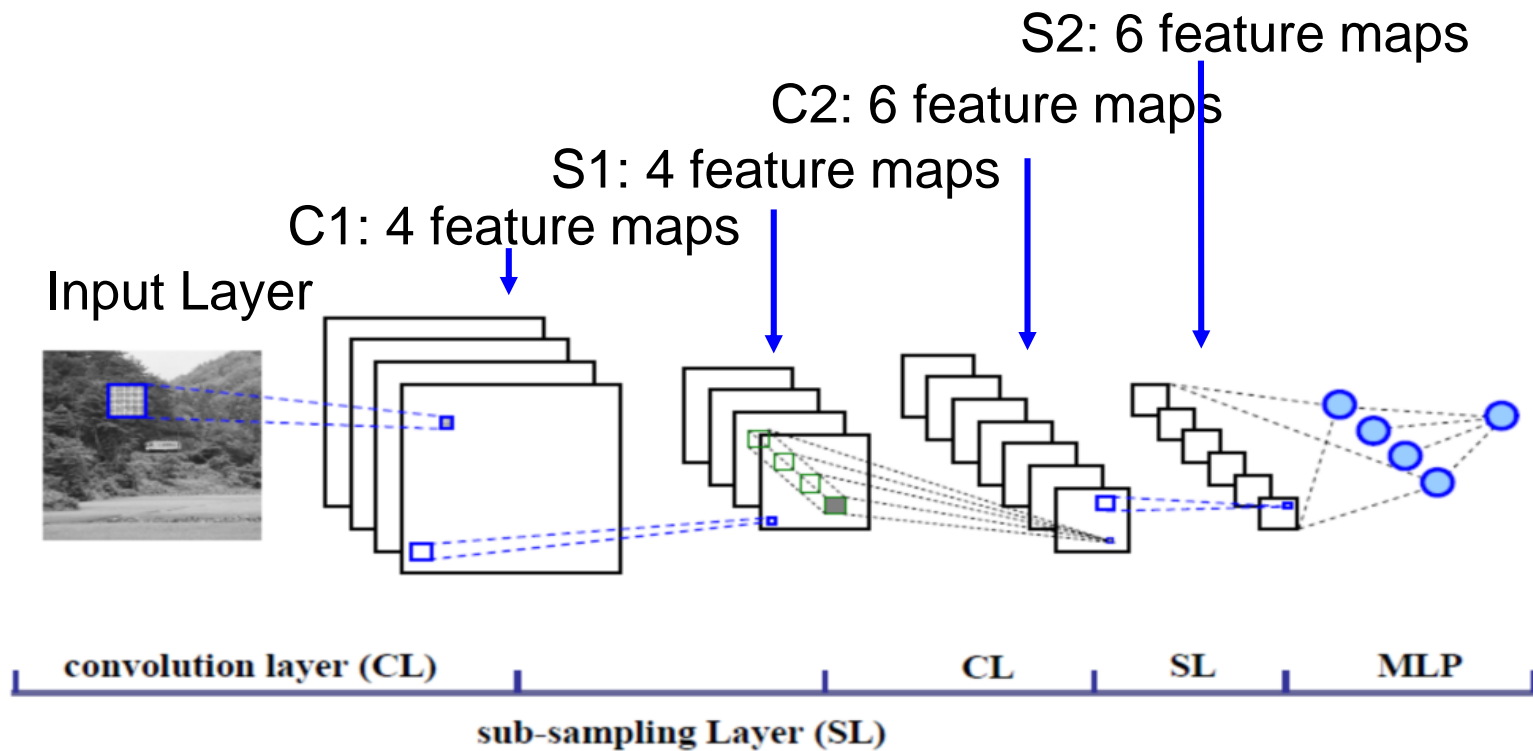
卷积神经网络 (CNN)

- 卷积层
- 子采样层



卷积神经网络 (CNN)

- 卷积神经网络是一个多层的神经网络
 - 每层由多个二维平面组成
 - 每个平面由多个独立神经元组成



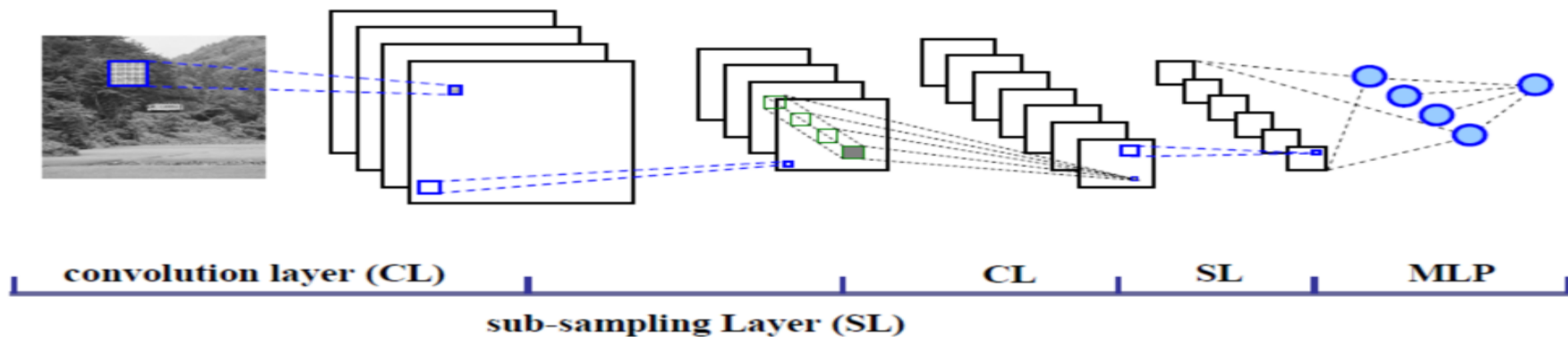
CNN的经典应用

■ 代价函数

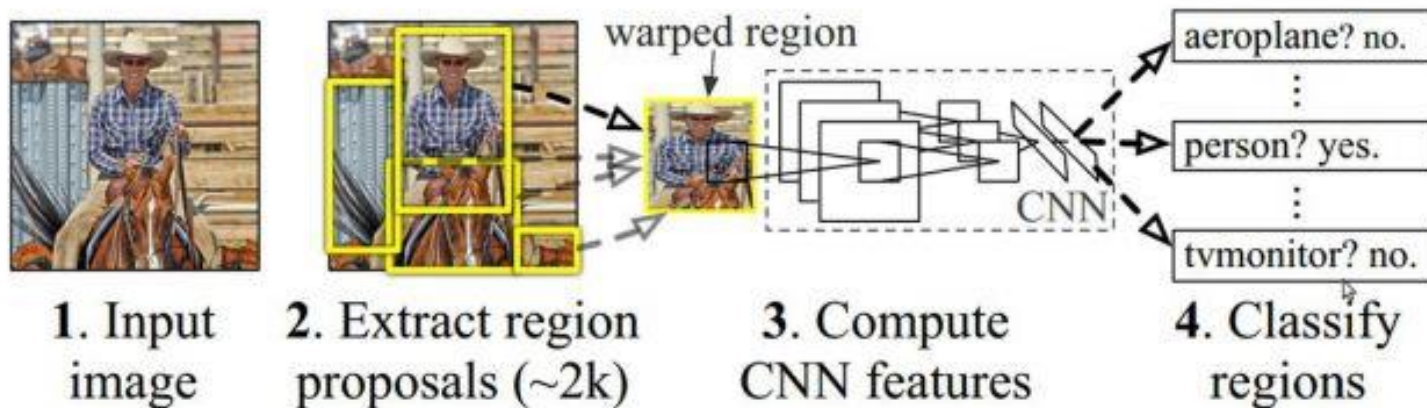
- 最小化平方误差(MSE), 最小化相对熵(Relative Entropy)

■ 反向传播主要考虑三个方面:

- 输出层, 代价函数的确定及求导
- Pooling, 数据的下采样及残差的上采样
- 卷积层, 数据的卷积运算及残差的反卷积运算



R-CNN

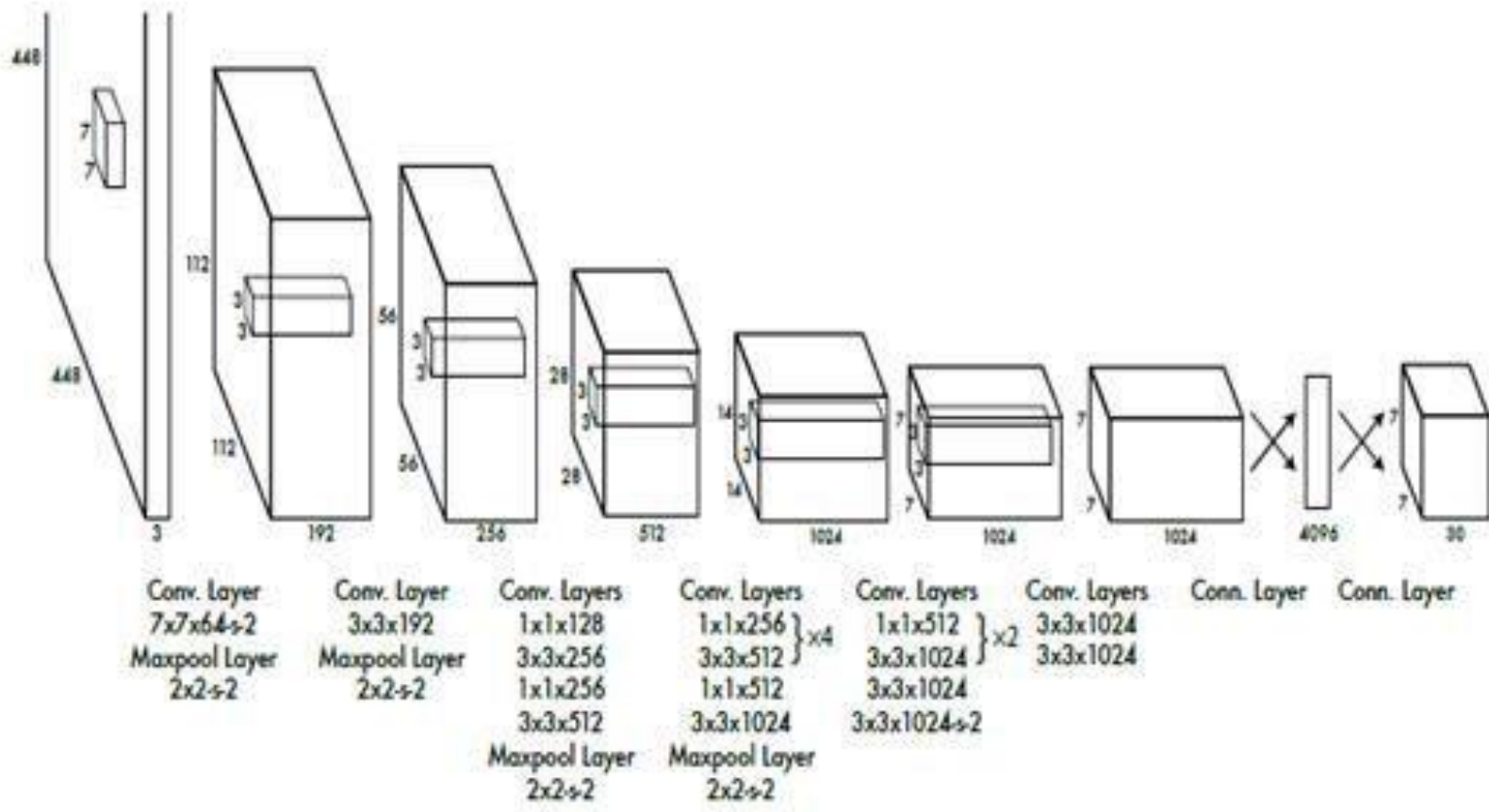


General framework: Region proposal + DCNN based region classification

Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation, R. Girshick, J. Donahue, T. Darrell, J. Malik, in CVPR 2014

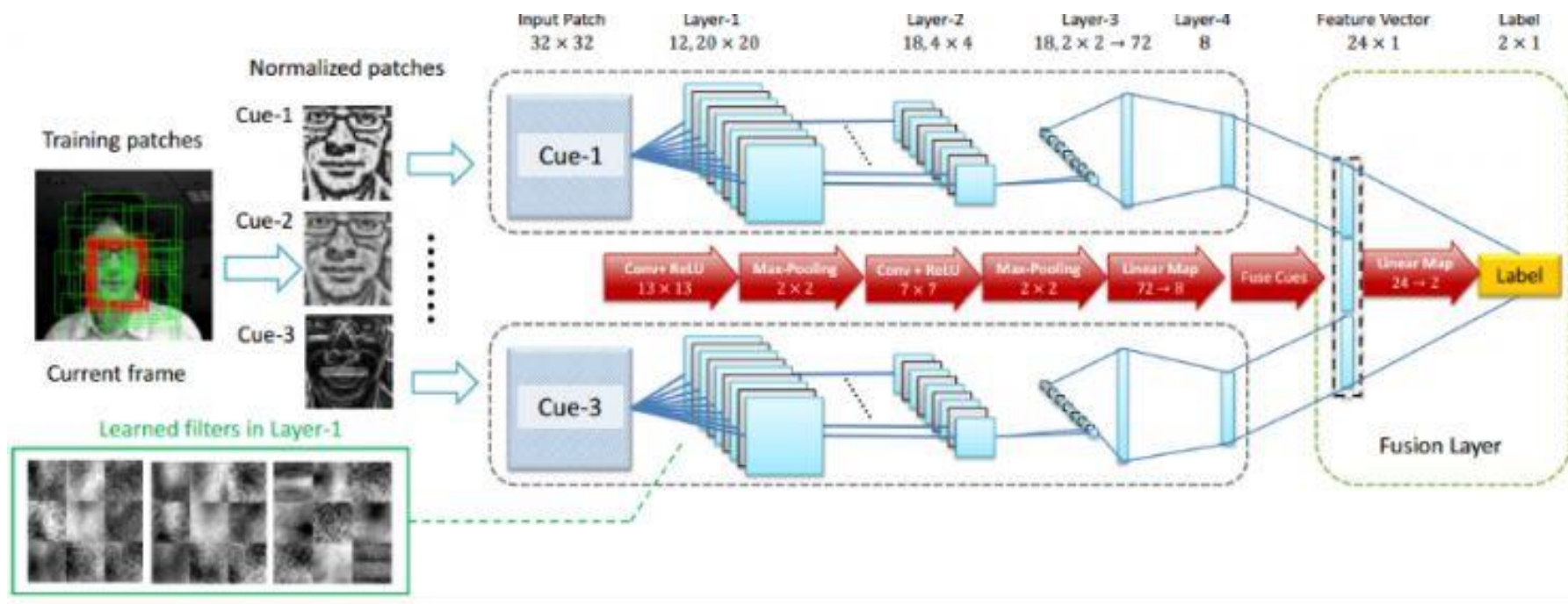
深度学习在物体检测方面也取得了非常好的成果。2014年的Region CNN算法，基本思想是首先用一个非深度的方法，在图像中提取可能是物体的图形块，然后深度学习算法根据这些图像块，判断属性和一个具体物体的位置。

深度学习-物体检测



FACEBOOK提出来的YOLO网络，也是进行物体检测，最快达到每秒钟155帧，达到了完全实时。它让一整张图像进入到神经网络，让神经网络自己判断这物体可能在哪里，可能是什么。但它缩减了可能图像块的个数，从原来Faster R-CNN的2000多个缩减缩减到了98个。

深度学习-物体跟踪



跟踪，就是在视频里面第一帧时锁定感兴趣的物体，让计算机跟着走，不管怎么旋转晃动，甚至躲在树丛后面也要跟踪。