

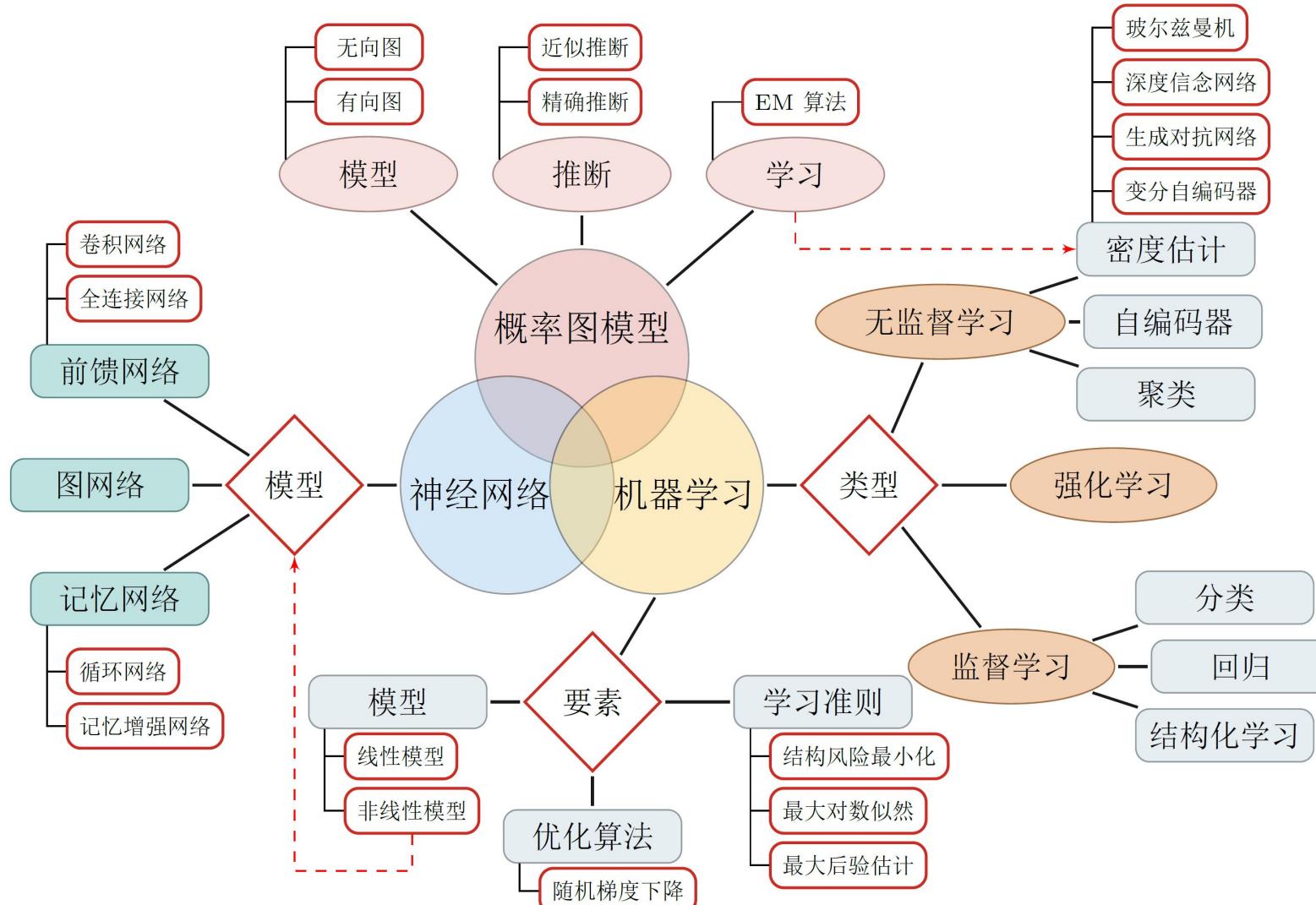
《神经网络与深度学习》



第六章：基础模型

<https://nndl.github.io/>

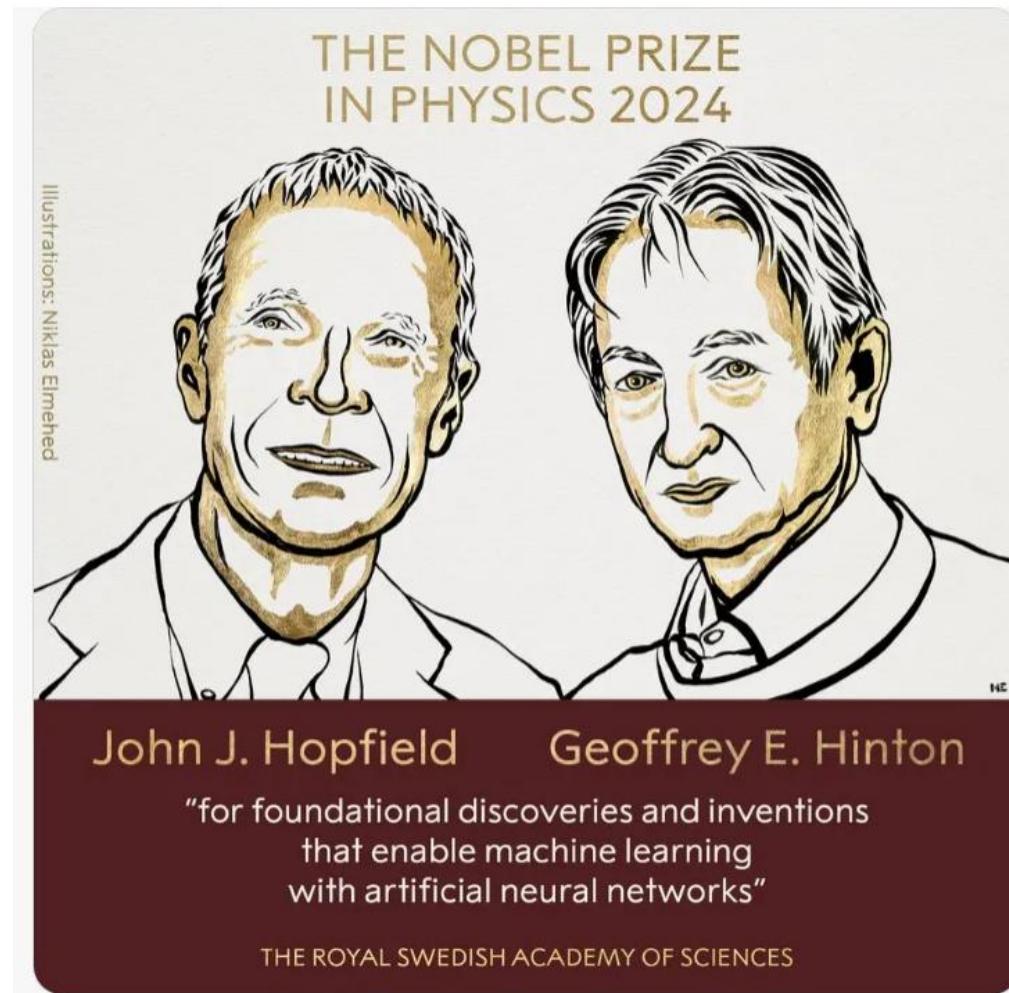
深度神经网络





6.1 引言

深度学习革命



神经网络发展史

- ▶ 神经网络的发展大致经过五个阶段
- ▶ 第一阶段：模型提出
 - ▶ 在1943年，心理学家Warren McCulloch和数学家Walter Pitts最早描述了一种理想化的人工神经网络，并构建了一种基于简单逻辑运算的计算机制。他们提出的神经网络模型称为MP模型
 - ▶ 阿兰·图灵在1948年的论文中描述了一种“B型图灵机”
 - ▶ 1951年，McCulloch和Pitts的学生Marvin Minsky建造了第一台神经网络机，称为SNARC
 - ▶ Rosenblatt [1958]最早提出可以模拟人类感知能力的神经网络模型，并称之为感知器（Perceptron），并提出了一种接近于人类学习过程（迭代、试错）的学习算法

神经网络发展史

► 第二阶段：冰河期

- 1969年，Marvin Minsky出版《感知器》一书，书中论断直接将神经网络打入冷宫，导致神经网络十多年的“冰河期”。他们发现了神经网络的两个关键问题
 - 基本感知器无法处理异或回路
 - 电脑没有足够的能力来处理大型神经网络所需要的很长的计算时间
- 1974年，哈佛大学的Paul Webos发明反向传播算法，但当时未受到应有的重视
- 1980年，Kunihiko Fukushima（福岛邦彦）提出了一种带卷积和子采样操作的多层神经网络：新知机（Neocognitron）

神经网络发展史

► 第三阶段：反向传播算法引起的复兴

- 1983年，物理学家John Hopfield对神经网络引入能量函数的概念，并提出了用于联想记忆和优化计算的网络（称为Hopfield网络），在旅行商问题上获得当时最好结果，引起轰动
 - 1984年，Geoffrey Hinton提出一种随机化版本的Hopfield网络，即玻尔兹曼机
 - 1986年，David Rumelhart和James McClelland对于联结主义在计算机模拟神经活动中的应用提供了全面的论述，并重新发明了反向传播算法
 - 1986年，Geoffrey Hinton等人将引入反向传播算法到多层感知器
 - 1989年，LeCun等人将反向传播算法引入了卷积神经网络，并在手写体数字识别上取得了很大的成功
-

神经网络发展史

► 第四阶段：流行度降低

- 在20世纪90年代中期，统计学习理论和以支持向量机为代表的机器学习模型开始兴起
- 相比之下，神经网络的理论基础不清晰、优化困难、可解释性差等缺点更加凸显，神经网络的研究又一次陷入低潮

神经网络发展史

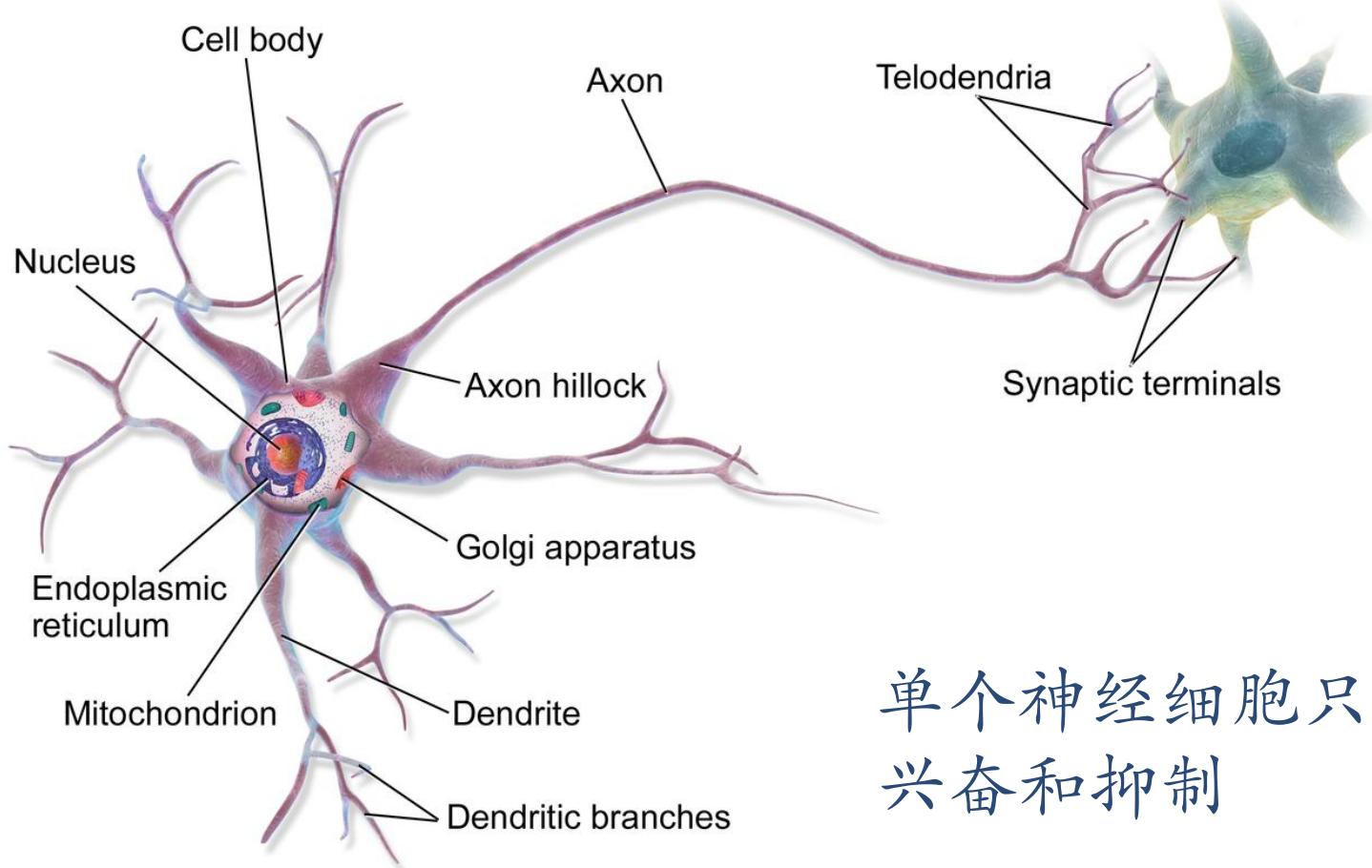
► 第五阶段：深度学习的崛起

- 2006年，Hinton等人发现多层前馈神经网络可以先通过逐层预训练，再用反向传播算法进行精调的方式进行有效学习
- 深度神经网络在语音识别和图像分类等任务上的巨大成功
- 2013年，AlexNet：第一个现代深度卷积网络模型，是深度学习技术在图像分类上取得真正突破的开端
- AlexNet不用预训练和逐层训练，首次使用了很多现代深度网络的技术

- 随着大规模并行计算以及GPU设备的普及，计算机的计算能力得以大幅提高，同时可供机器学习的数据规模也越来越大，使得计算机已经可以训练大规模的人工神经网络

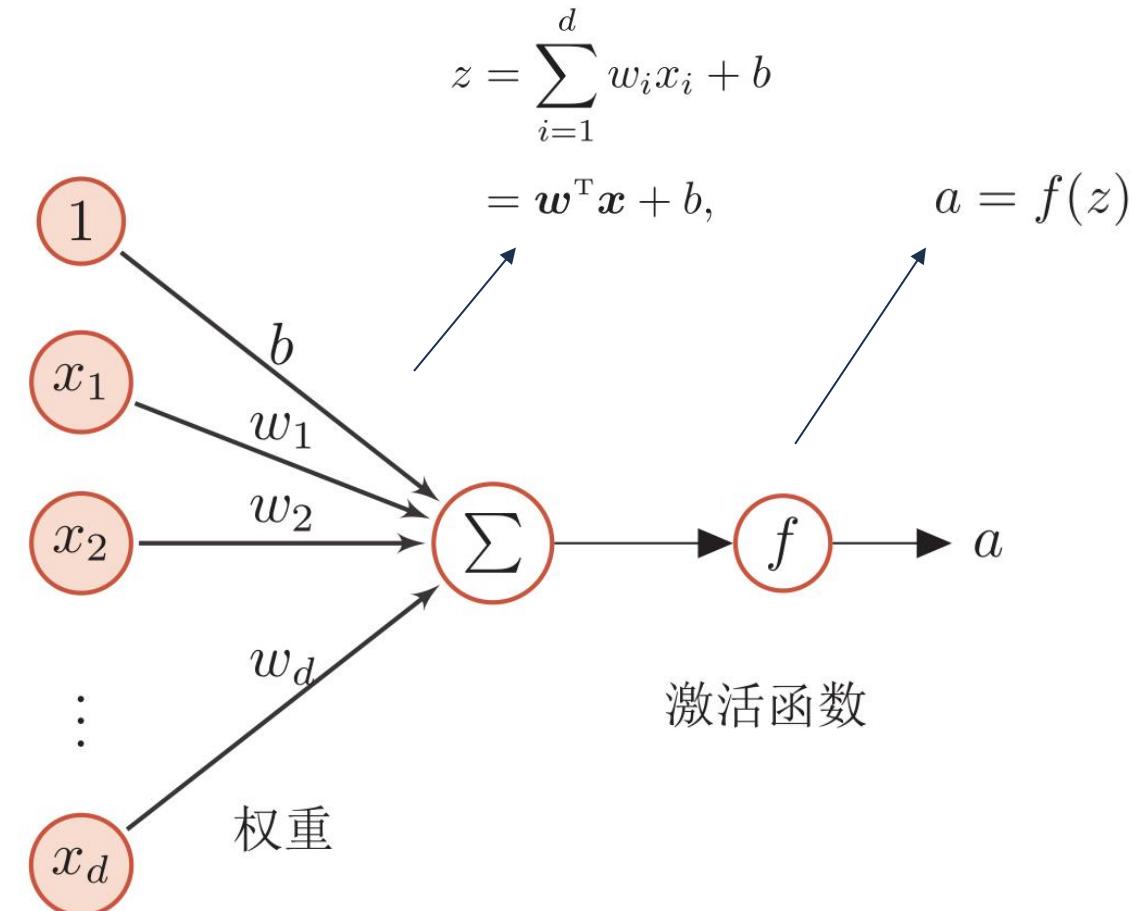
生物神经元

[video: structure of brain](#)



单个神经细胞只有两种状态：
兴奋和抑制

人工神经元



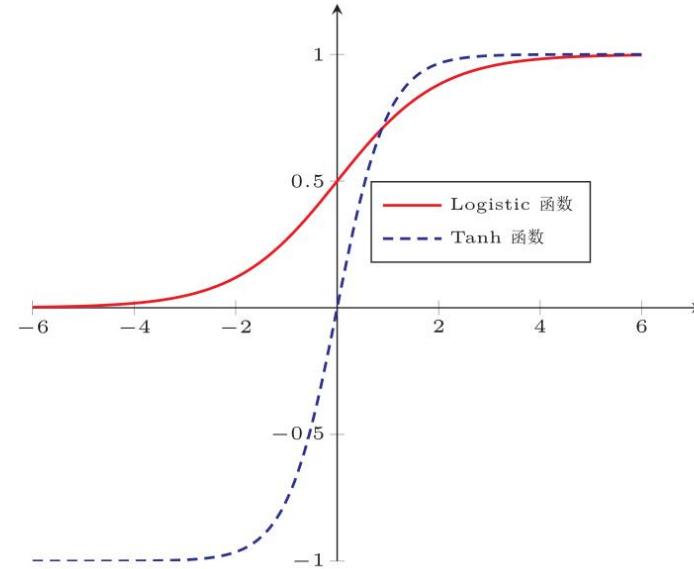
激活函数的性质

- ▶ 连续并可导（允许少数点上不可导）的非线性函数
 - ▶ 可导的激活函数可以直接利用数值优化的方法来学习网络参数
-
- ▶ 激活函数及其导函数要尽可能的简单
 - ▶ 有利于提高网络计算效率
-
- ▶ 激活函数的导函数的值域要在一个合适的区间内
 - ▶ 不能太大也不能太小，否则会影响训练的效率和稳定性

常见激活函数

$$\sigma(x) = \frac{1}{1 + \exp(-x)}$$

$$\tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}$$



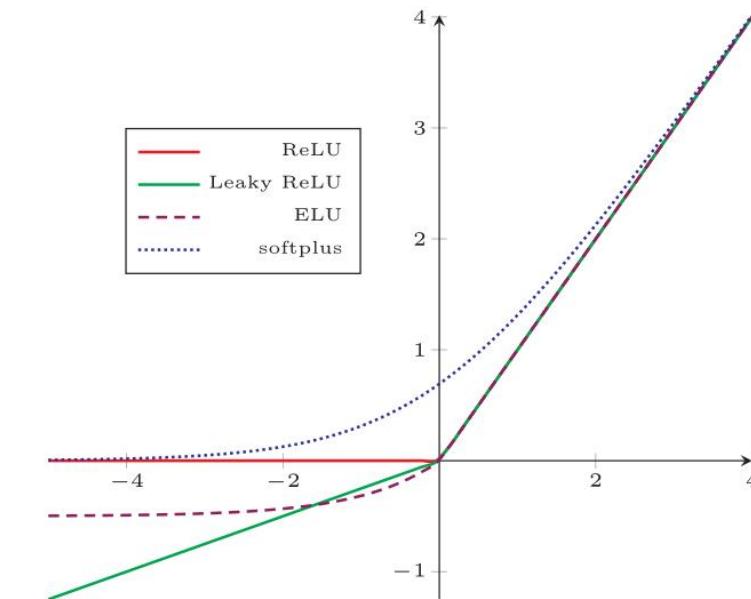
- 性质：
- 饱和函数
- Tanh函数是零中心化的，而logistic函数的输出恒大于0

常见激活函数

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

$$\text{LeakyReLU}(x) = \begin{cases} x & \text{if } x > 0 \\ \gamma x & \text{if } x \leq 0 \end{cases} = \max(0, x) + \gamma \min(0, x)$$

$$\text{PReLU}_i(x) = \begin{cases} x & \text{if } x > 0 \\ \gamma_i x & \text{if } x \leq 0 \end{cases} = \max(0, x) + \gamma_i \min(0, x)$$



$$\text{ELU}(x) = \begin{cases} x & \text{if } x > 0 \\ \gamma(\exp(x) - 1) & \text{if } x \leq 0 \end{cases} = \max(0, x) + \min(0, \gamma(\exp(x) - 1))$$

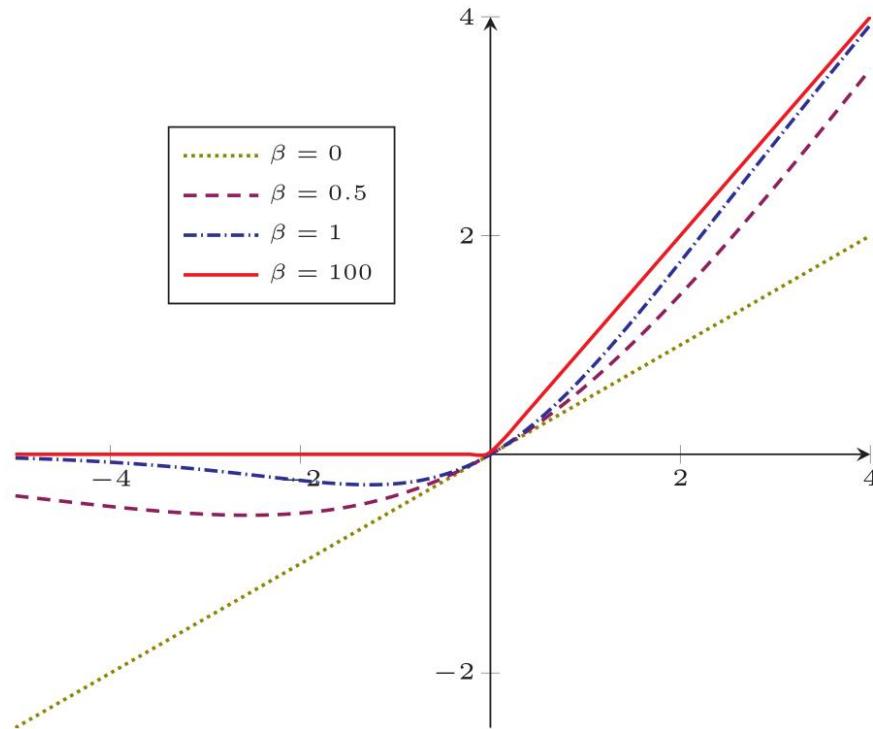
- ▶ 计算上更加高效
- ▶ 生物学合理性
 - ▶ 单侧抑制、宽兴奋边界
- ▶ 在一定程度上缓解梯度消失问题

$$\text{softplus}(x) = \log(1 + \exp(x))$$

常见激活函数

Swish函数 $\text{swish}(x) = x\sigma(\beta x)$

$$\sigma(x) = \frac{1}{1 + \exp(-x)}$$



常见激活函数

► 高斯误差线性单元 (Gaussian Error Linear Unit, GELU)

$$\text{GELU}(x) = xP(X \leq x)$$

- 其中 $P(X \leq x)$ 是高斯分布 $N(\mu, \sigma^2)$ 的累积分布函数，其中 μ, σ 为超参数，一般设 $\mu = 0, \sigma = 1$ 即可
- 由于高斯分布的累积分布函数为 S 型函数，因此 GELU 可以用 Tanh 函数或 Logistic 函数来近似

$$\text{GELU}(x) \approx 0.5x \left(1 + \tanh \left(\sqrt{\frac{2}{\pi}} (x + 0.044715x^3) \right) \right)$$

或 $\text{GELU}(x) \approx x\sigma(1.702x).$

常见激活函数及其导数

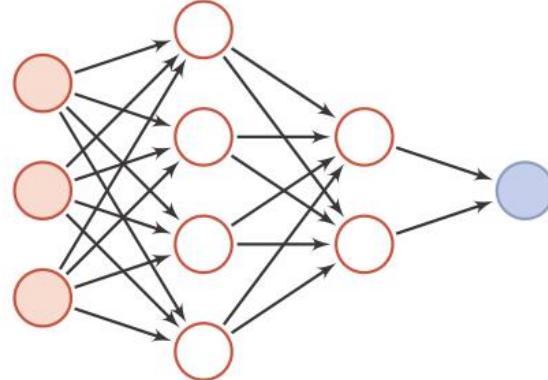
激活函数	函数	导数
Logistic 函数	$f(x) = \frac{1}{1+\exp(-x)}$	$f'(x) = f(x)(1-f(x))$
Tanh 函数	$f(x) = \frac{\exp(x)-\exp(-x)}{\exp(x)+\exp(-x)}$	$f'(x) = 1-f(x)^2$
ReLU 函数	$f(x) = \max(0, x)$	$f'(x) = I(x > 0)$
ELU 函数	$f(x) = \max(0, x) + \min(0, \gamma(\exp(x) - 1))$	$f'(x) = I(x > 0) + I(x \leq 0) \cdot \gamma \exp(x)$
SoftPlus 函数	$f(x) = \log(1 + \exp(x))$	$f'(x) = \frac{1}{1+\exp(-x)}$

人工神经网络

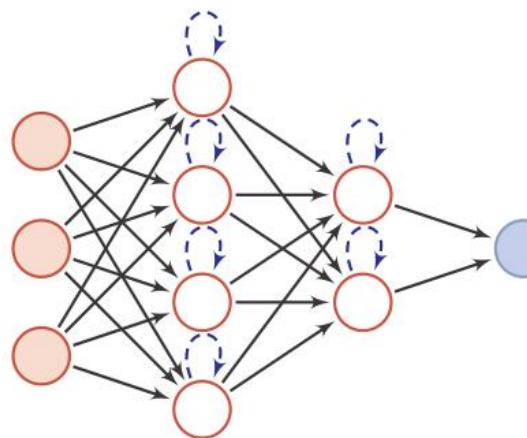
- ▶ 人工神经网络主要由大量的神经元以及它们之间的有向连接构成。因此考虑三方面
- ▶ 神经元的激活规则
 - ▶ 主要是指神经元输入到输出之间的映射关系，一般为非线性函数
- ▶ 网络的拓扑结构
 - ▶ 不同神经元之间的连接关系
- ▶ 学习算法
 - ▶ 通过训练数据来学习神经网络的参数

网络结构

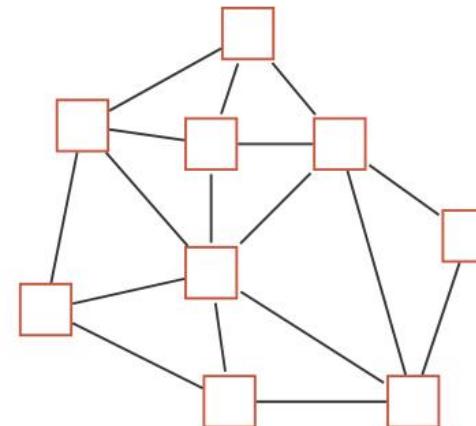
► 人工神经网络由神经元模型构成，这种由许多神经元组成的信息处理网络具有并行分布结构



(a) 前馈网络



(b) 记忆网络



(c) 图网络

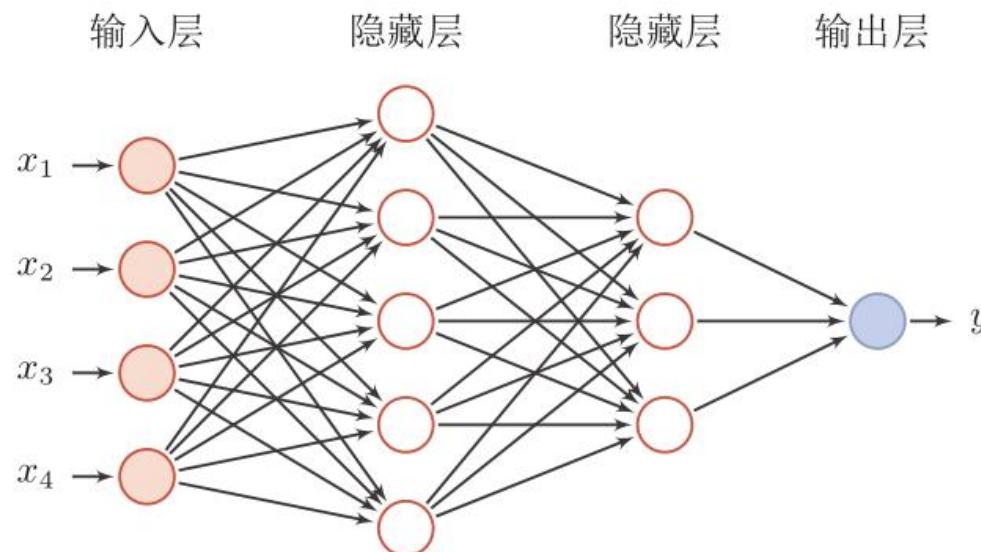
圆形节点表示一个神经元，方形节点表示一组神经元



6.2 前馈神经网络

前馈网络

- ▶ 前馈神经网络（全连接神经网络、多层感知器）
- ▶ 各神经元分别属于不同的层，层内无连接
- ▶ 相邻两层之间的神经元全部两两连接
- ▶ 整个网络中无反馈，信号从输入层向输出层单向传播

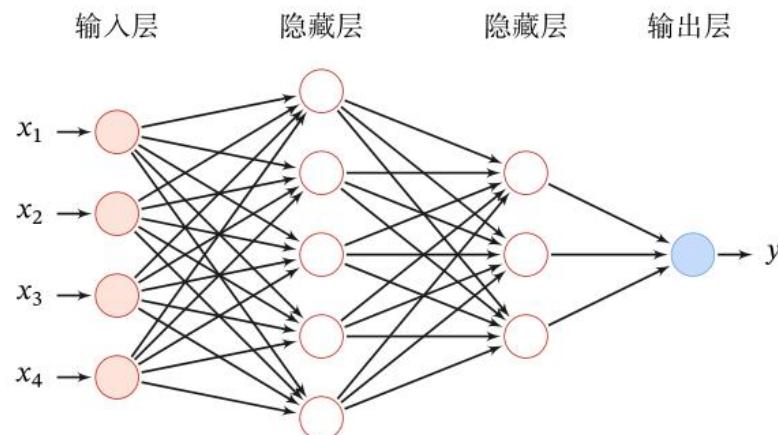


前馈网络

► 前馈神经网络通过下面公式进行信息传播

$$\mathbf{z}^{(l)} = \mathbf{W}^{(l)} \mathbf{a}^{(l-1)} + \mathbf{b}^{(l)},$$

$$\mathbf{a}^{(l)} = f_l(\mathbf{z}^{(l)}).$$

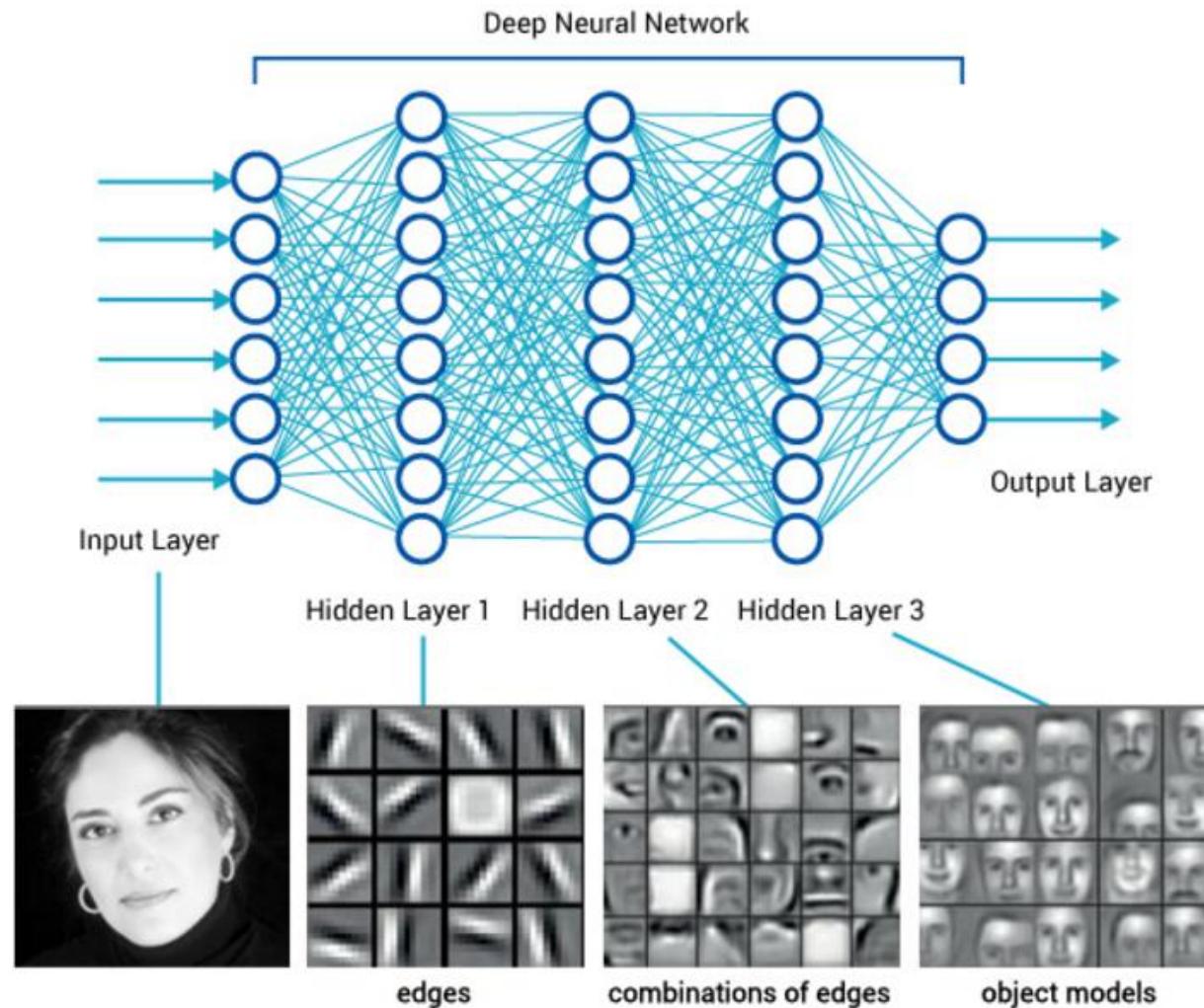


记号	含义
L	神经网络的层数
M_l	第 l 层神经元的个数
$f_l(\cdot)$	第 l 层神经元的激活函数
$\mathbf{W}^{(l)} \in \mathbb{R}^{M_l \times M_{l-1}}$	第 $l-1$ 层到第 l 层的权重矩阵
$\mathbf{b}^{(l)} \in \mathbb{R}^{M_l}$	第 $l-1$ 层到第 l 层的偏置
$\mathbf{z}^{(l)} \in \mathbb{R}^{M_l}$	第 l 层神经元的净输入 (净活性值)
$\mathbf{a}^{(l)} \in \mathbb{R}^{M_l}$	第 l 层神经元的输出 (活性值)

► 前馈计算

$$\mathbf{x} = \mathbf{a}^{(0)} \rightarrow \mathbf{z}^{(1)} \rightarrow \mathbf{a}^{(1)} \rightarrow \mathbf{z}^{(2)} \rightarrow \dots \rightarrow \mathbf{a}^{(L-1)} \rightarrow \mathbf{z}^{(L)} \rightarrow \mathbf{a}^{(L)} = \phi(\mathbf{x}; \mathbf{W}, \mathbf{b})$$

深层前馈神经网络



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，就可以以任意的精度来近似任何一个定义在实数空间中的有界闭集函数

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。

应用到机器学习

- ▶ 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

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

分类器 神经网络

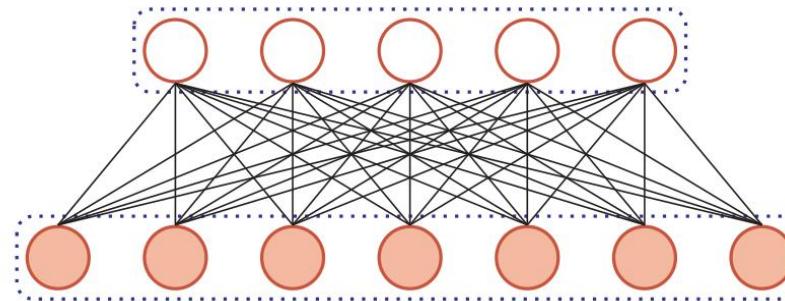
- ▶ 如果 $g(\cdot)$ 为Logistic回归，那么Logistic回归分类器可以看成神经网络的最后一层；如果使用Softmax回归分类器，可用于多分类问题



6.3 卷积神经网络

全连接前馈神经网络

► 权重矩阵的参数非常多



► 局部不变性特征

- 自然图像中的物体都具有局部不变性特征
 - 尺度缩放、平移、旋转等操作不影响其语义信息
- 全连接前馈网络很难提取这些局部不变特征

卷积神经网络

- ▶ 卷积神经网络 (Convolutional Neural Networks, CNN)
 - ▶ 一种前馈神经网络
 - ▶ 受生物学上感受野 (Receptive Field) 的机制而提出的
 - ▶ 在视觉神经系统中，一个神经元的感受野是指视网膜上的特定区域，只有这个区域内的刺激才能够激活该神经元
- ▶ 卷积神经网络有三个结构上的特性
 - ▶ 局部连接
 - ▶ 权重共享
 - ▶ 空间或时间上的次采样

卷积

- ▶ 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

$$\begin{aligned}y_t &= 1 \times x_t + 1/2 \times x_{t-1} + 1/4 \times x_{t-2} \\&= w_1 \times x_t + w_2 \times x_{t-1} + w_3 \times x_{t-2} \\&= \sum_{k=1}^3 w_k \cdot x_{t-k+1}.\end{aligned}$$

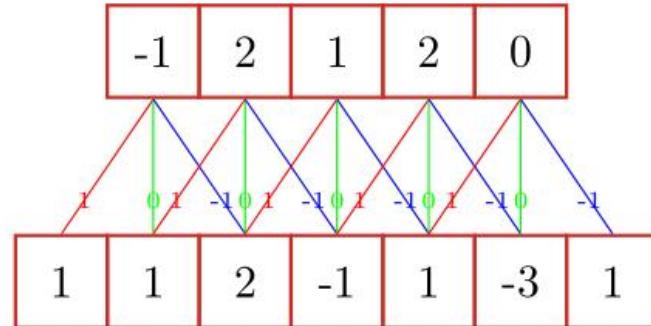
滤波器 (filter) 或卷积核 (convolution kernel)

卷积

► 给定一个输入信号序列 x 和滤波器 w , 卷积的输出为

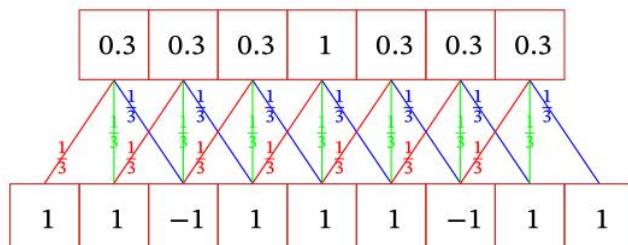
$$y_t = \sum_{k=1}^K w_k x_{t-k+1}$$

Filter: [-1,0,1]



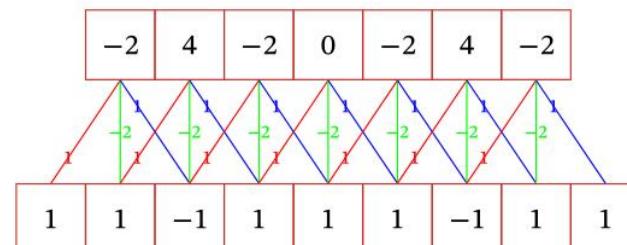
► 不同的滤波器来提取信号序列中的不同特征

低频信息



(a) 滤波器 $[1/3, 1/3, 1/3]$

高频信息

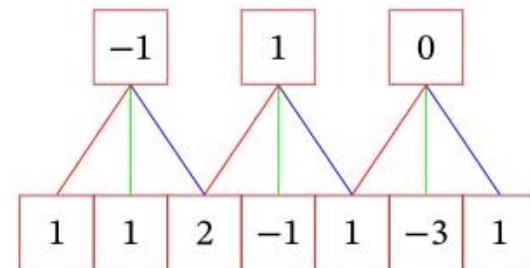


(b) 滤波器 $[1, -2, 1]$

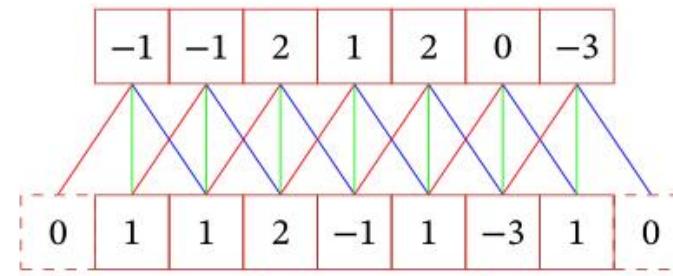
$$y''(u) = y(u+1) + y(u-1) - 2y(u) \quad \text{二阶微分}$$

卷积扩展

► 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

- 窄卷积：步长 $T = 1$ ，两端不补零 $P = 0$ ，卷积后输出长度为 $M - K + 1$
- 宽卷积：步长 $T = 1$ ，两端补零 $P = K - 1$ ，卷积后输出长度 $M + K - 1$
- 等宽卷积：步长 $T = 1$ ，两端补零 $P = (K - 1)/2$ ，卷积后输出长度 M

目前的文献中，卷积一般默认认为等宽卷积

二维卷积

► 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

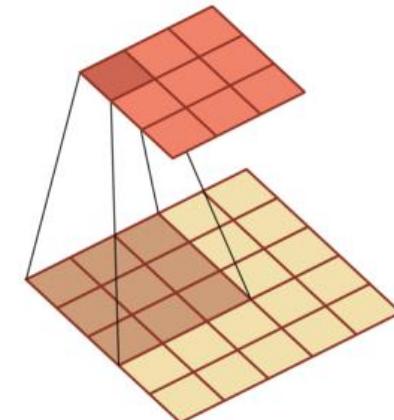
一个输入信息 \mathbf{X} 和滤波器 \mathbf{W} 的二维卷积定义为

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

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

1	1	1 <small>$\times -1$</small>	1 <small>$\times 0$</small>	1 <small>$\times 0$</small>
-1	0	-3 <small>$\times 0$</small>	0 <small>$\times 0$</small>	1 <small>$\times 0$</small>
2	1	1 <small>$\times 0$</small>	-1 <small>$\times 0$</small>	0 <small>$\times 1$</small>
0	-1	1	2	1
1	2	1	1	1

$$* \begin{array}{|c|c|c|} \hline 1 & 0 & 0 \\ \hline 0 & 0 & 0 \\ \hline 0 & 0 & -1 \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline 0 & -2 & -1 \\ \hline 2 & 2 & 4 \\ \hline -1 & 0 & 0 \\ \hline \end{array}$$



卷积作为特征提取器



原始图像

$$\begin{array}{|c|c|c|} \hline \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \\ \hline \frac{1}{8} & \frac{1}{4} & \frac{1}{8} \\ \hline \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \\ \hline \end{array}$$

\otimes

$$\begin{array}{|c|c|c|} \hline 0 & 1 & 0 \\ \hline 1 & -4 & 1 \\ \hline 0 & 1 & 0 \\ \hline \end{array}$$

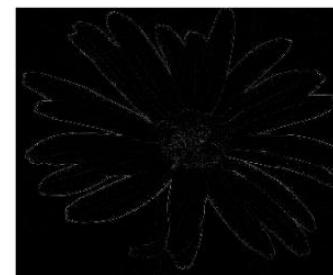
=



滤波器

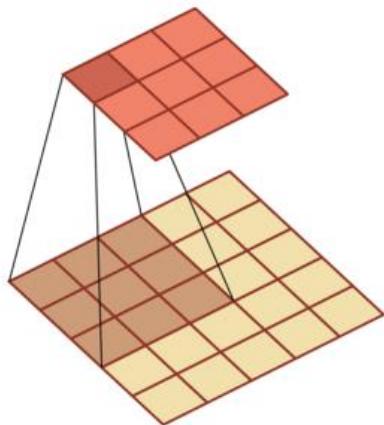
$$\begin{array}{|c|c|c|} \hline 0 & 1 & 1 \\ \hline -1 & 0 & 1 \\ \hline -1 & -1 & 0 \\ \hline \end{array}$$

=

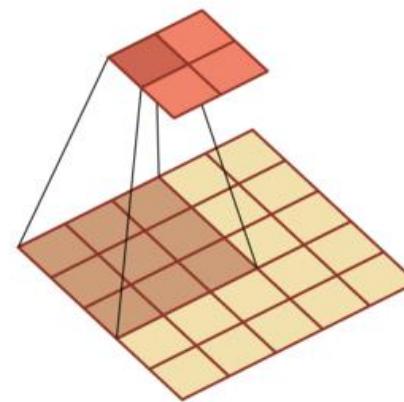


输出特征映射

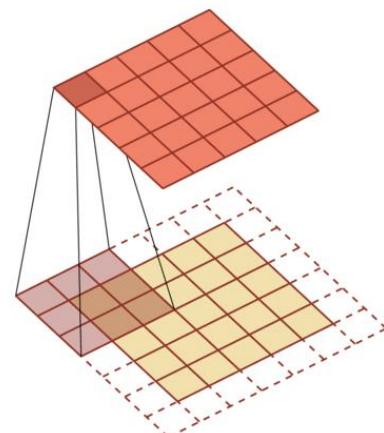
二维卷积



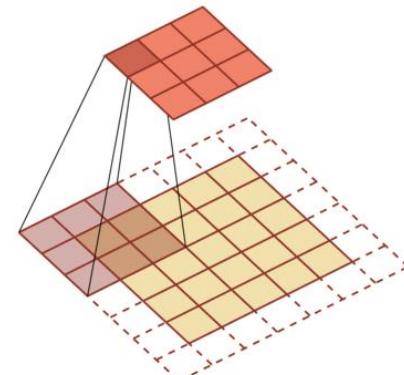
步长1，零填充0



步长2，零填充0



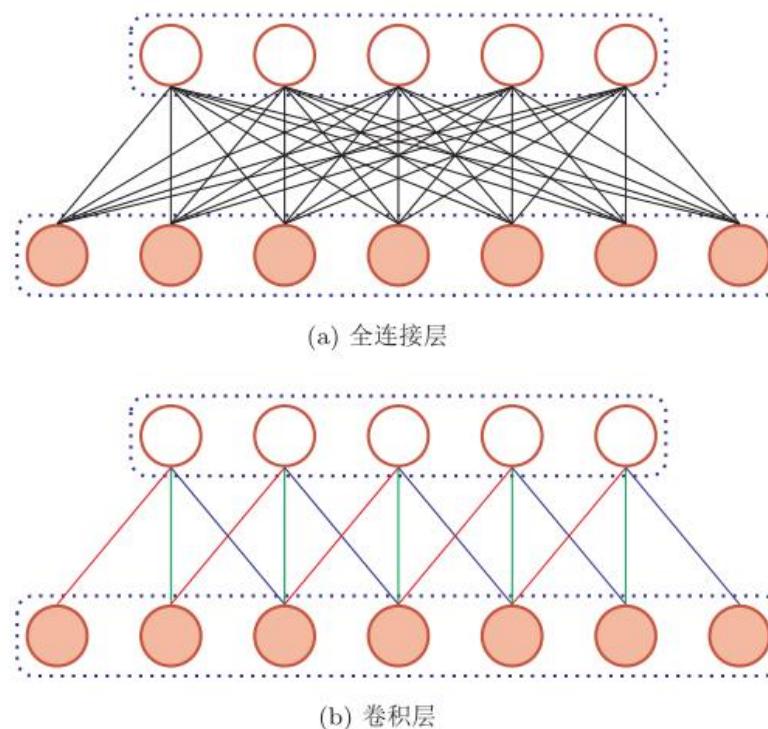
步长1，零填充1



步长2，零填充1

卷积神经网络

► 用卷积层代替全连接层



互相关

- ▶ 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

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

↓

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

除非特别声明，卷积一般指“互相关”

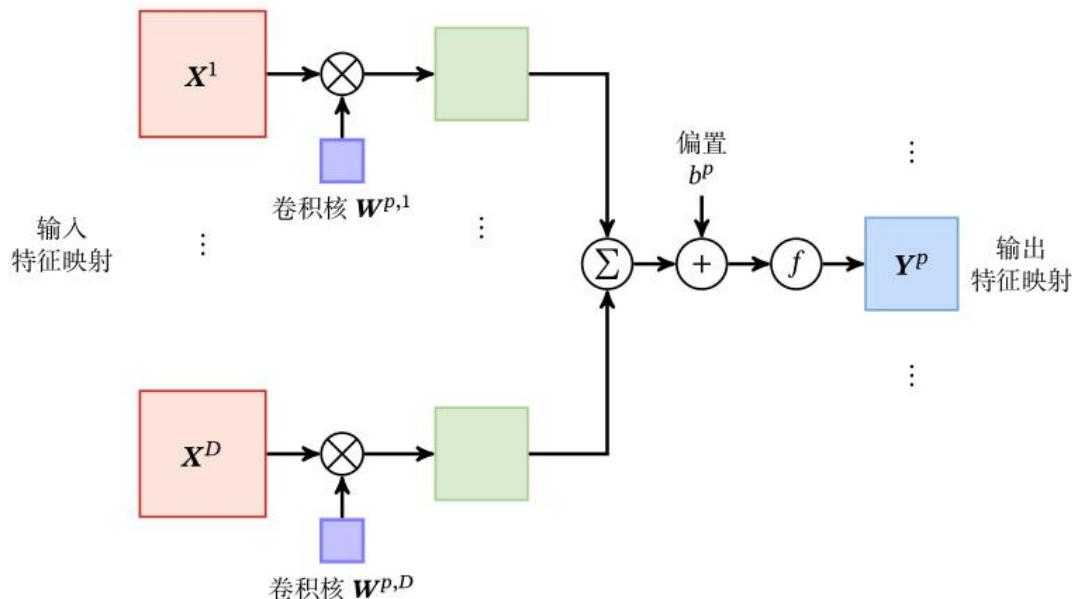
多个卷积核

► 特征映射 (Feature Map) : 图像经过卷积后得到的特征

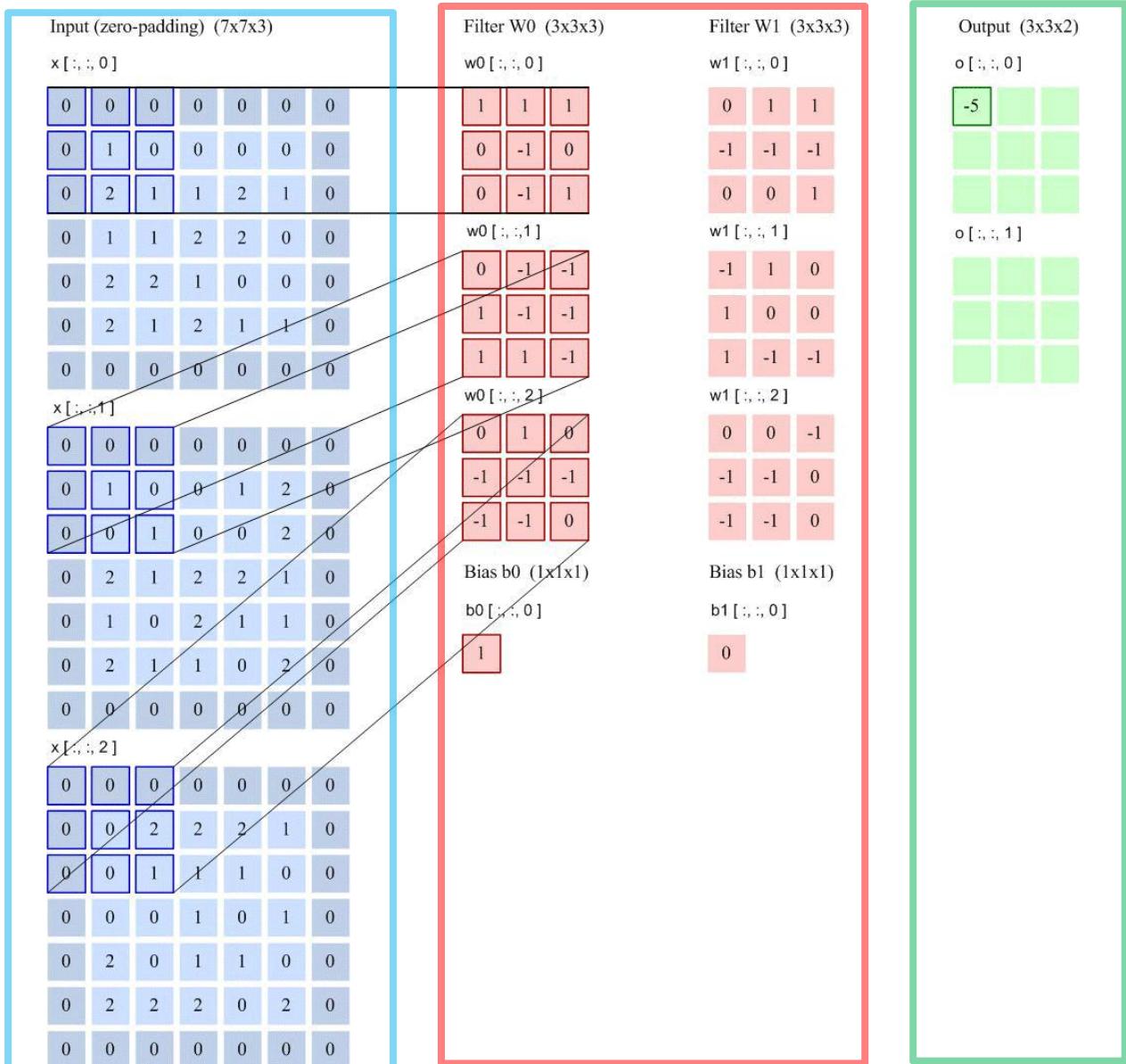
► 卷积层

► 输入: D个特征映射 $M \times N \times D$

► 输出: P个特征映射 $M' \times N' \times P$



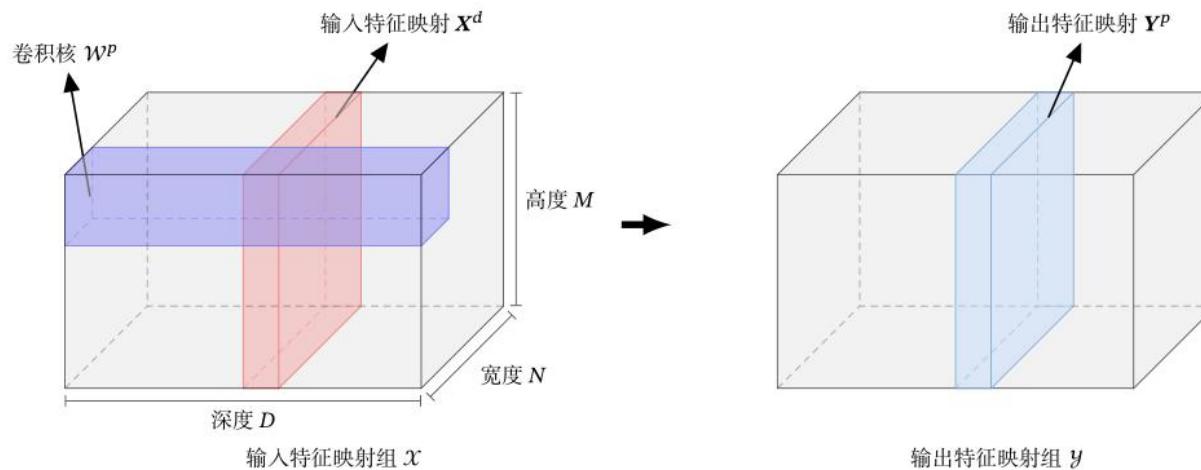
$$\begin{aligned} Z^p &= W^p \otimes X + b^p = \sum_{d=1}^D W^{p,d} \otimes X^d + b^p, \\ Y^p &= f(Z^p). \end{aligned}$$



步长2
filter 3*3
filter个数6
零填充 1

卷积层

► 典型的卷积层为3维结构

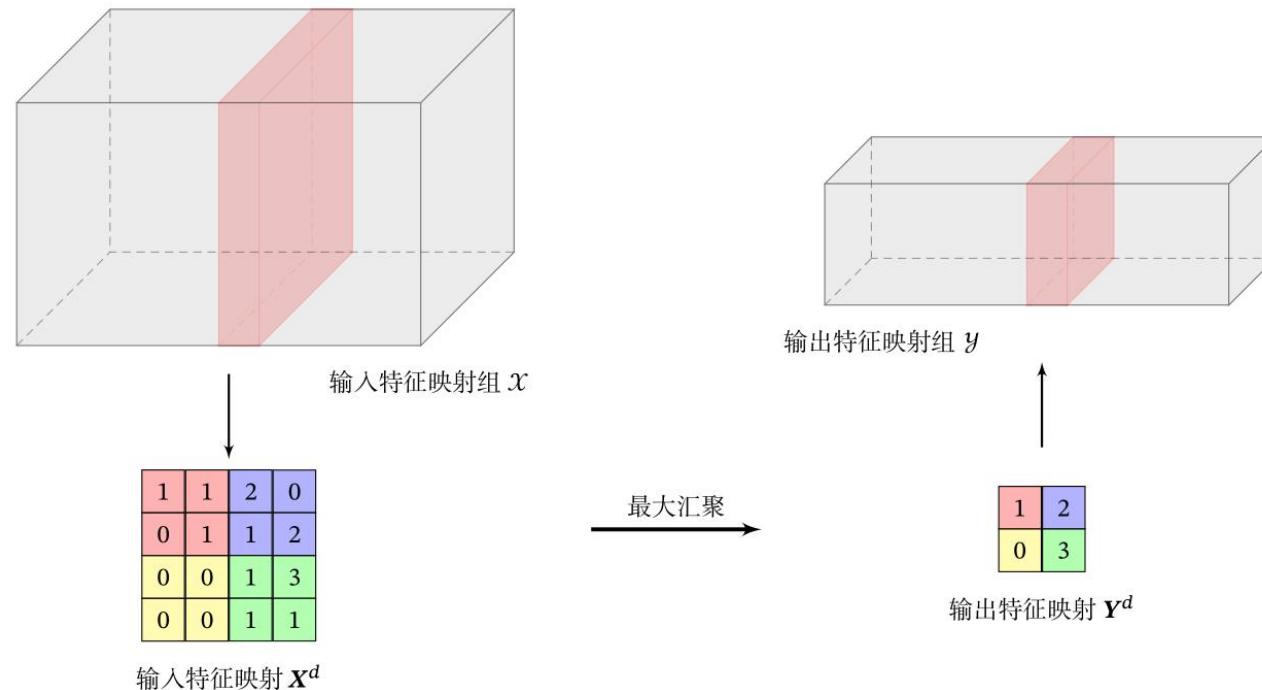


$$\mathbf{Z}^p = \mathbf{W}^p \otimes \mathbf{X} + b^p = \sum_{d=1}^D \mathbf{W}^{p,d} \otimes \mathbf{X}^d + b^p,$$

$$\mathbf{Y}^p = f(\mathbf{Z}^p).$$

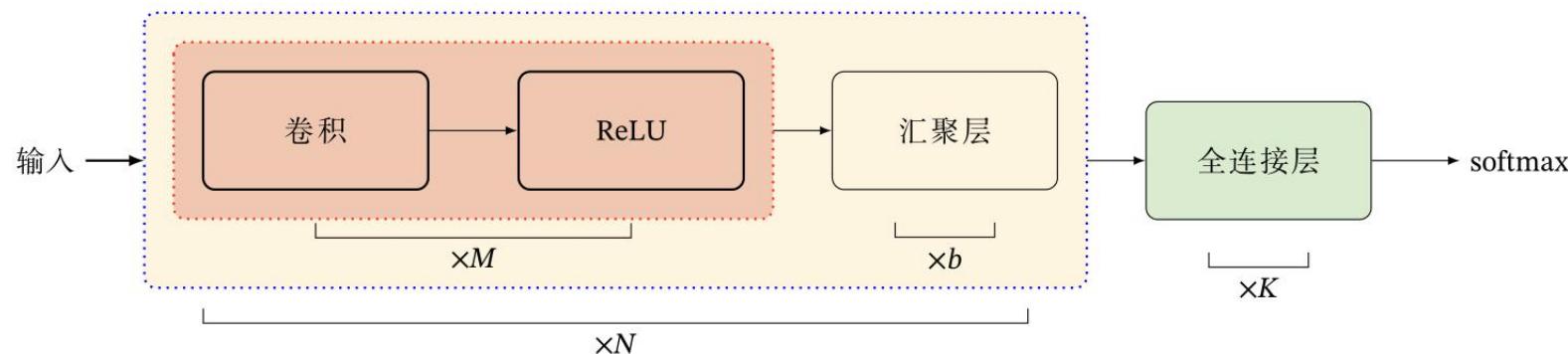
汇聚层（池化层）

▶ 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



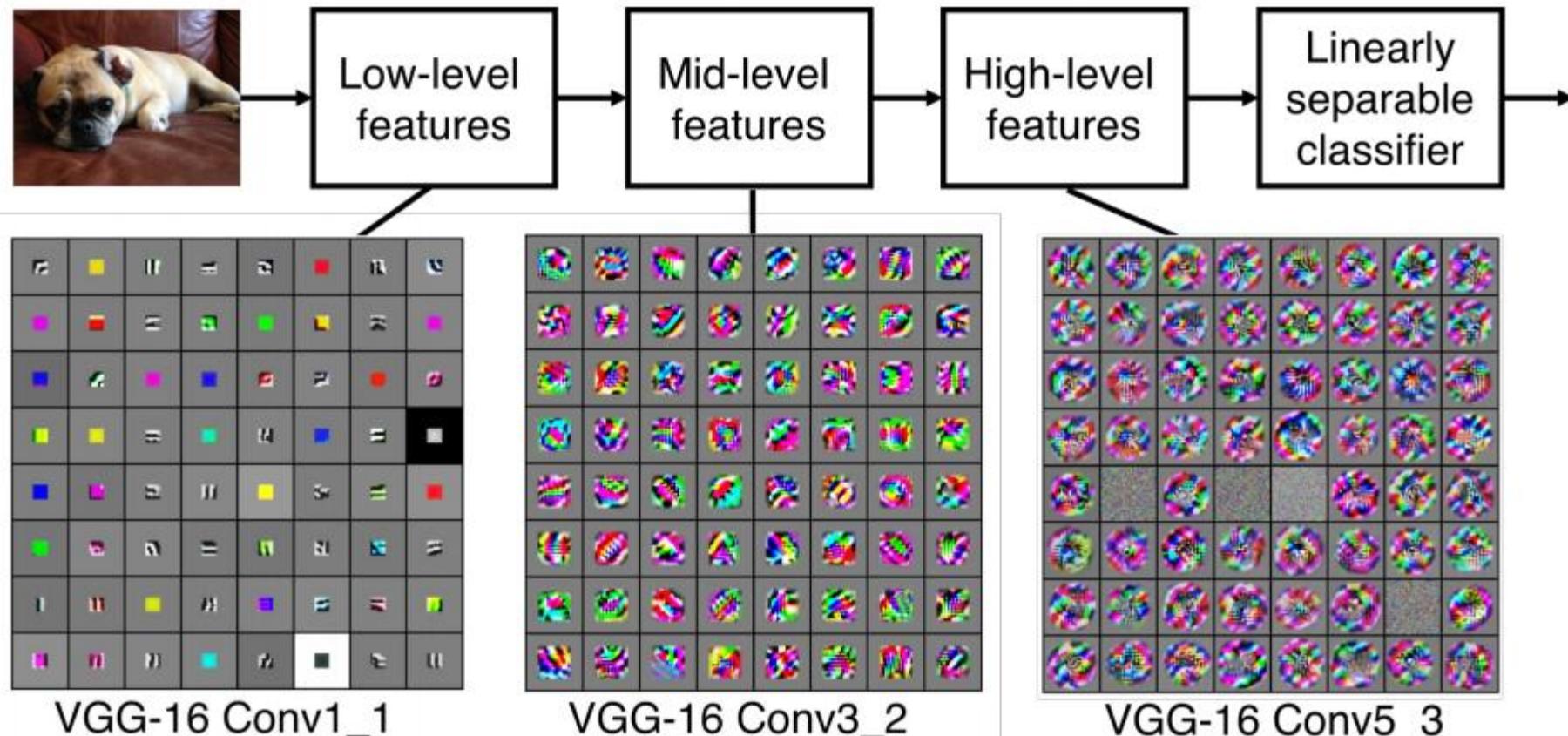
卷积网络结构

- ▶ 卷积网络是由卷积层、汇聚层、全连接层交叉堆叠而成
- ▶ 趋向于小卷积、大深度
- ▶ 趋向于全卷积
- ▶ 典型结构



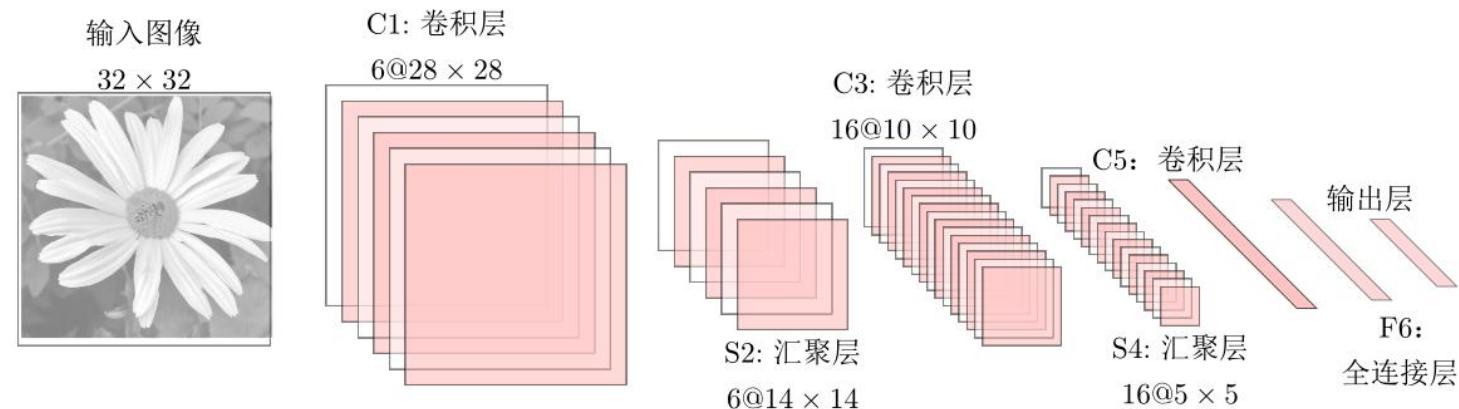
- ▶ 一个卷积块为连续M个卷积层和b个汇聚层（M通常设置为2~5，b为0或1）
- ▶ 一个卷积网络中可以堆叠N个连续的卷积块，然后接着K个全连接层（N的取值区间比较大，比如1~100或者更大；K一般为0~2）

卷积网络结构



代表：LeNet-5

- LeNet-5 是一个非常成功的神经网络模型
- 基于 LeNet-5 的手写数字识别系统在 90 年代被美国很多银行使用，用来识别支票上面的手写数字
- LeNet-5 共有 7 层



代表：AlexNet

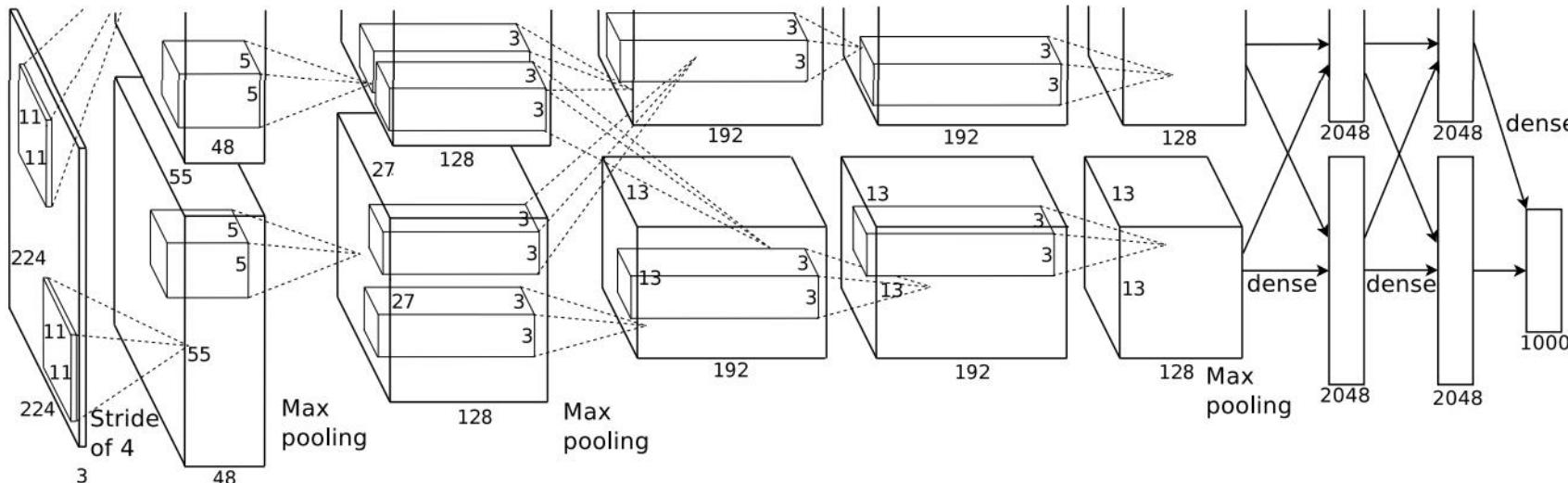
► 2012 ILSVRC winner

► 第一个现代深度卷积网络模型

► 首次使用了很多现代深度卷积网络的一些技术方法

► 使用GPU进行并行训练，采用了ReLU作为非线性激活函数，使用Dropout防止过拟合，使用数据增强

► 5个卷积层、3个汇聚层和3个全连接层



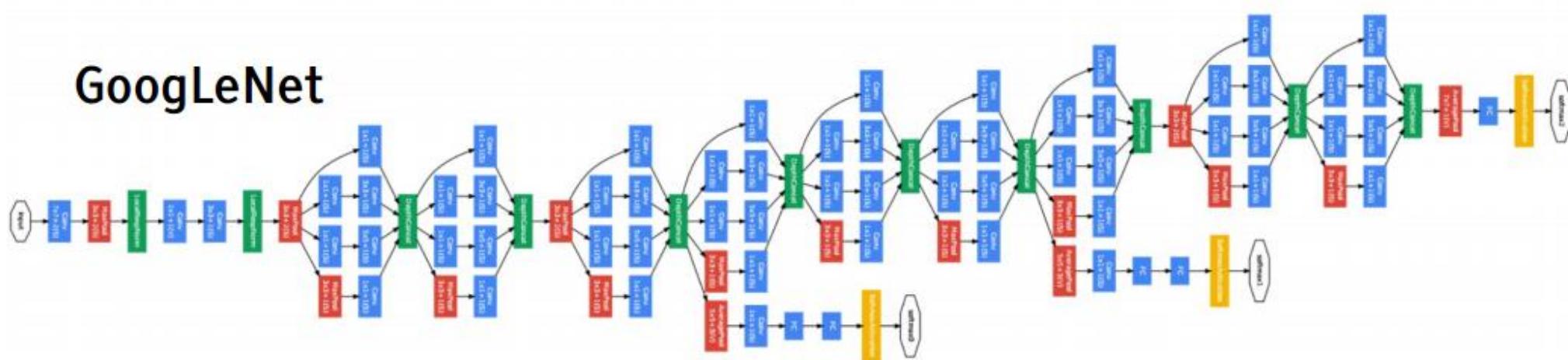
代表：Inception网络

► 2014 ILSVRC winner (22层)

► 参数：GoogLeNet: 4M VS AlexNet: 60M

► 错误率：6.7%

► Inception网络是由多个inception模块和少量的汇聚层堆叠而成

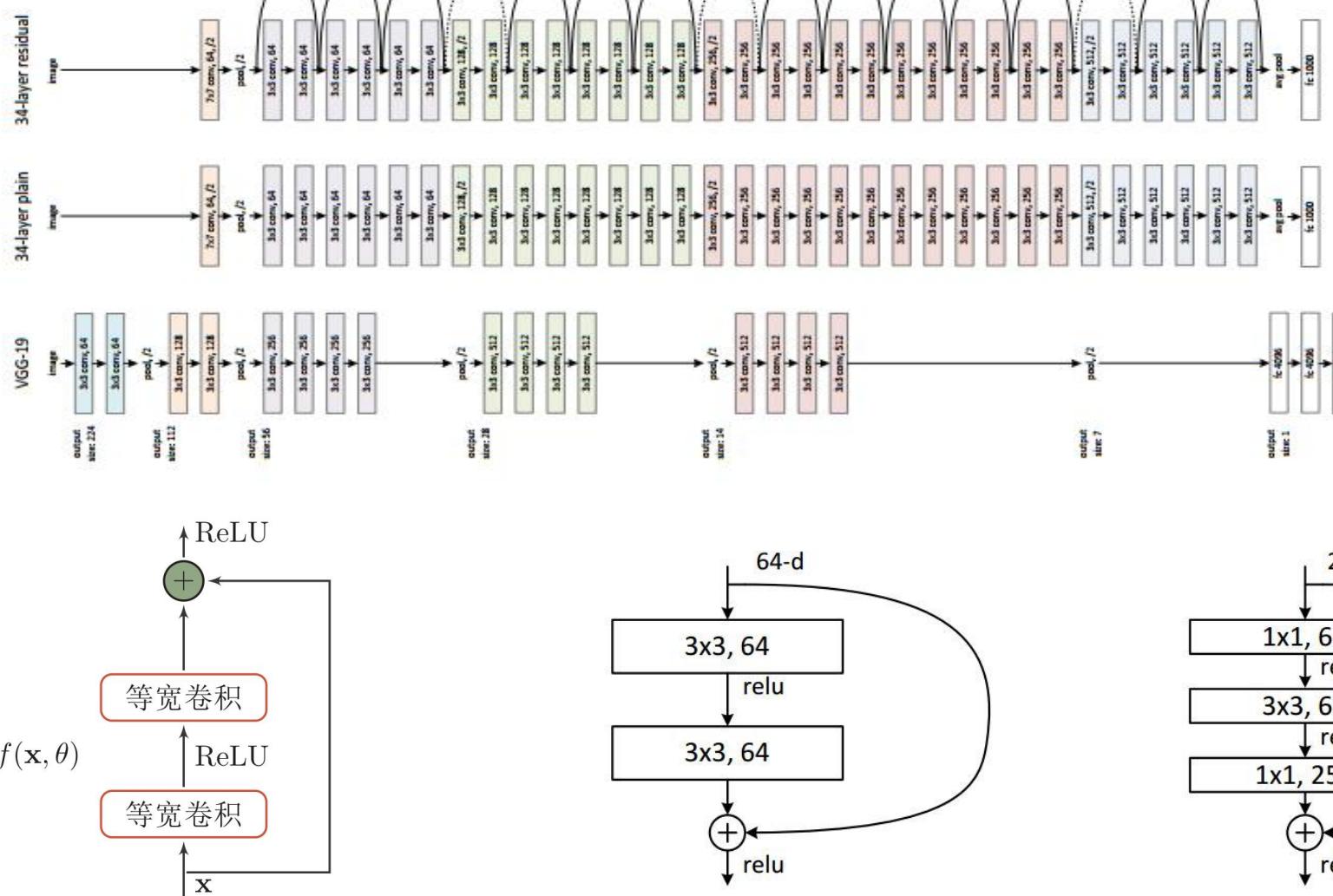


代表：ResNet

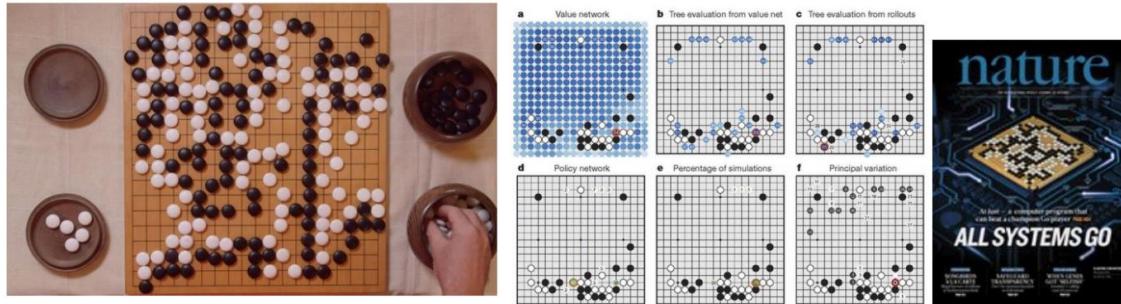
- ▶ 2015 ILSVRC winner (152层)
- ▶ 错误率：3.57%
- ▶ 残差网络（Residual Network，ResNet）是通过给非线性的卷积层增加直连边的方式来提高信息的传播效率
- ▶ 假设在一个深度网络中，期望一个非线性单元（可以为一层或多层的卷积层） $f(\mathbf{x}, \theta)$ 去逼近一个目标函数为 $h(\mathbf{x})$
- ▶ 将目标函数拆分成两部分：恒等函数和残差函数

$$h(\mathbf{x}) = \underbrace{\mathbf{x}}_{\text{恒等函数}} + \underbrace{(h(\mathbf{x}) - \mathbf{x})}_{\text{残差函数}} \rightarrow f(\mathbf{x}, \theta)$$

代表：ResNet



应用：AlphaGo



The input to the policy network is a $19 \times 19 \times 48$ image stack consisting of 48 feature planes. The first hidden layer zero pads the input into a 23×23 image, then convolves k filters of kernel size 5×5 with stride 1 with the input image and applies a rectifier nonlinearity. Each of the subsequent hidden layers 2 to 12 zero pads the respective previous hidden layer into a 21×21 image, then convolves k filters of kernel size 3×3 with stride 1, again followed by a rectifier nonlinearity. The final layer convolves 1 filter of kernel size 1×1 with stride 1, with a different bias for each position, and applies a softmax function. The match version of AlphaGo used $k = 192$ filters; Fig. 2b and Extended Data Table 3 additionally show the results of training with $k = 128, 256$ and 384 filters.

policy network:

[$19 \times 19 \times 48$] Input

CONV1: 192 5×5 filters , stride 1, pad 2 => [$19 \times 19 \times 192$]

CONV2..12: 192 3×3 filters, stride 1, pad 1 => [$19 \times 19 \times 192$]

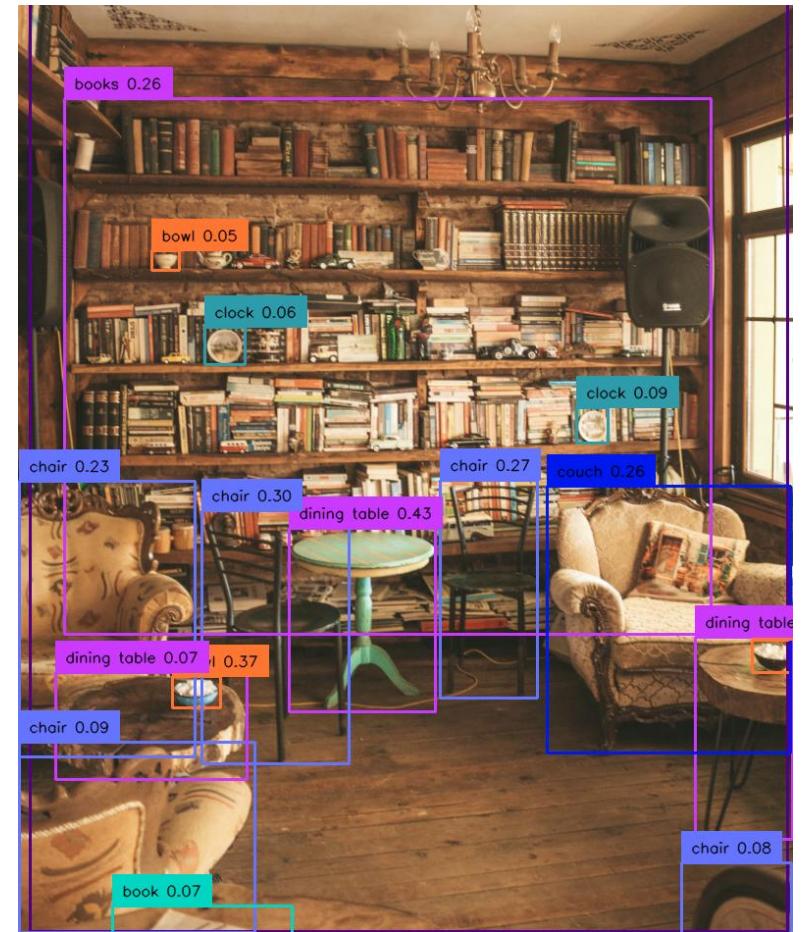
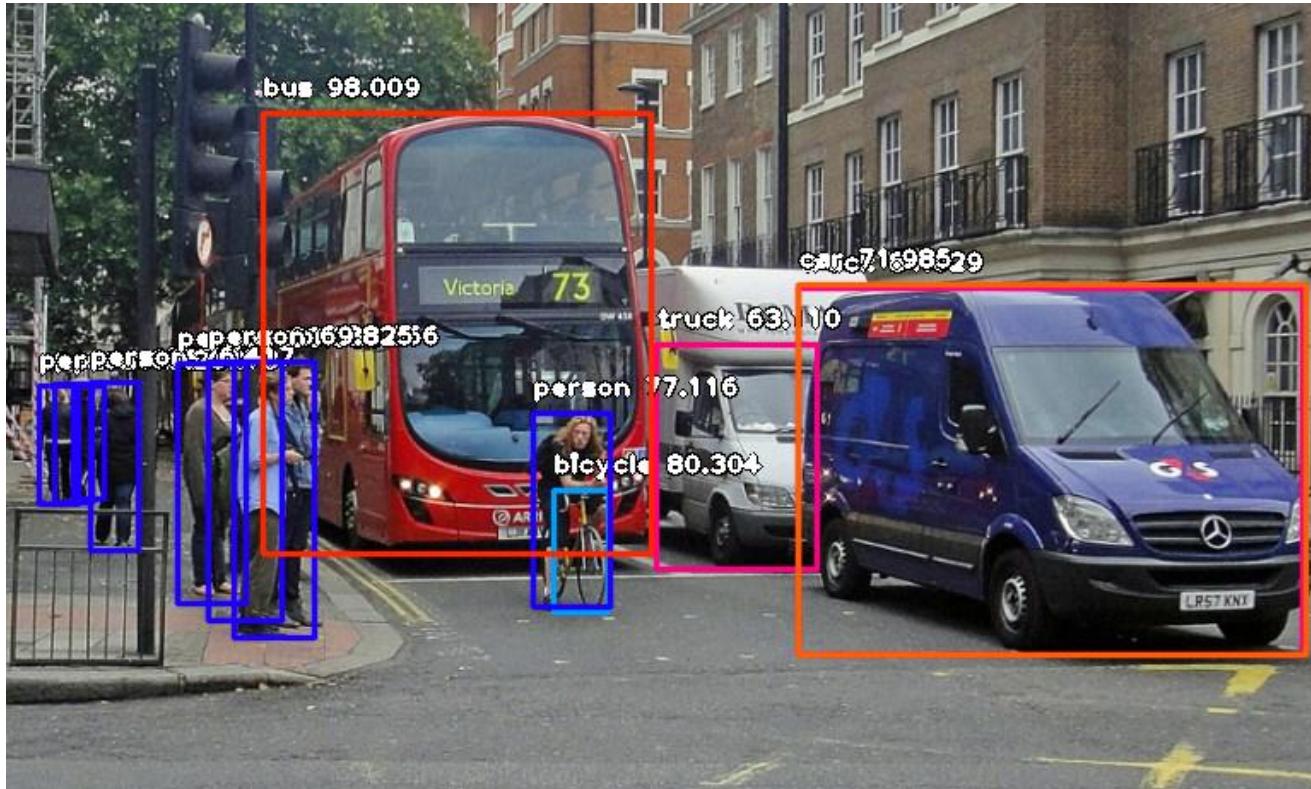
CONV: 1 1×1 filter, stride 1, pad 0 => [19×19] (*probability map of promising moves*)

分布式系统：1202 个CPU 和176 块GPU

单机版：48 个CPU 和8 块GPU

走子速度：3 毫秒-2 微秒

应用：目标检测（Object Detection）

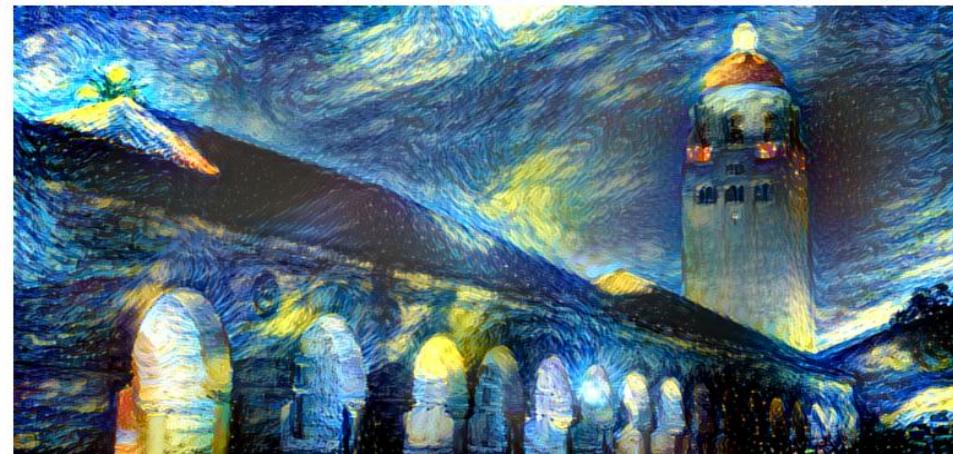


应用：图像分割



<https://segment-anything.com/demo>

更多



OCR

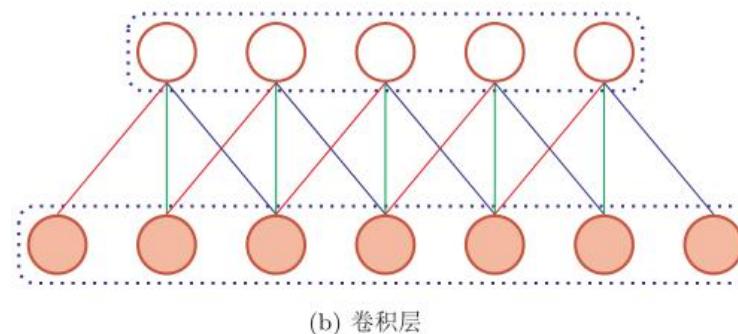
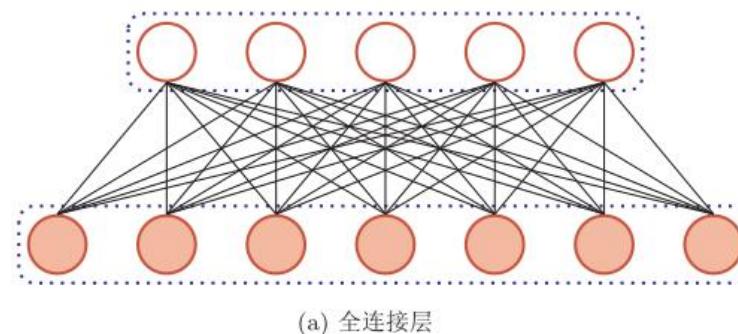
画风迁移



6.4 循环神经网络

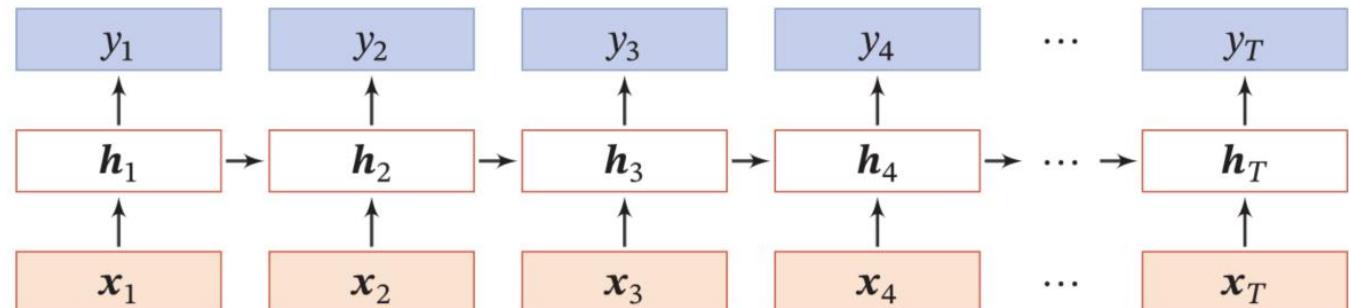
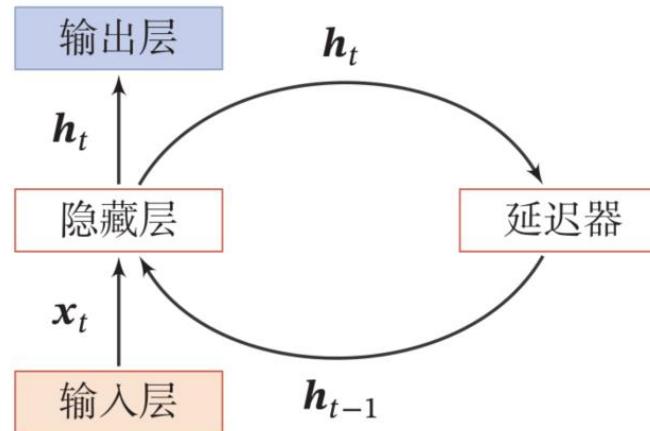
前馈网络

- ▶ 连接存在层与层之间，每层的节点之间是无连接的
- ▶ 输入和输出的维数都是固定的，不能任意改变
- ▶ 无法处理变长的序列数据



循环神经网络（ Recurrent Neural Network , RNN ）

► 循环神经网络通过使用带自反馈的神经元，能够处理任意长度的时序数据



$$\mathbf{h}_t = f(\mathbf{h}_{t-1}, \mathbf{x}_t)$$

► 循环神经网络比前馈神经网络更加符合生物神经网络的结构
► 广泛应用在语音识别、语言模型以及自然语言生成等任务上

简单循环网络 (Simple Recurrent Network , SRN)

► 状态更新：

$$\mathbf{h}_t = f(\mathbf{U}\mathbf{h}_{t-1} + \mathbf{W}\mathbf{x}_t + \mathbf{b})$$

► 一个完全连接的循环网络是任何非线性动力系统的近似器

定理 6.1 - 循环神经网络的通用近似定理 [Haykin, 2009]: 如果一个完全连接的循环神经网络有足够的 sigmoid 型隐藏神经元，它可以以任意的准确率去近似任何一个非线性动力系统

$$\mathbf{s}_t = g(\mathbf{s}_{t-1}, \mathbf{x}_t), \quad (6.10)$$

$$\mathbf{y}_t = o(\mathbf{s}_t), \quad (6.11)$$

其中 \mathbf{s}_t 为每个时刻的隐状态, \mathbf{x}_t 是外部输入, $g(\cdot)$ 是可测的状态转换函数, $o(\cdot)$ 是连续输出函数, 并且对状态空间的紧致性没有限制.

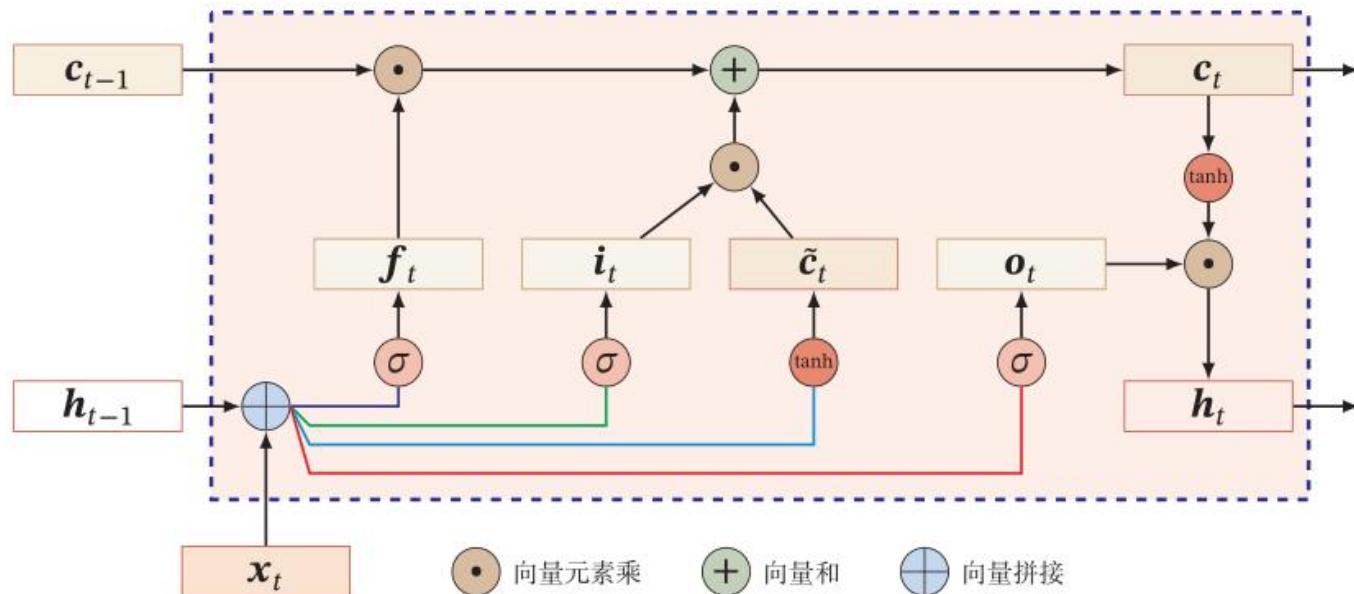
图灵完备

- ▶ 图灵完备 (Turing Completeness) 是指一种数据操作规则，比如一种计算机编程语言，可以实现图灵机的所有功能，解决所有的可计算问题

定理 6.2 - 图灵完备 [Siegelmann et al., 1991]: 所有的图灵机都可以被一个由使用 Sigmoid 型激活函数的神经元构成的全连接循环网络来进行模拟。

一个完全连接的循环神经网络可以
近似解决所有的可计算问题！

长短期记忆神经网络 (Long Short-Term Memory, LSTM)



$$\mathbf{i}_t = \sigma(W_i \mathbf{x}_t + U_i \mathbf{h}_{t-1} + \mathbf{b}_i),$$

$$\mathbf{f}_t = \sigma(W_f \mathbf{x}_t + U_f \mathbf{h}_{t-1} + \mathbf{b}_f),$$

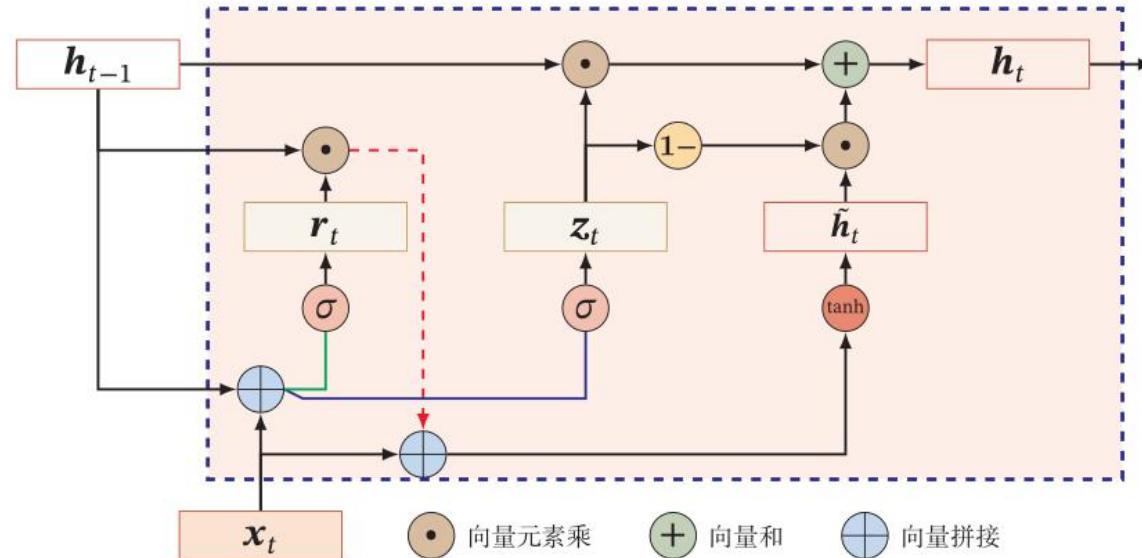
$$\mathbf{o}_t = \sigma(W_o \mathbf{x}_t + U_o \mathbf{h}_{t-1} + \mathbf{b}_o),$$

$$\tilde{\mathbf{c}}_t = \tanh(W_c \mathbf{x}_t + U_c \mathbf{h}_{t-1} + \mathbf{b}_c)$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t,$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t),$$

Gated Recurrent Unit, GRU



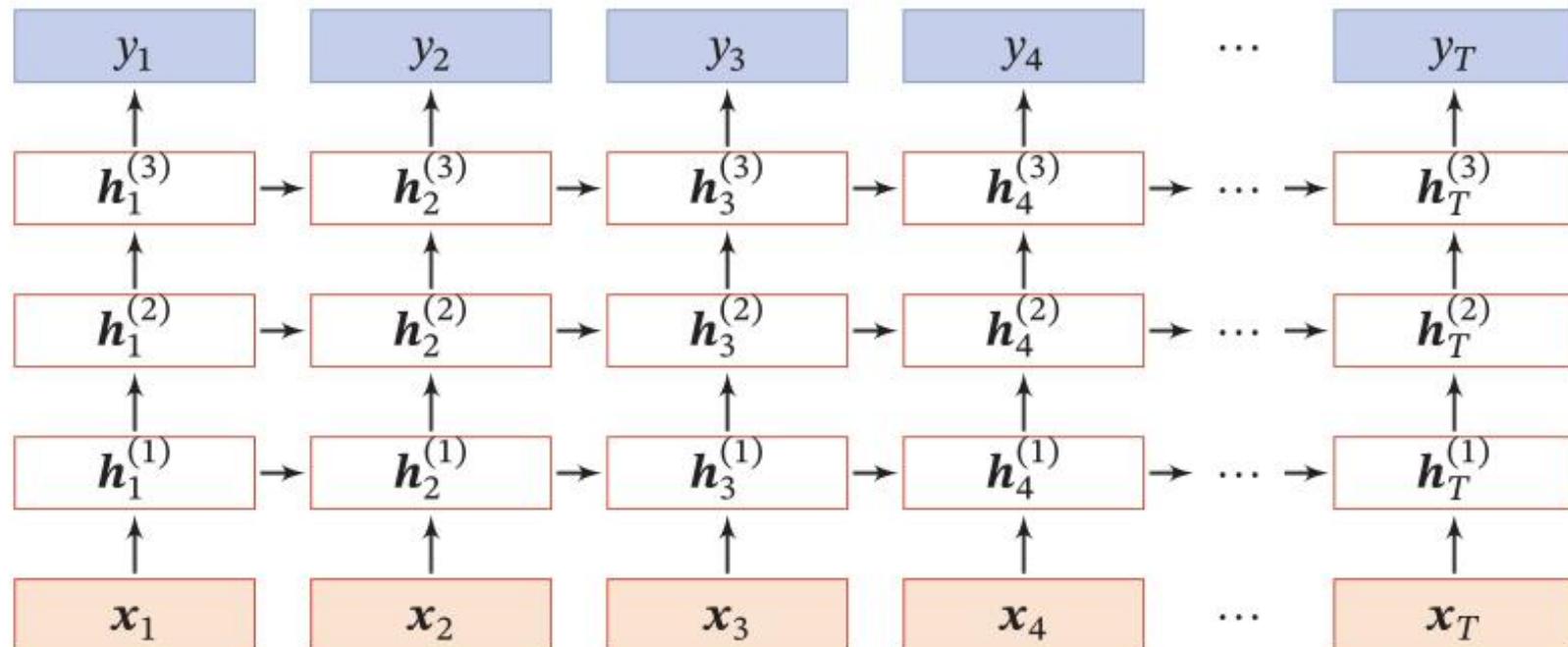
$$\mathbf{r}_t = \sigma(\mathbf{W}_r \mathbf{x}_t + \mathbf{U}_r \mathbf{h}_{t-1} + \mathbf{b}_r),$$

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{W}_c \mathbf{x}_t + \mathbf{U}(\mathbf{r}_t \odot \mathbf{h}_{t-1}))$$

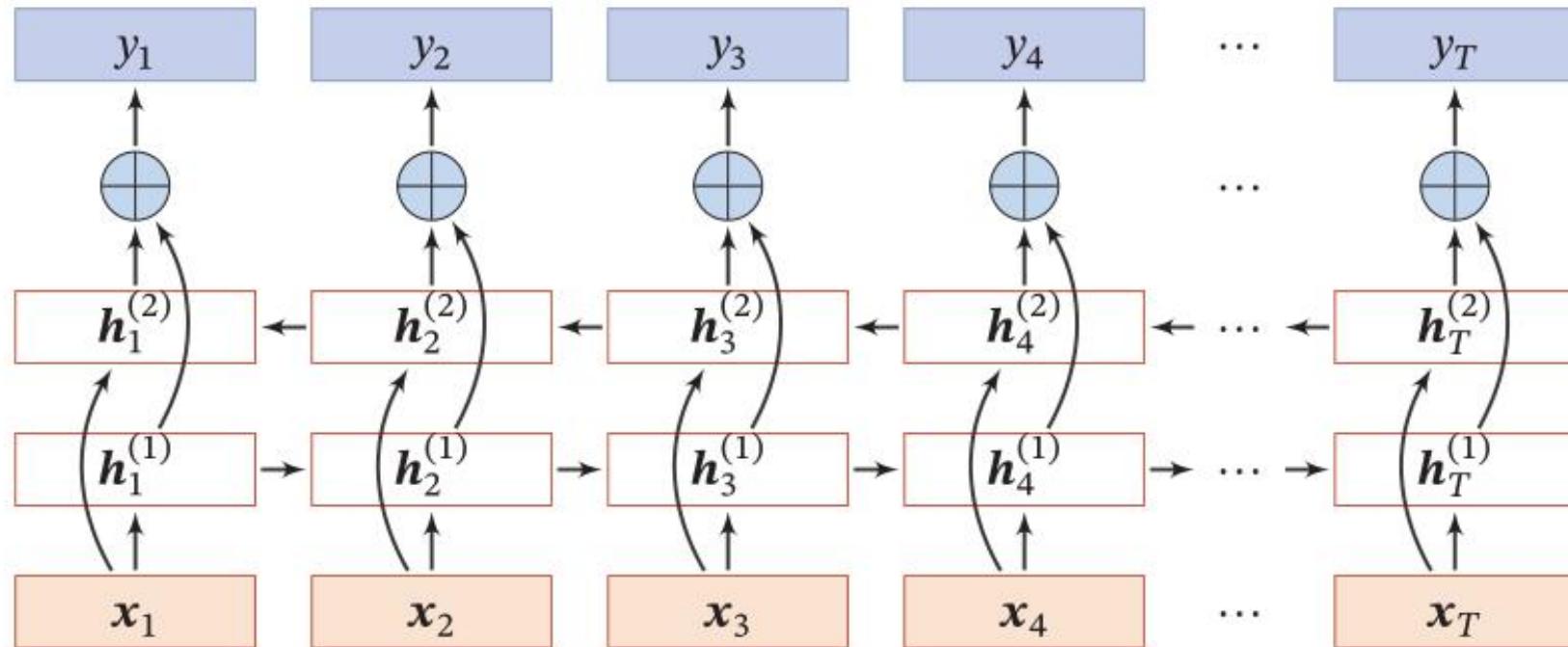
$$\mathbf{z}_t = \sigma(\mathbf{W}_z \mathbf{x}_t + \mathbf{U}_z \mathbf{h}_{t-1} + \mathbf{b}_z),$$

$$\mathbf{h}_t = \mathbf{z}_t \odot \mathbf{h}_{t-1} + (1 - \mathbf{z}_t) \odot \tilde{\mathbf{h}}_t,$$

堆叠循环神经网络



双向循环神经网络



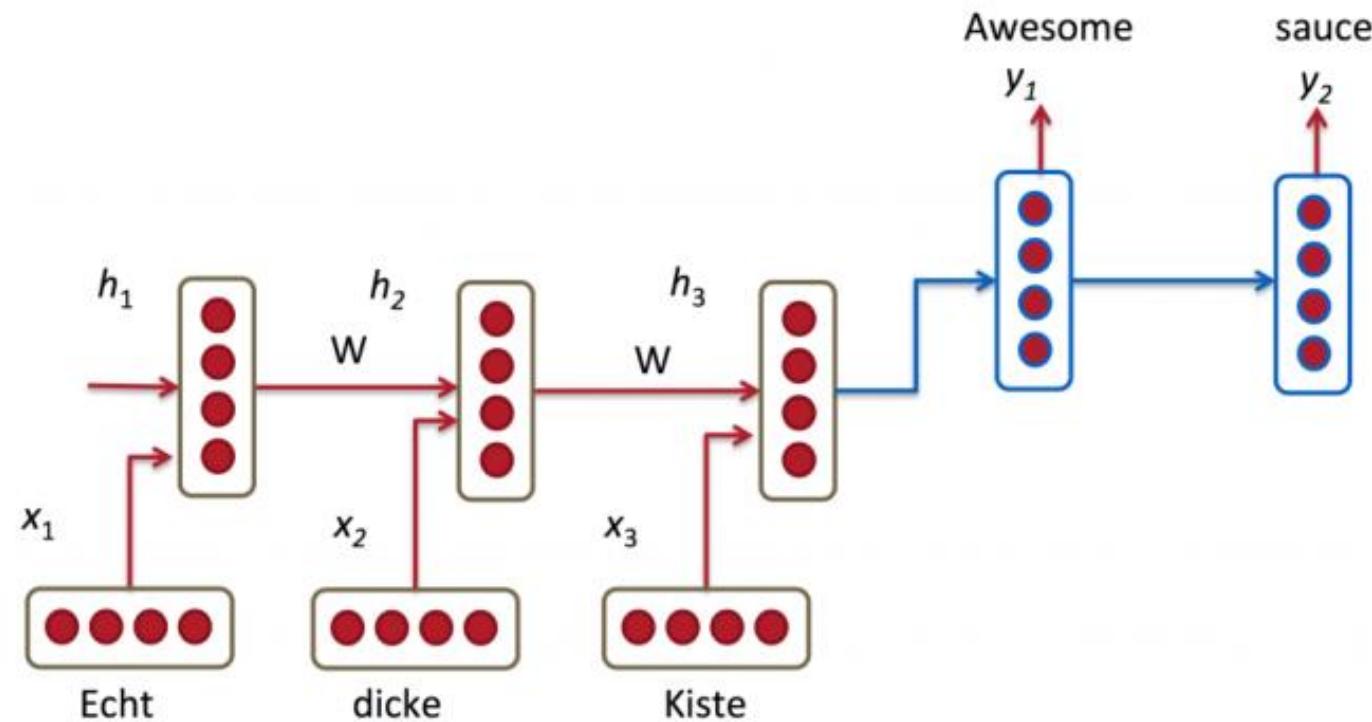
应用：作词机

- RNN在“学习”过汪峰全部作品后自动生成的歌词
- <https://github.com/phunterlau/wangfeng-rnn>

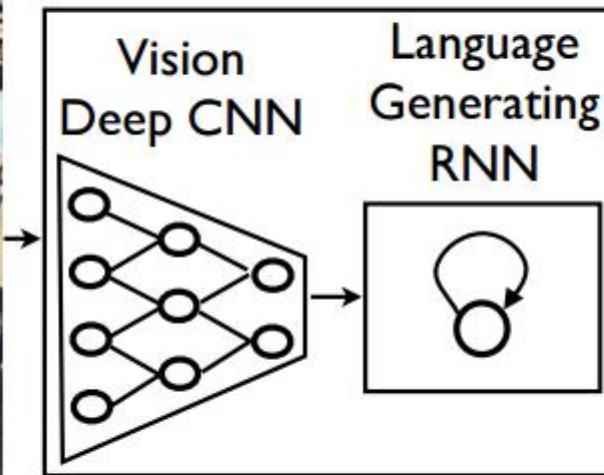
我在这里中的夜里
就像一场是一种生命的意叶
就像我的生活变得在我一样
可我们这是一个知道
我只是一天你会怎吗
可我们这是我们的是不要为你
我们想这有一种生活的时候

应用：机器翻译

►一个RNN用来编码，另一个RNN用来解码



应用：看图说话

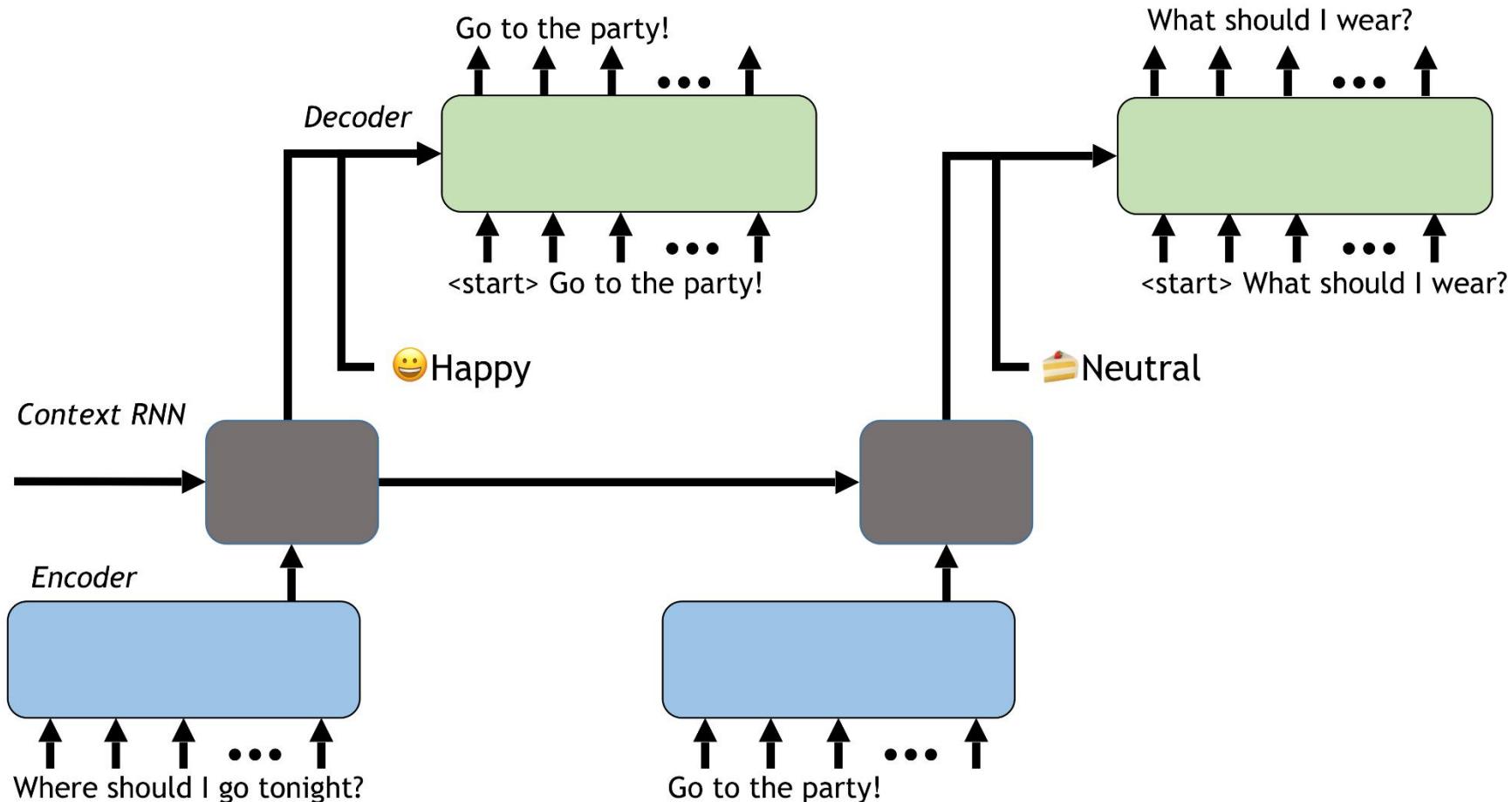


**A group of people
shopping at an
outdoor market.**

**There are many
vegetables at the
fruit stand.**

应用：对话系统

<https://github.com/lukalabs/cakechat>



循环神经网络总结

优点：

- ▶ 引入记忆
- ▶ 图灵完备

缺点：

- ▶ 长程依赖问题
- ▶ 记忆容量问题
- ▶ 并行能力

