神经网络与深度学习

邱锡鹏
复旦大学

9/11/2018

https://nndl.github.io/

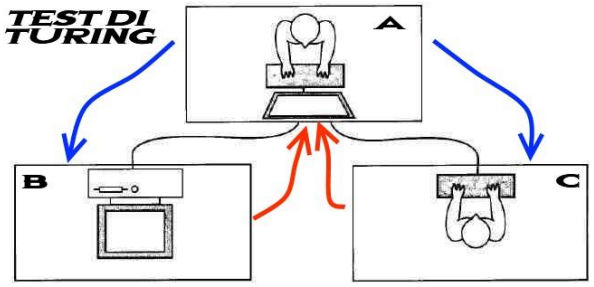
大纲

- 概述
  - 机器学习概述
  - 线性模型
- 基础网络模型
  - 前馈神经网络
  - 卷积神经网络
  - 循环神经网络
  - 网络优化与正则化
  - 记忆与注意力机制
  - 无监督学习
- 进阶模型
  - 概率图模型
  - 玻尔兹曼机
  - 深度信念网络
  - 深度生成模型
  - 深度强化学习

从人工智能开始

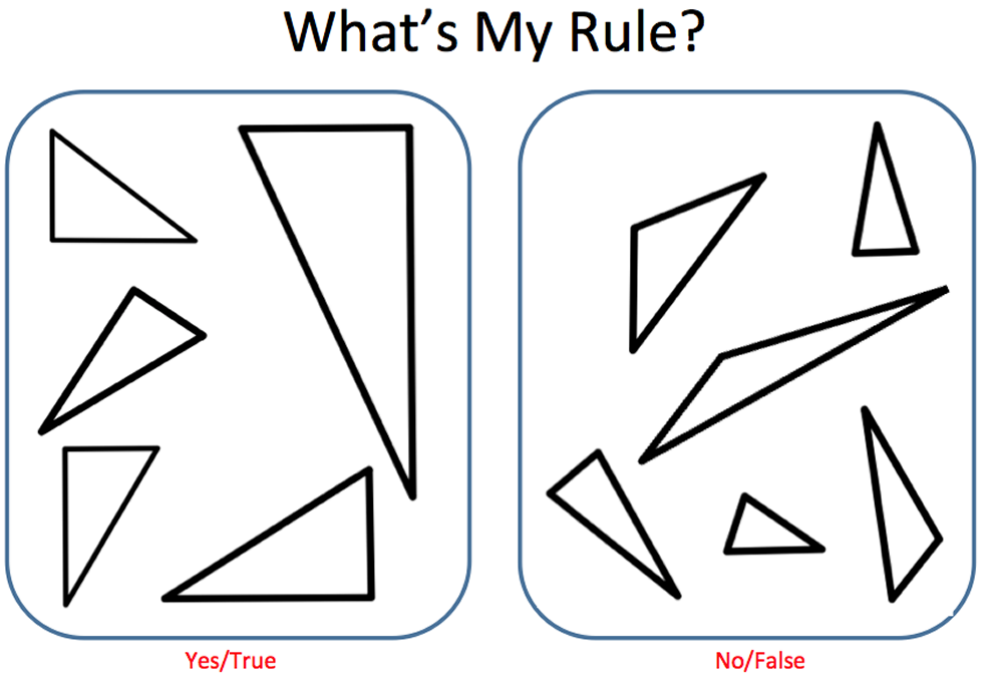
- 让机器具有人类的智能
  - 机器感知（计算机视觉、语音信息处理）
  - 学习（模式识别、机器学习、强化学习）
  - 语言（自然语言处理）
  - 记忆（知识表示）
  - 决策（规划、数据挖掘）

Alan Turing

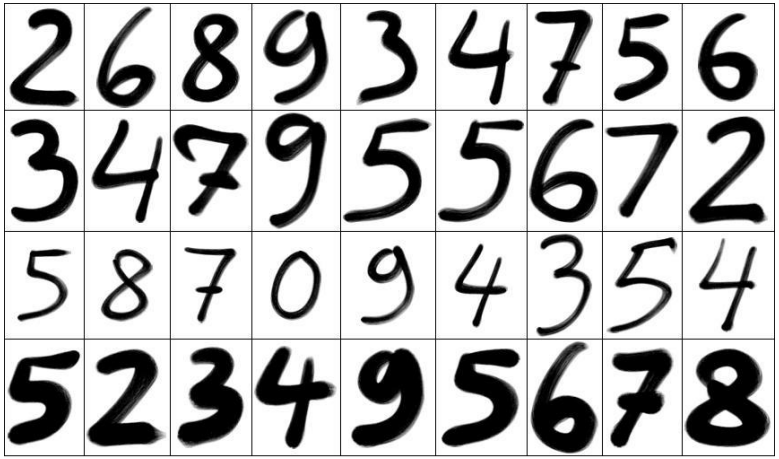


如何开发一个人工智能系统?

- 人工规则

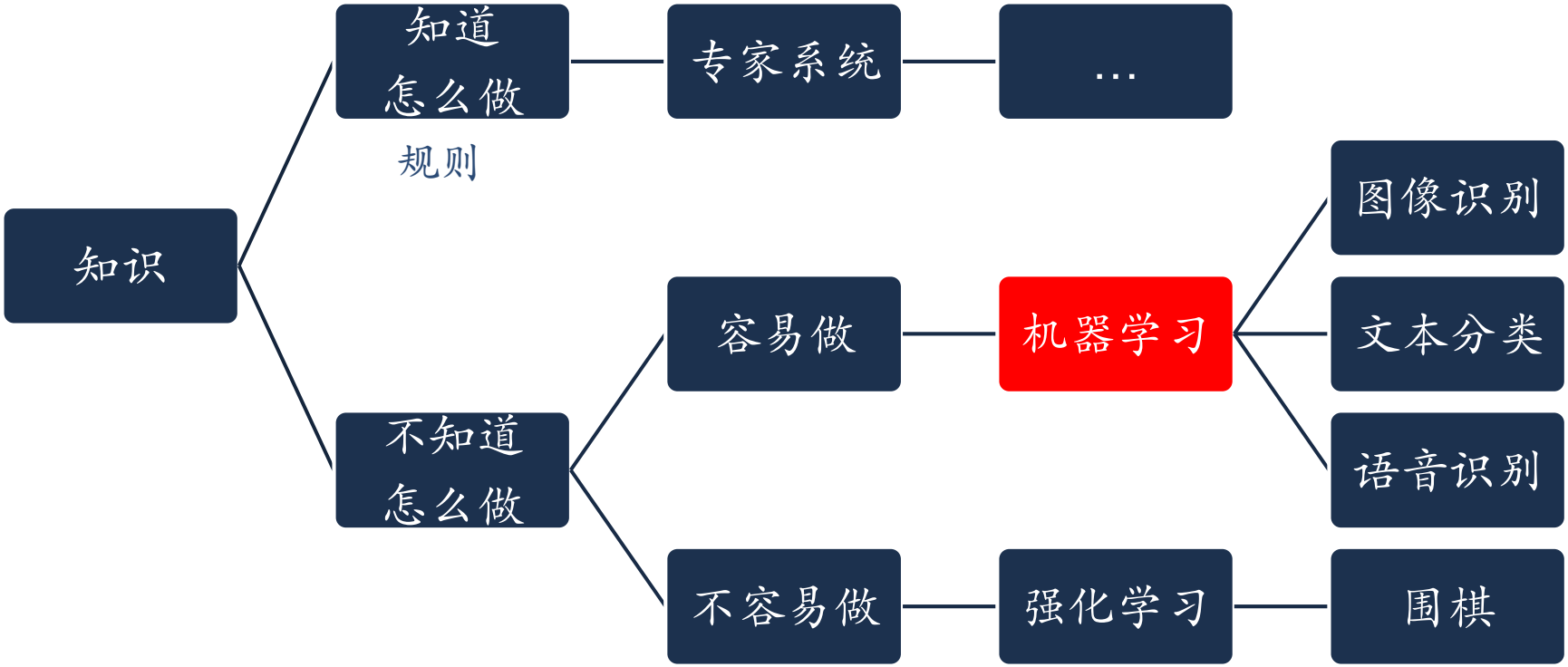


What's the Rule?

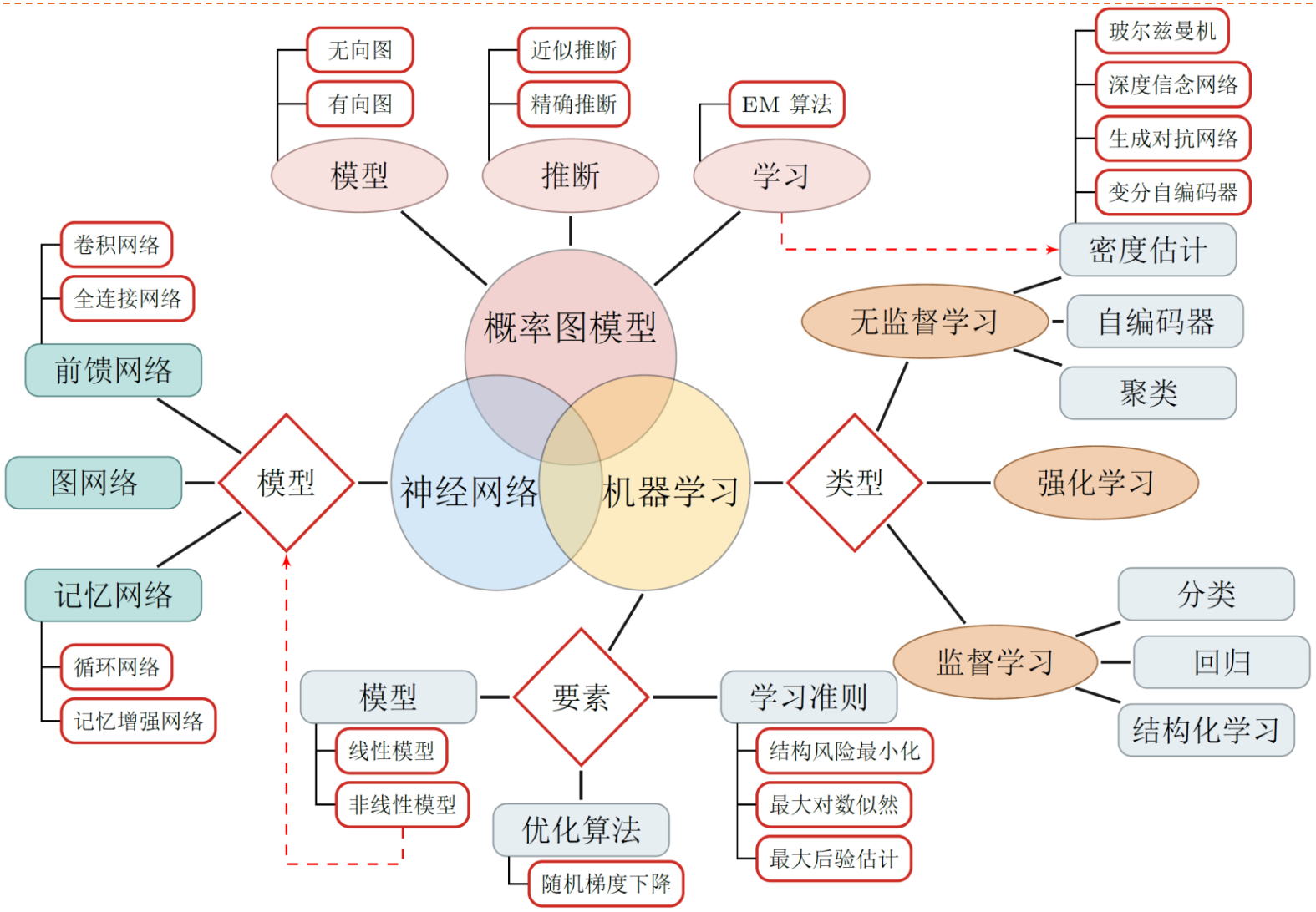


机器学习

如何开发一个人工智能系统?



A Big Picture



机器学习概述

机器学习 ≈ 构建一个映射函数

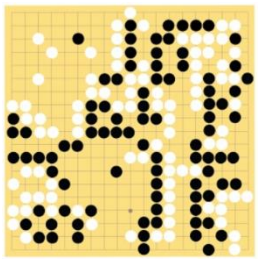
- 语音识别

$f($  $)=$ “你好”

- 图像识别

$f($  $)=$ “9”

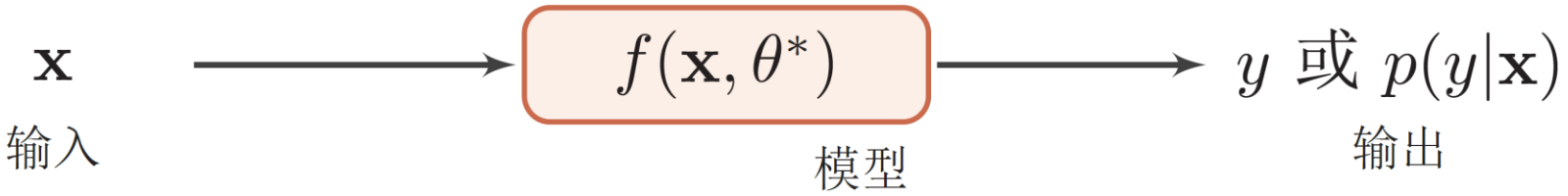
- 围棋

$f($  $)=$ “6-5” （落子位置）

- 机器翻译

$f($ “你好！” $)=$ “Hello!”

机器学习概览



机器学习的三要素

- 模型
  - 线性方法： $f(\mathbf{x}, \theta) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + b$
  - 广义线性方法： $f(\mathbf{x}, \theta) = \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}) + b$
    - 如果$\phi(x)$为可学习的非线性基函数，$f(\mathbf{x},\theta)$就等价于神经网络。

- 学习准则
  - 期望风险

$$\mathcal{R}(f) = \mathbb{E}_{(\mathbf{x},y)\sim p(\mathbf{x},y)}[\mathcal{L}(f(\mathbf{x}), y)],$$

- 优化
  - 梯度下降

常见的机器学习类型

	监督学习	无监督学习	强化学习
训练样本	训练集 $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{N}$	训练集 $\{\mathbf{x}^{n}\}_{n=1}^{N}$	智能体和环境交互的轨迹 τ 和累积奖励 G_τ
优化目标	$y = f(\mathbf{x})$ 或 $p(y\|\mathbf{x})$	$p(\mathbf{x})$ 或带隐变量 $\mathbf{z}$ 的 $p(\mathbf{x}\|\mathbf{z})$	期望总回报 $\mathbb{E}_\tau[G_\tau]$
学习准则	期望风险最小化 最大似然估计	最大似然估计 最小重构错误	策略评估 策略改进

参数学习

- 期望风险未知，通过经验风险近似
  - 训练数据：$\mathcal{D} = \{\mathrm{x}^{(i)}, y^{(i)}\}, i \in [1, N]$

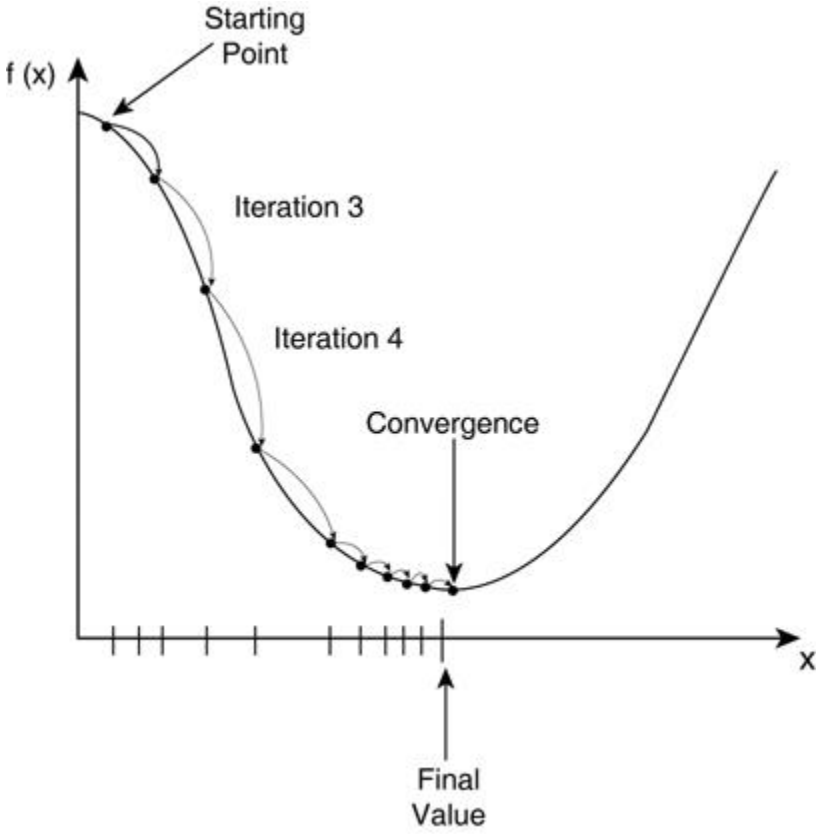
$$\mathcal{R}_{\mathcal{D}}^{emp}(\theta) = \frac{1}{N}\sum_{n=1}^{N}\mathcal{L}(y^{(n)}, f(x^{(n)}, \theta))$$

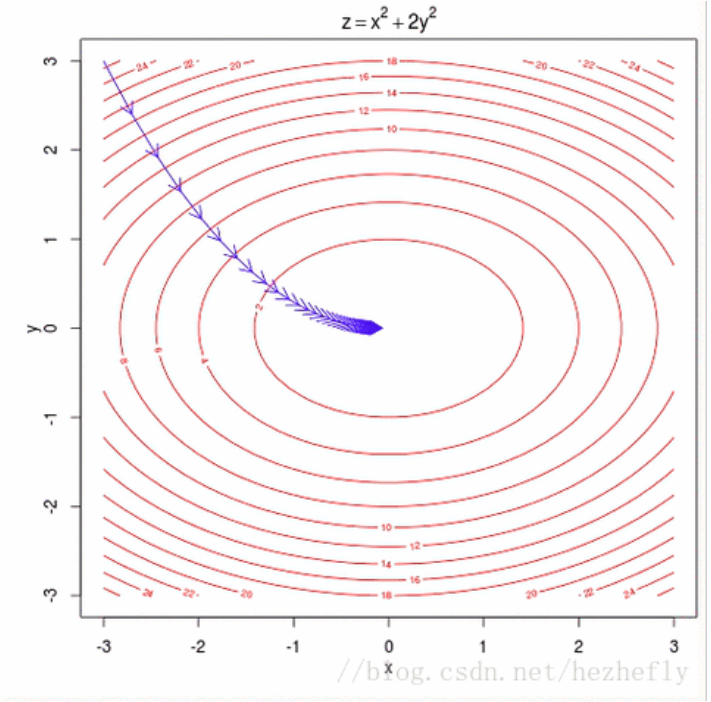
- 经验风险最小化
  - 在选择合适的风险函数后，我们寻找一个参数θ^*，使得经验风险函数最小化。

$$\theta^* = \arg\min_{\theta} \mathcal{R}_{\mathcal{D}}^{emp}(\theta)$$

- 机器学习问题转化成为一个最优化问题

优化：梯度下降法


Starting
Point
f (x)
Iteration 3
Iteration 4
Convergence
x
Final
Value


z = x² + 2y²
y
x
-3
-2
-1
0
1
2
3

随机梯度下降法

输入: 训练集 $\mathcal{D} = \{(\mathbf{x}^{(n)}, y^{(n)})\}, n = 1, \cdots, N$，验证集 $\mathcal{V}$，学习率 α

随机初始化 θ;

repeat

  对训练集 $\mathcal{D}$ 中的样本随机重排序;

  for $n = 1 \cdots N$ **do**

    从训练集 $\mathcal{D}$ 中选取样本 $(\mathbf{x}^{(n)}, y^{(n)})$;

    // 更新参数

    $\theta \leftarrow \theta - \alpha \dfrac{\partial \mathcal{L}\left(\theta; x^{(n)}, y^{(n)}\right)}{\partial \theta}$;

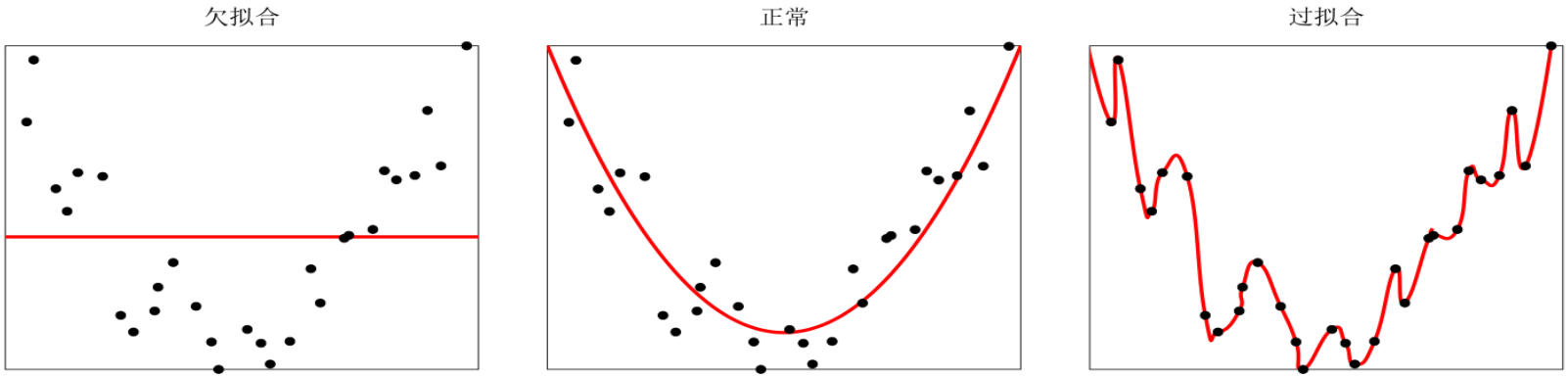
  end

until 模型 $f(\mathbf{x}, \theta)$ 在验证集 $\mathcal{V}$ 上的错误率不再下降;

输出: θ

机器学习 = 优化?

机器学习 = 优化? NO!



过拟合：经验风险最小化原则很容易导致模型在训练集上错误率很低，但是在未知数据上错误率很高。

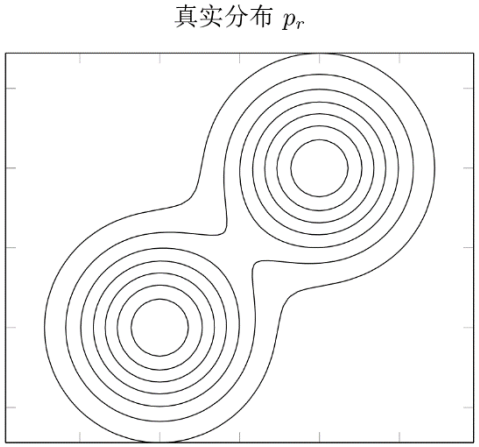
泛化错误

期望风险

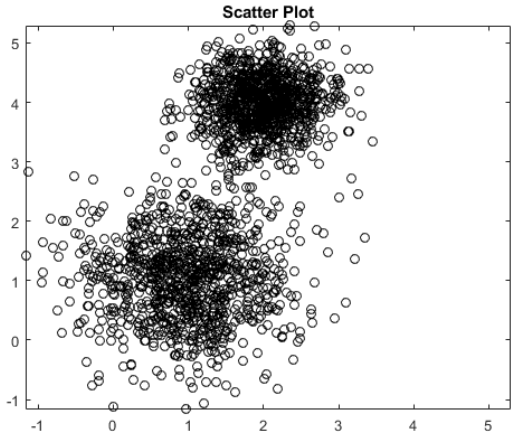
$$\mathcal{R}(f) = \mathbb{E}_{(\mathbf{x},y)\sim p(\mathbf{x},y)}[\mathcal{L}(f(\mathbf{x}), y)],$$

经验风险

$$\mathcal{R}_{\mathcal{D}}^{emp}(\theta) = \frac{1}{N}\sum_{n=1}^{N}\mathcal{L}(y^{(n)}, f(x^{(n)}, \theta))$$



$\neq$



$$\mathcal{G}_{\mathcal{D}}(f) = \mathcal{R}(f) - \mathcal{R}_{\mathcal{D}}^{emp}(f)$$

泛化错误

PAC学习

Probably Approximately Correct

- 根据大数定律，当训练集大小|D|趋向无穷大时，泛化错误趋向于0，即经验风险趋近于期望风险。

$$\lim_{|\mathcal{D}|\to\infty} \mathcal{R}(f) - \mathcal{R}_{\mathcal{D}}^{emp}(f) = 0$$

- PAC学习

$$P\left((\mathcal{R}(f) - \mathcal{R}_{\mathcal{D}}^{emp}(f)) \leq \epsilon\right) \geq 1 - \delta$$

近似正确，$0 < \epsilon < 0.5$

可能，$0 < \delta < 0.5$

样本复杂度

- 如果固定ϵ,δ，可以反过来计算出样本复杂度为

$$n(\epsilon, \delta) \geq \frac{1}{2\epsilon^2}(\ln |\mathcal{F}| + \ln \frac{2}{\delta})$$

  - 其中$|\mathcal{F}|$为假设空间的大小，可以用Rademacher复杂性或VC维来衡量。

- PAC学习理论可以帮助分析一个机器学习方法在什么条件下可以学习到一个近似正确的分类器。

- 如果希望模型的假设空间越大，泛化错误越小，其需要的样本数量越多。

如何减少泛化错误?

优化
经验风险最小

正则化
降低模型复杂度

正则化（regularization）

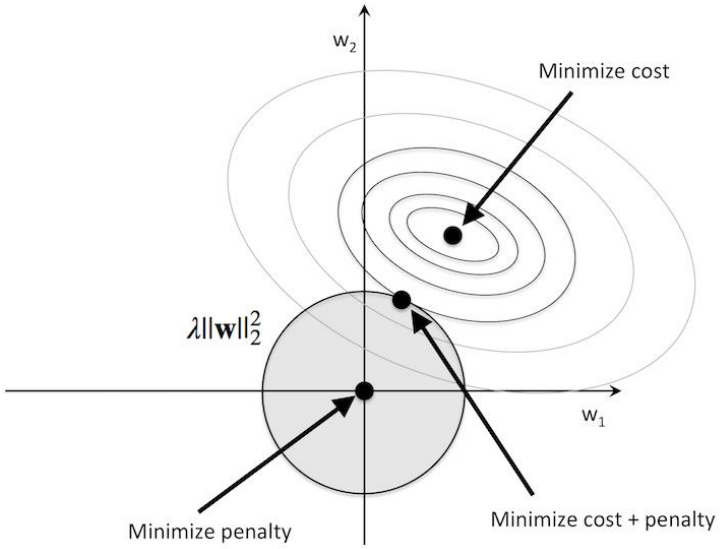
所有损害优化的方法都是正则化。

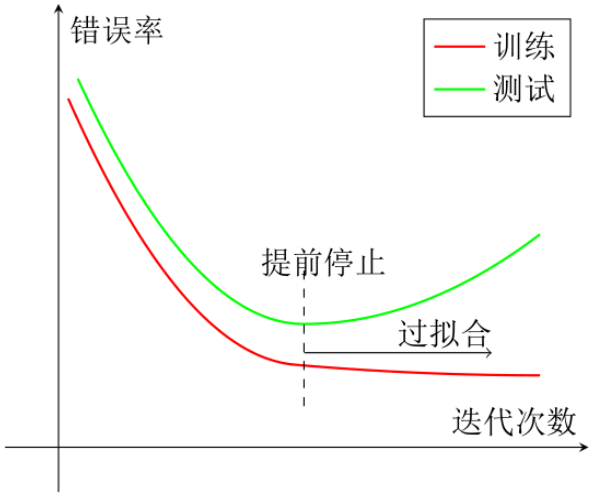
增加优化约束

L1/L2约束、数据增强

干扰优化过程

权重衰减、随机梯度下降、提前停止





如何选择一个合适的模型?

- 模型选择
  - 拟合能力强的模型一般复杂度会比较高，容易过拟合。
  - 如果限制模型复杂度，降低拟合能力，可能会欠拟合。
- 偏差与方差分解
  - 期望错误可以分解为

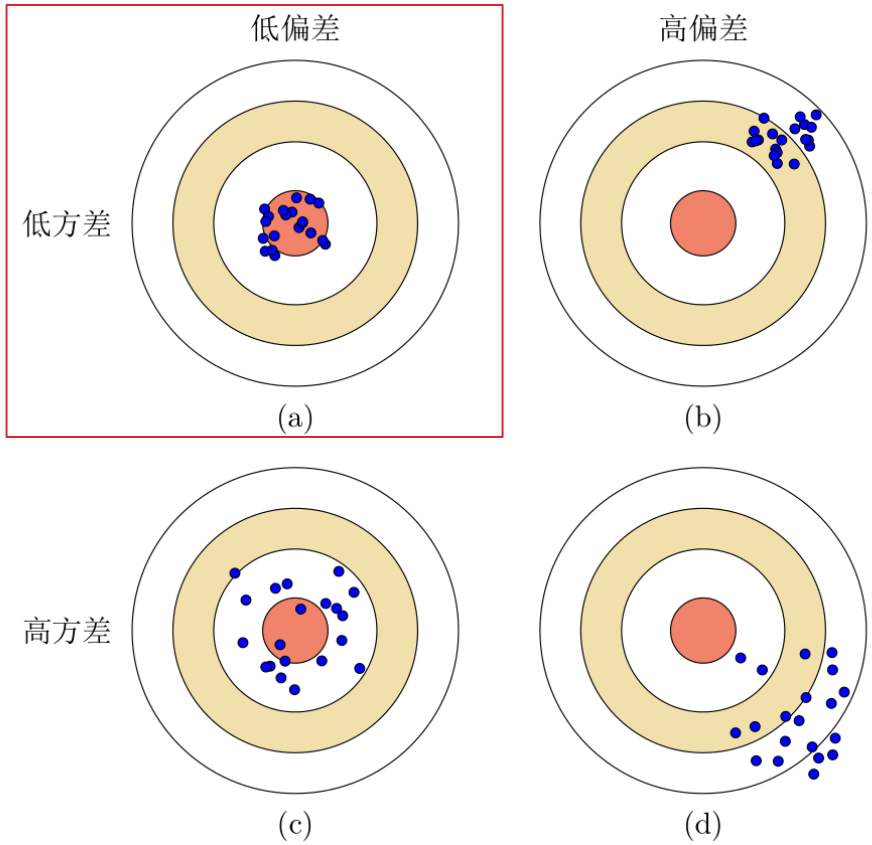
$$\mathcal{R}(f) = (\text{bias})^2 + \text{variance} + \varepsilon.$$

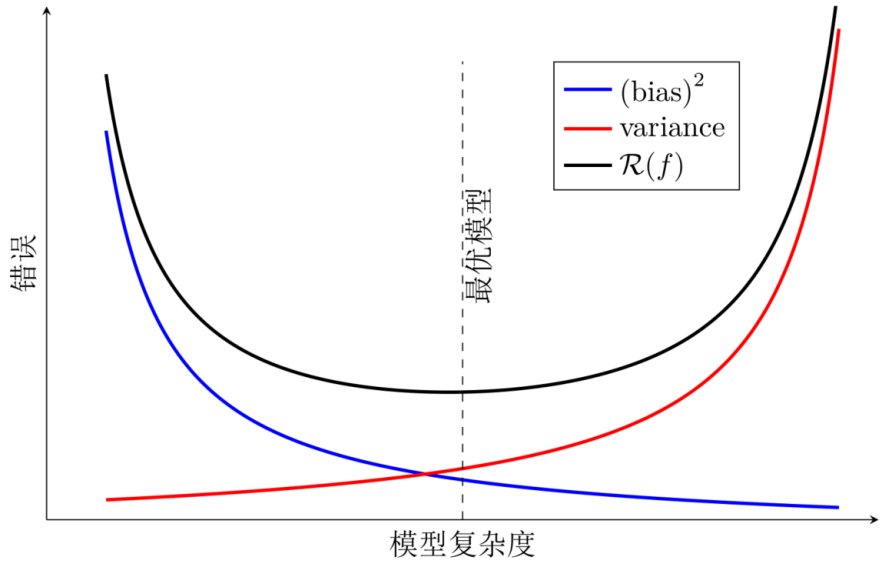
$$\mathbb{E}_{\mathbf{x}}\left[\left(\mathbb{E}_{\mathcal{D}}\left[f_{\mathcal{D}}(\mathbf{x})\right] - f^*(\mathbf{x})\right)^2\right]$$

$$\mathbb{E}_{\mathbf{x}}\left[\mathbb{E}_{\mathcal{D}}\left[\left(f_{\mathcal{D}}(\mathbf{x}) - \mathbb{E}_{\mathcal{D}}\left[f_{\mathcal{D}}(\mathbf{x})\right]\right)^2\right]\right]$$

$$\mathbb{E}_{(\mathbf{x},y)\sim p_r(\mathbf{x},y)}\left[\left(y - f^*(\mathbf{x})\right)^2\right]$$

模型选择：偏差与方差


低偏差
高偏差
低方差
高方差
(a)
(b)
(c)
(d)


$(\text{bias})^2$
variance
$\mathcal{R}(f)$
错误
最优模型
模型复杂度

集成模型：有效的降低方差的方法

- 集成模型

$$f^{(c)}(\mathbf{x}) = \frac{1}{M}\sum_{m=1}^{M} f_m(\mathbf{x})$$

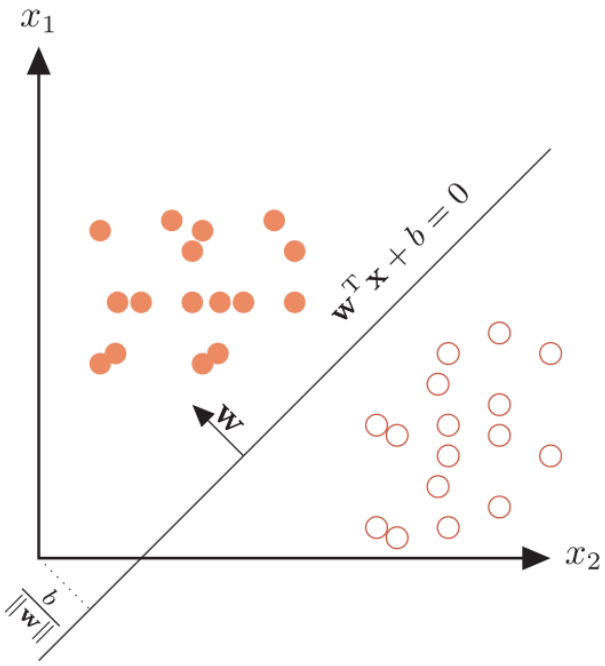
  - 通过多个高方差模型的平均来降低方差。

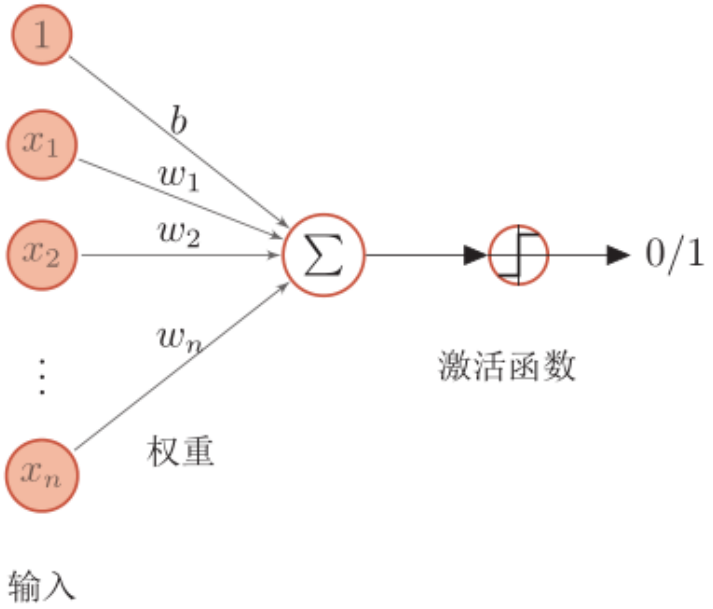
- 集成模型的期望错误大于等于所有模型的平均期望错误的1/M，小于等于所有模型的平均期望错误。

$$\bar{\mathcal{R}}(f) \geq \mathcal{R}(f^{(c)}) \geq \frac{1}{M}\bar{\mathcal{R}}(f)$$

线性模型

线性模型


x_1
x_2
$\mathbf{w}^T\mathbf{x}+b=0$
$\mathbf{w}$
$\frac{b}{\|\mathbf{w}\|}$

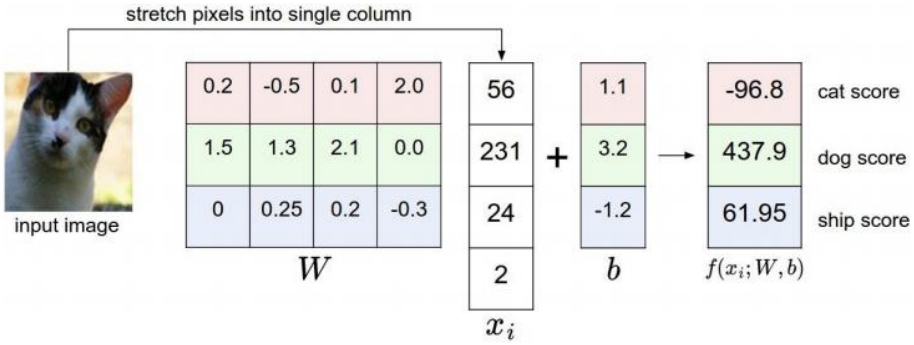

1
x_1
x_2
x_n
b
w_1
w_2
w_n
$\sum$
0/1
激活函数
权重
输入

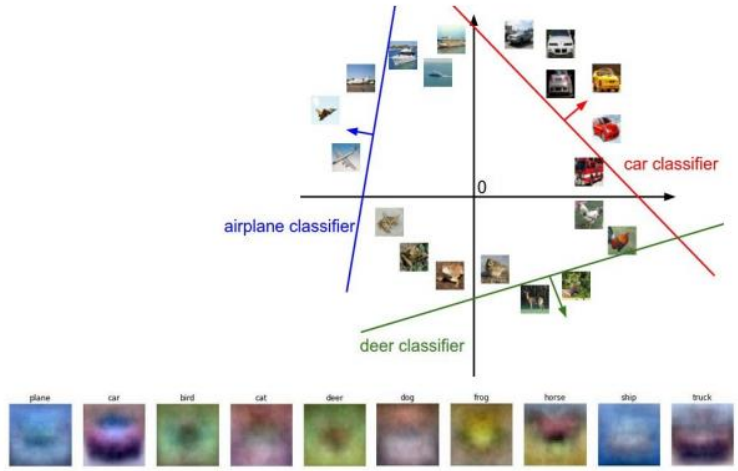
应用：图像分类

[32x32x3]
array of numbers 0...1
(3072 numbers total)

image parameters
f(x,W)

10 numbers, indicating
class scores





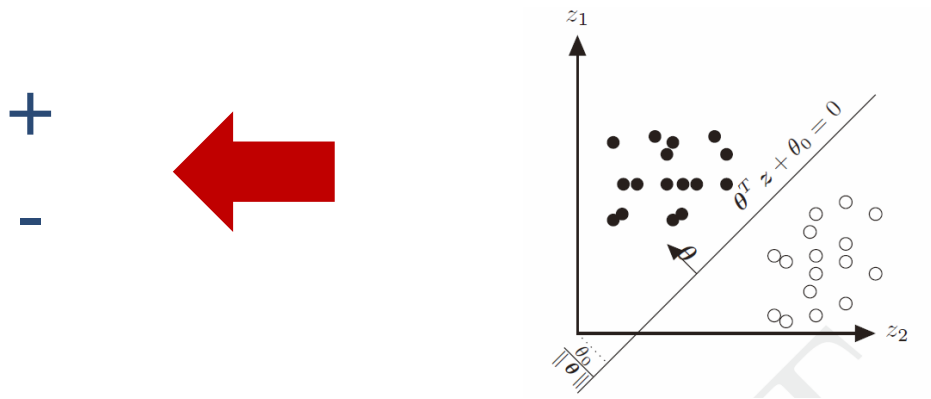
应用：文本分类

根据文本内容来判断文本的相应类别

D_1：“我喜欢读书”

D_2：“我讨厌读书”

	我	喜欢	讨厌	读书
D_1	1	1	0	1
D_2	1	0	1	1



感知器

- 模型

$$g(\mathbf{x}, \mathbf{w}) = \begin{cases} +1 & \text{当} \quad \mathbf{w}^{\mathrm{T}}\mathbf{x} > 0, \\ -1 & \text{当} \quad \mathbf{w}^{\mathrm{T}}\mathbf{x} < 0. \end{cases}$$

- 学习准则

$$\mathcal{L}(\mathbf{w}; \mathbf{x}, y) = \max(0, -y\mathbf{w}^{\mathrm{T}}\mathbf{x}).$$

- 优化：随机梯度下降

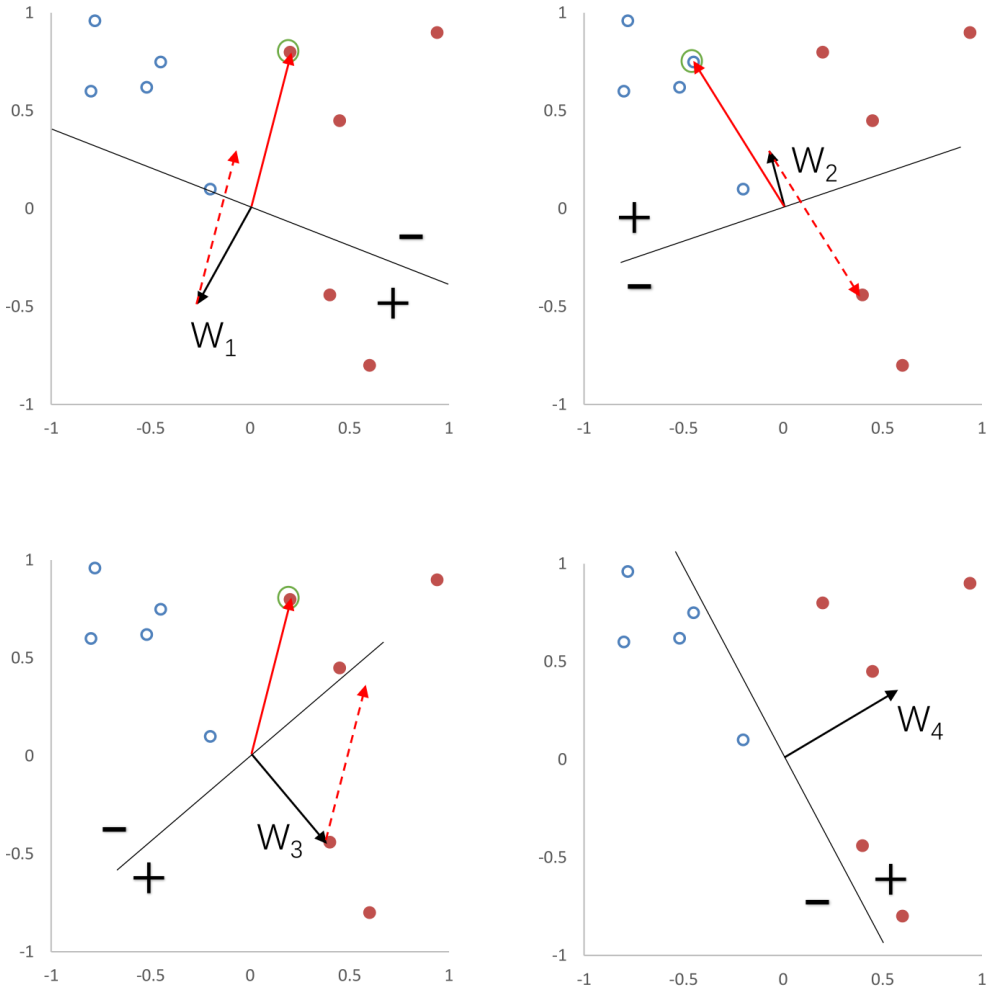
$$\frac{\partial \mathcal{L}(\mathbf{w}; \mathbf{x}, y)}{\partial \mathbf{w}} = \begin{cases} 0 & \text{当} \quad y\mathbf{w}^{\mathrm{T}}\mathbf{x} > 0, \\ -y\mathbf{x} & \text{当} \quad y\mathbf{w}^{\mathrm{T}}\mathbf{x} < 0. \end{cases}$$

两类感知器算法

输入: 训练集 $\{(\mathbf{x}^{(n)}, y^{(n)})\}, n = 1, \cdots, N$，迭代次数 T

初始化：$\mathbf{w}_0 \leftarrow 0, k \leftarrow 0$;
for $t = 1 \cdots T$ **do**
  随机对训练样本进行随机排序;
  for $n = 1 \cdots N$ **do**
    选取一个样本 $(\mathbf{x}^{(n)}, y^{(n)})$;
    if $\mathbf{w}_k^{\mathrm{T}}(y^{(n)}\mathbf{x}^{(n)}) \leq 0$ **then**
      $\mathbf{w}_{k+1} \leftarrow \mathbf{w}_k + y^{(n)}\mathbf{x}^{(n)}$;
      $k \leftarrow k + 1$;
    end
  end
end
输出: $\mathbf{w}_k$

感知器参数学习的更新过程



直接建模条件概率$p_\theta(y|x)$

- 真实条件概率$p_r(y|x)$
- 模型预测条件概率$p_\theta(y|x)$
- 如何衡量两个条件分布的差异?

- KL散度

$$\mathrm{D}_{\mathrm{kl}}(p_r(y|x)||p_\theta(y|x)) = \sum_{y=1}^{C} p_r(y|x) \log \frac{p_r(y|x)}{p_\theta(y|x)}$$

$$\propto -\sum_{y=1}^{C} p_r(y|x) \log p_\theta(y|x)$$

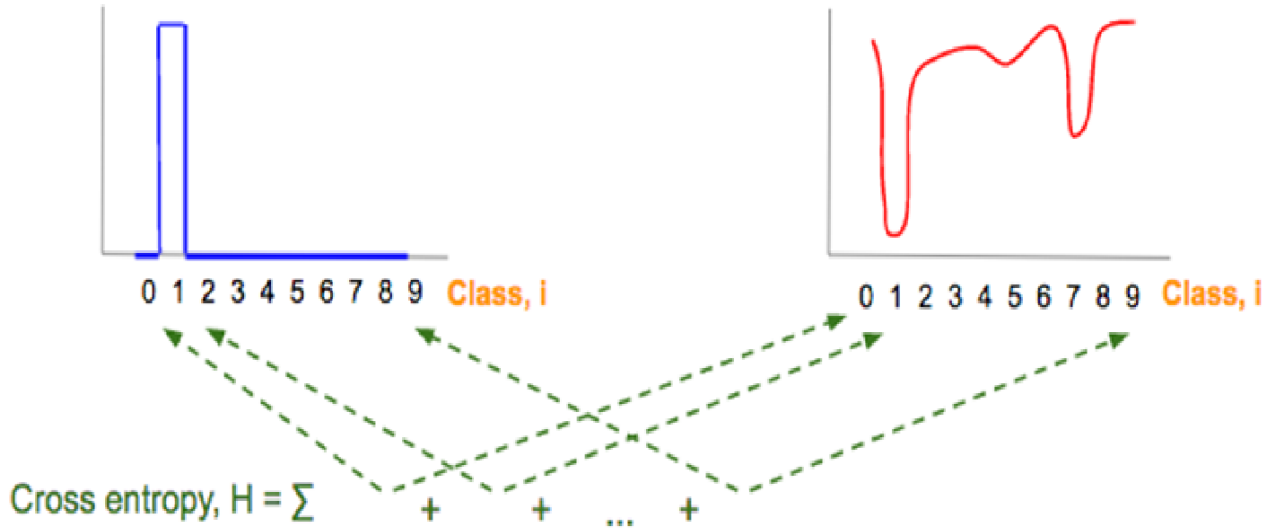
交叉熵损失

交叉熵损失

$$-\sum_{y=1}^{C} p_r(y|x) \log p_\theta(y|x)$$

真实概率$p_r(y|x)$

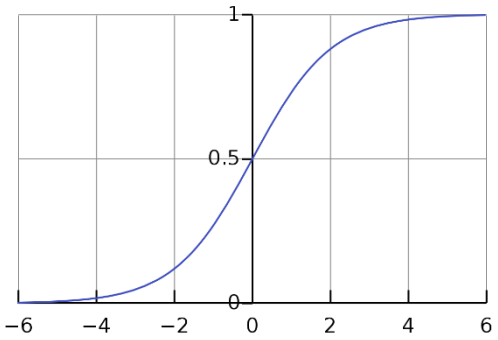
预测概率的负对数$-\log p_\theta(y|x)$



Logistic回归

- 模型

$$p(y=1|\mathbf{x}) = \sigma(\mathbf{w}^{\mathrm{T}}\mathbf{x}) \triangleq \frac{1}{1+\exp(-\mathbf{w}^{\mathrm{T}}\mathbf{x})}$$



- 学习准则：交叉熵

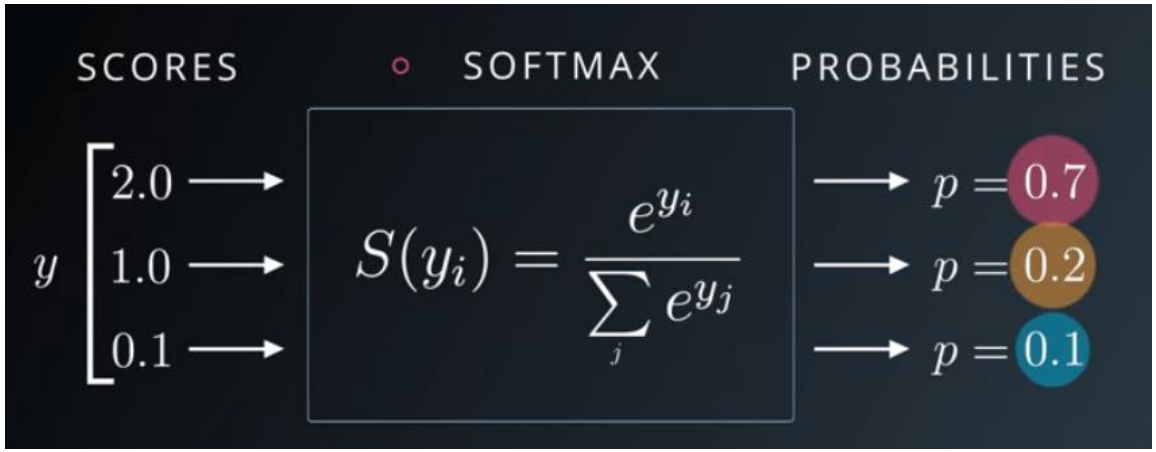
$$\mathcal{R}(\mathbf{w}) = -\frac{1}{N}\sum_{n=1}^{N}\left(y^{(n)}\log\hat{y}^{(n)} + (1-y^{(n)})\log(1-\hat{y}^{(n)})\right)$$

- 优化：梯度下降

$$\frac{\partial\mathcal{R}(\mathbf{w})}{\partial\mathbf{w}} = -\frac{1}{N}\sum_{n=1}^{N}\mathbf{x}^{(n)}\left(y^{(n)}-\hat{y}^{(n)}\right)$$

扩展到多类

- Softmax函数



Softmax回归

- 模型：
$$P(y=c|\mathbf{x}) = \text{softmax}(\mathbf{w}_c^{\mathrm{T}}\mathbf{x}) = \frac{\exp(\mathbf{w}_c^{\top}\mathbf{x})}{\sum_{i=1}^{C}\exp(\mathbf{w}_i^{\top}\mathbf{x})}.$$

- 学习准则：交叉熵
$$\mathcal{R}(W) = -\frac{1}{N}\sum_{n=1}^{N}(\mathbf{y}^{(n)})^{\mathrm{T}}\log\hat{\mathbf{y}}^{(n)}$$

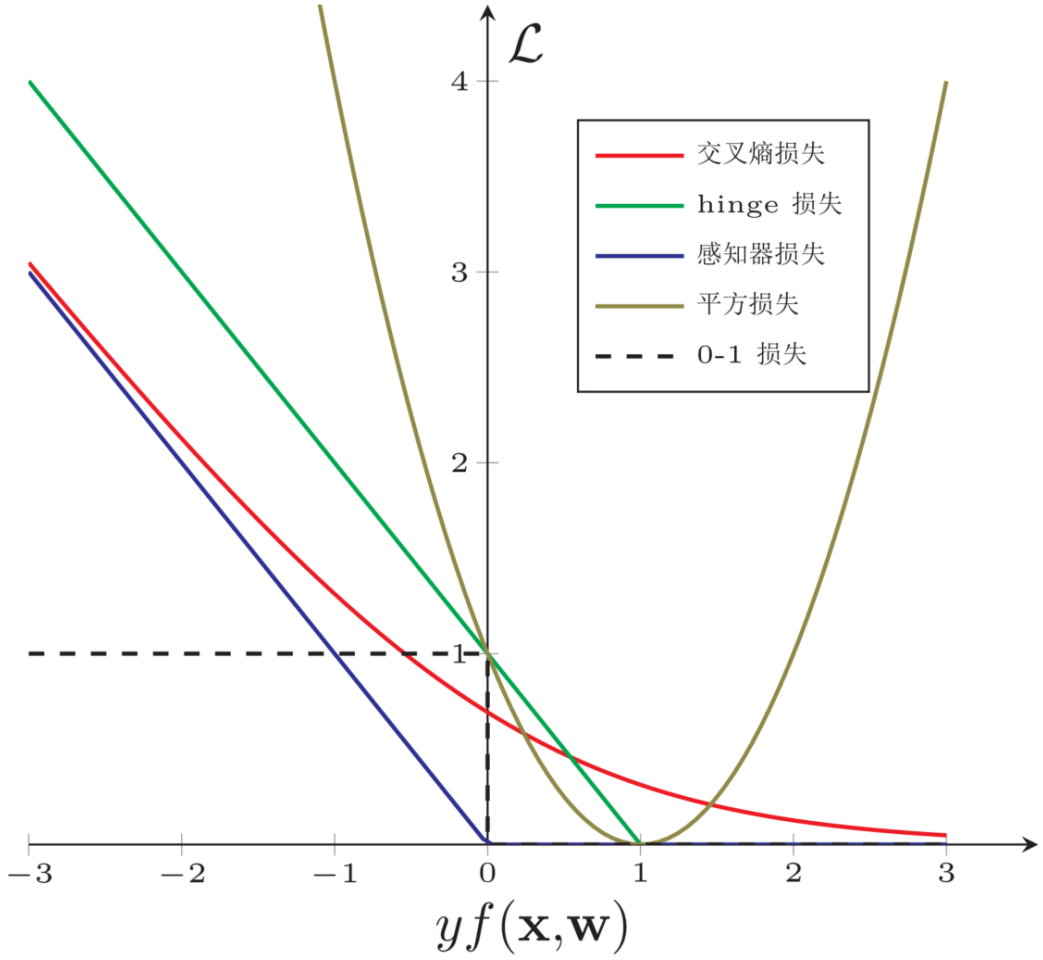
- 优化：梯度下降
$$\frac{\partial\mathcal{R}(W)}{\partial W} = -\frac{1}{N}\sum_{n=1}^{N}\mathbf{x}^{(n)}\left(\mathbf{y}^{(n)}-\hat{\mathbf{y}}^{(n)}\right)^{\mathrm{T}}$$

几种不同的线性模型对比

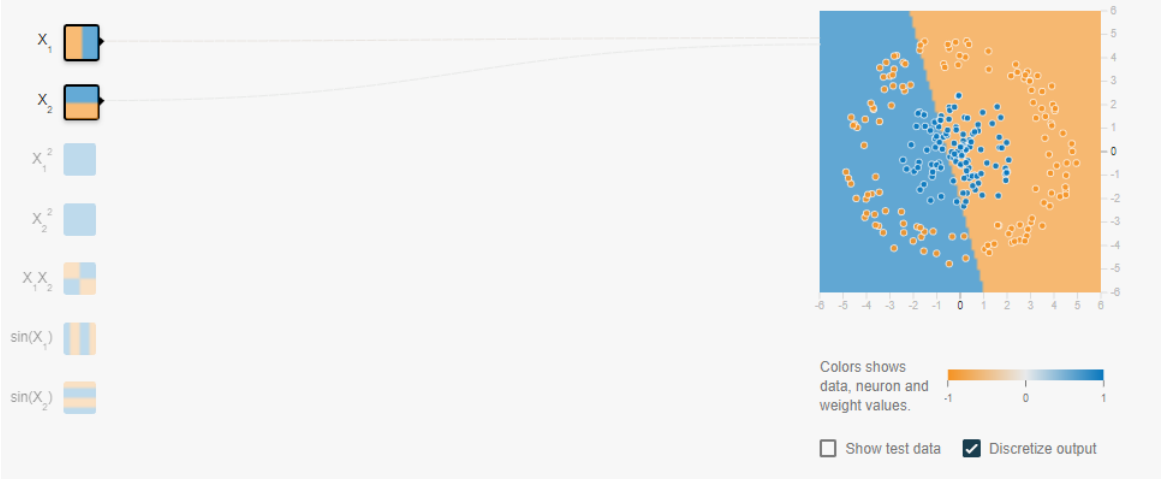
	激活函数	损失函数	优化方法
线性回归	-	$(y - \mathbf{w}^{\mathrm{T}}\mathbf{x})^2$	最小二乘、梯度下降
Logistic 回归	logistic 函数	$\mathbf{y} \log \sigma(\mathbf{w}^{\mathrm{T}}\mathbf{x})$	梯度下降
Softmax 回归	softmax 函数	$\mathbf{y} \log \mathrm{softmax}(W^{\mathrm{T}}\mathbf{x})$	梯度下降
感知器	阶跃函数	$\max(0, -y\mathbf{w}^{\mathrm{T}}\mathbf{x})$	随机梯度下降
支持向量机	阶跃函数	$\max(0, 1 - y\mathbf{w}^{\mathrm{T}}\mathbf{x})$	二次规划、SMO 等

在logistic回归和softmax回归中，y为类别的one-hot向量表示；
在感知器和支持向量机中，y为{+1,−1}

不同损失函数的对比

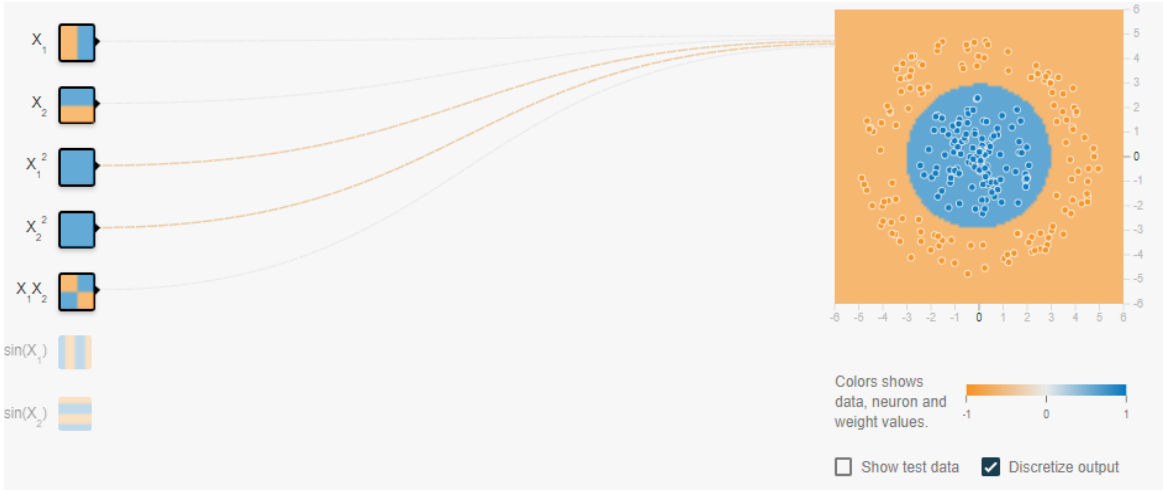

$\mathcal{L}$
4
3
2
1
−3
−2
−1
0
1
2
3
$yf(\mathbf{x},\mathbf{w})$
交叉熵损失
hinge 损失
感知器损失
平方损失
0-1 损失

如何处理非线性可分问题?



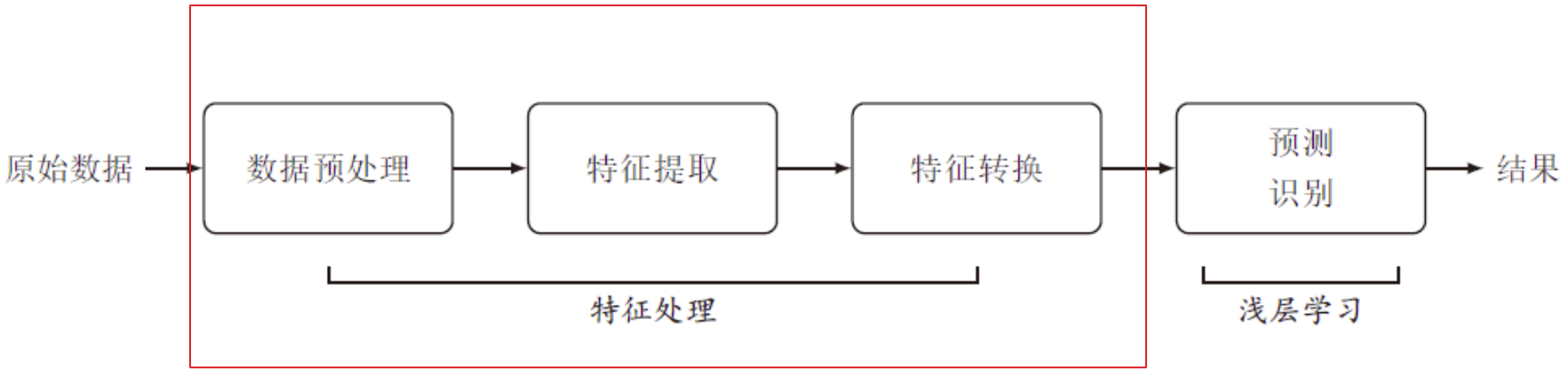
http://playground.tensorflow.org

增加非线性特征



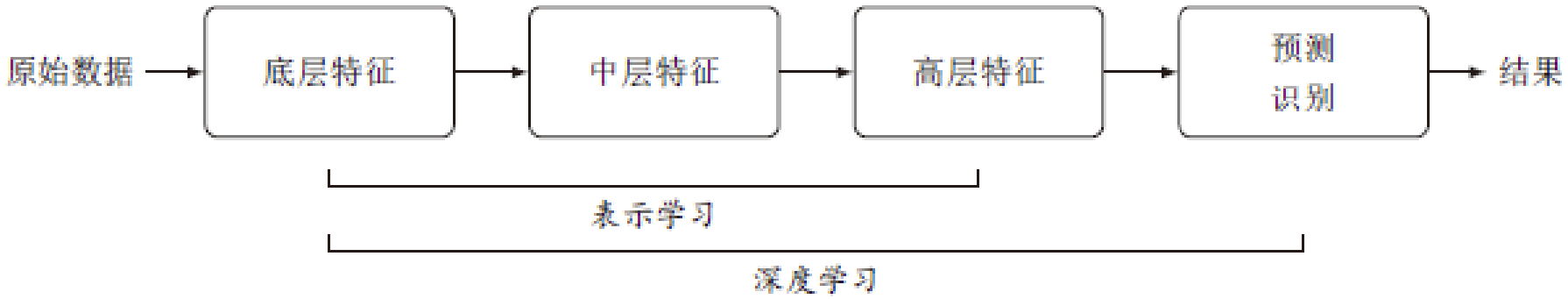
特征工程问题

- 在实际应用中，特征往往比分类器更重要
  - 预处理：经过数据的预处理，如去除噪声等。比如在文本分类中，去除停用词等。
  - 特征提取：从原始数据中提取一些有效的特征。比如在图像分类中，提取边缘、尺度不变特征变换特征等。
  - 特征转换：对特征进行一定的加工，比如降维和升维。降维包括
    - 特征抽取（Feature Extraction）： PCA、LDA
    - 特征选择（Feature Selection）： 互信息、TF-IDF



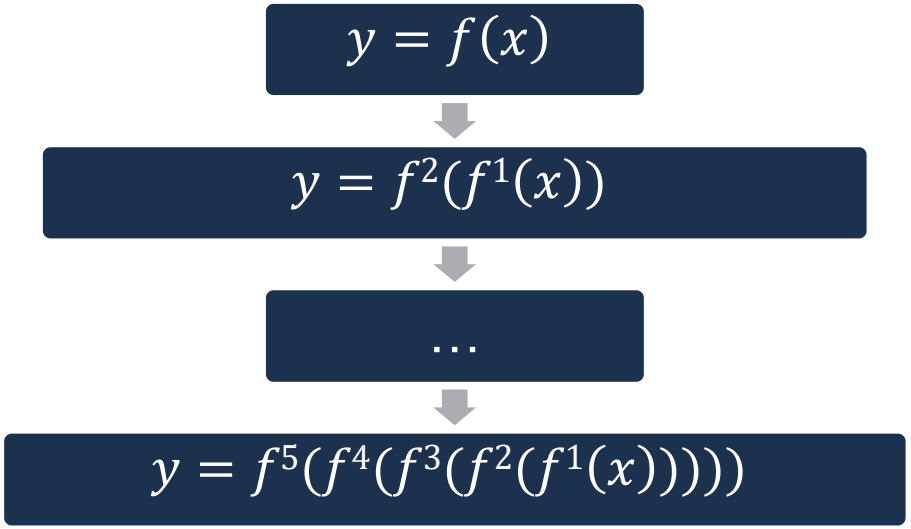
深度学习

- 深度学习=表示学习+浅层学习



- 难点：贡献度分配问题

深度学习的数学描述



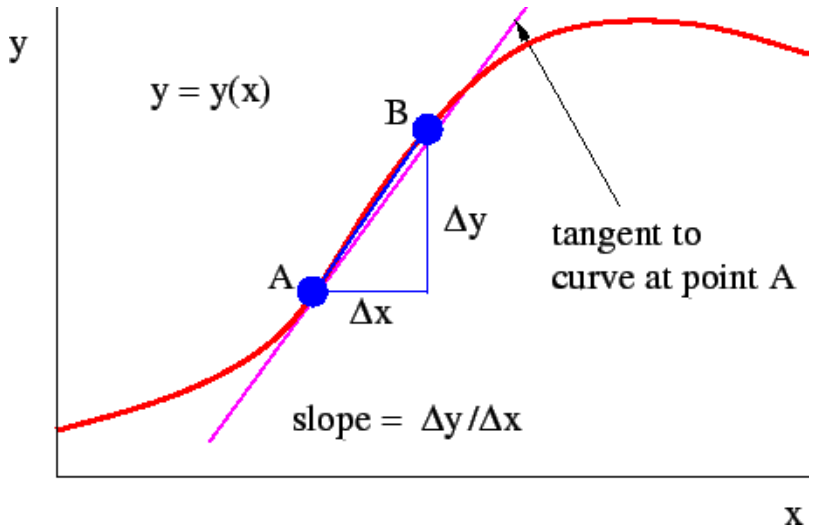
浅层学习

深度学习

当$f^l(x) = \sigma(W^l x)$时为神经网络!

如果解决贡献度分配问题?

- 偏导数



- 贡献度

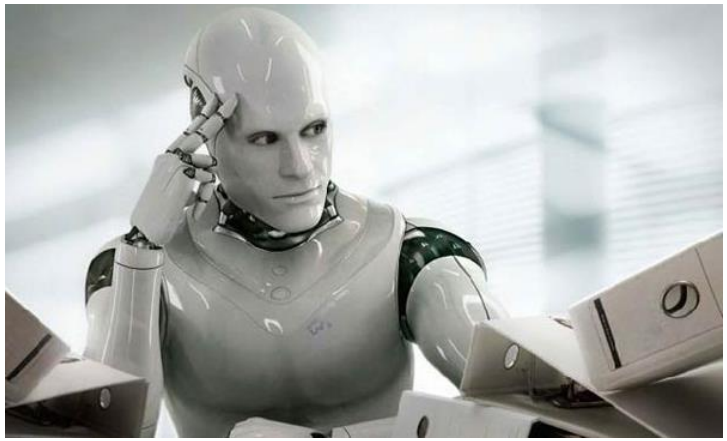
$$\frac{\partial y}{\partial W^l} = \frac{y(W^l + \Delta Wl) - y(Wl)}{\Delta Wl}$$

- 深度学习天然不是神经网络，但神经网络天然是深度学习！

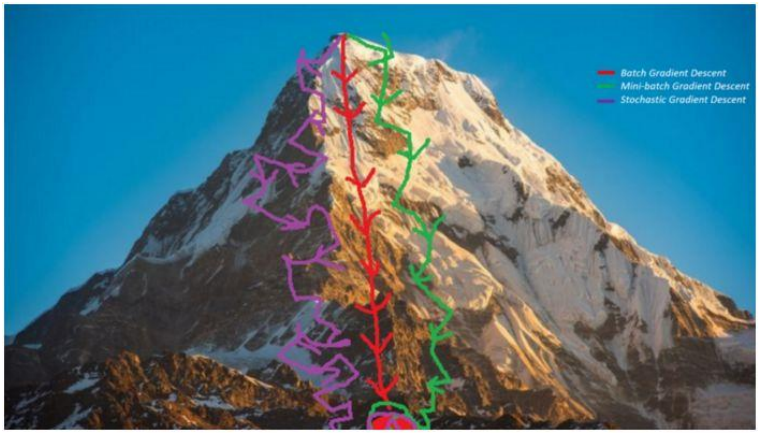
深度学习

模型：神经网络
学习准则：交叉熵损失等
优化：随机梯度下降

别人眼中的深度学习

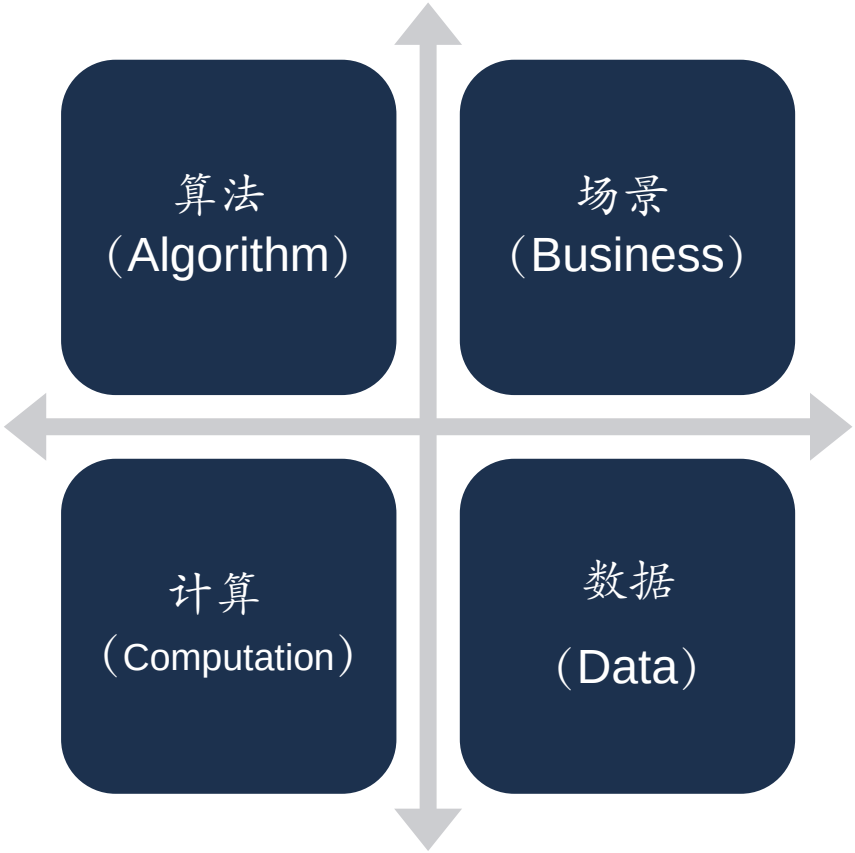


实际上的深度学习



为什么现在才显式威力?

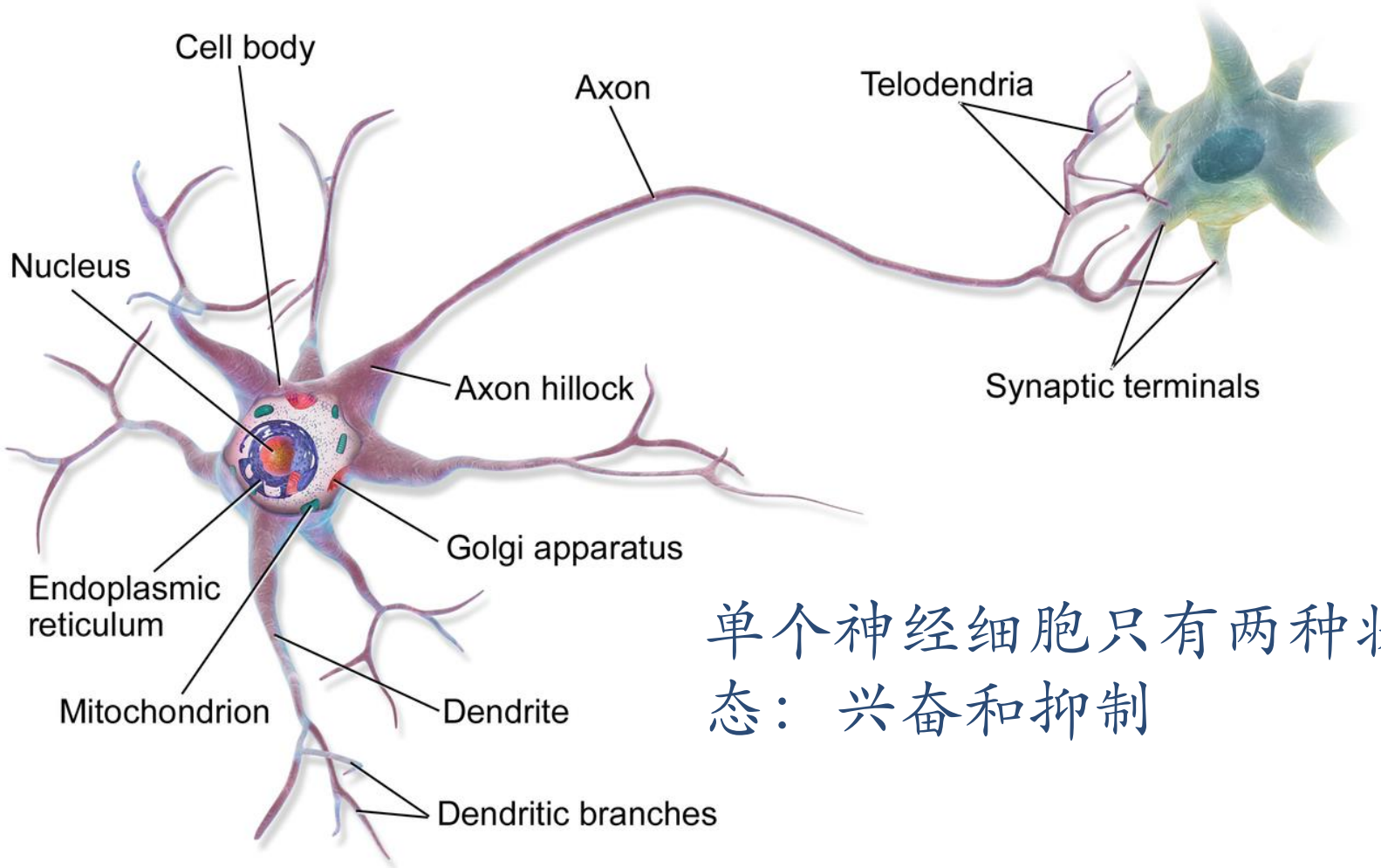
- 缺点
  - 参数过多
    - 影响训练
  - 非凸优化问题
    - 存在局部最优而非全局最优解
  - 梯度消失
    - 下层参数比较难调
  - 参数解释性差



神经网络

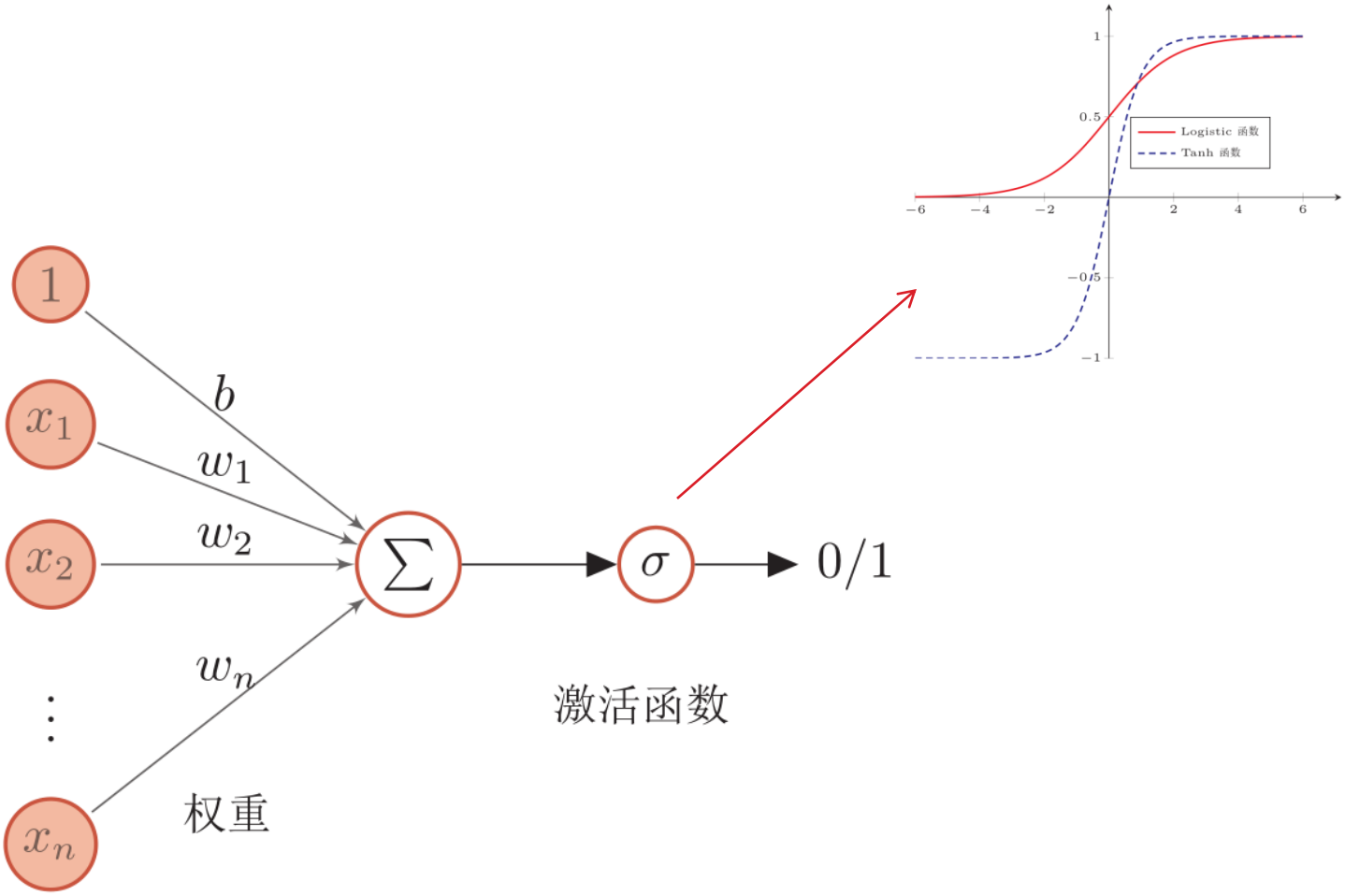
生物神经元

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



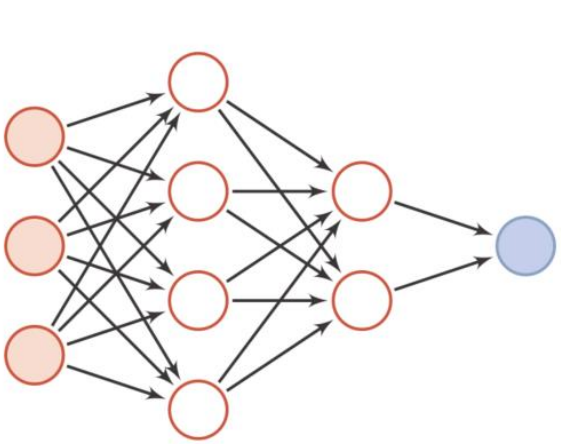
人工神经网络

- 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：

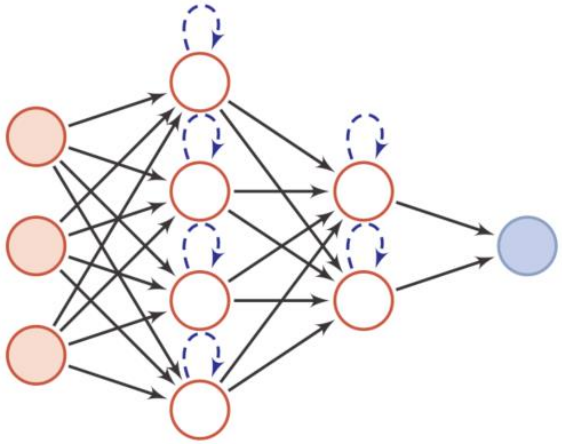
- 神经元的激活规则
  - 主要是指神经元输入到输出之间的映射关系，一般为非线性函数。
- 网络的拓扑结构
  - 不同神经元之间的连接关系。
- 学习算法
  - 通过训练数据来学习神经网络的参数。

人工神经网络

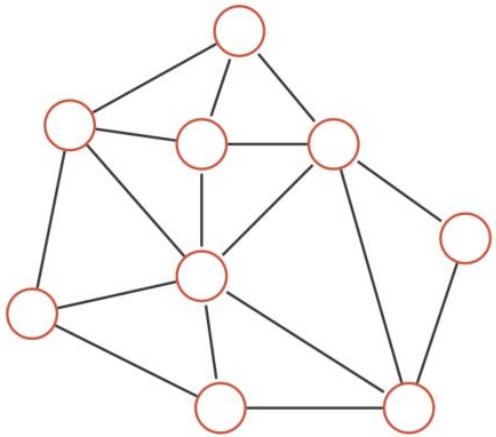
- 人工神经网络由神经元模型构成，这种由许多神经元组成的信息处理网络具有并行分布结构。



(a) 前馈网络



(b) 反馈网络

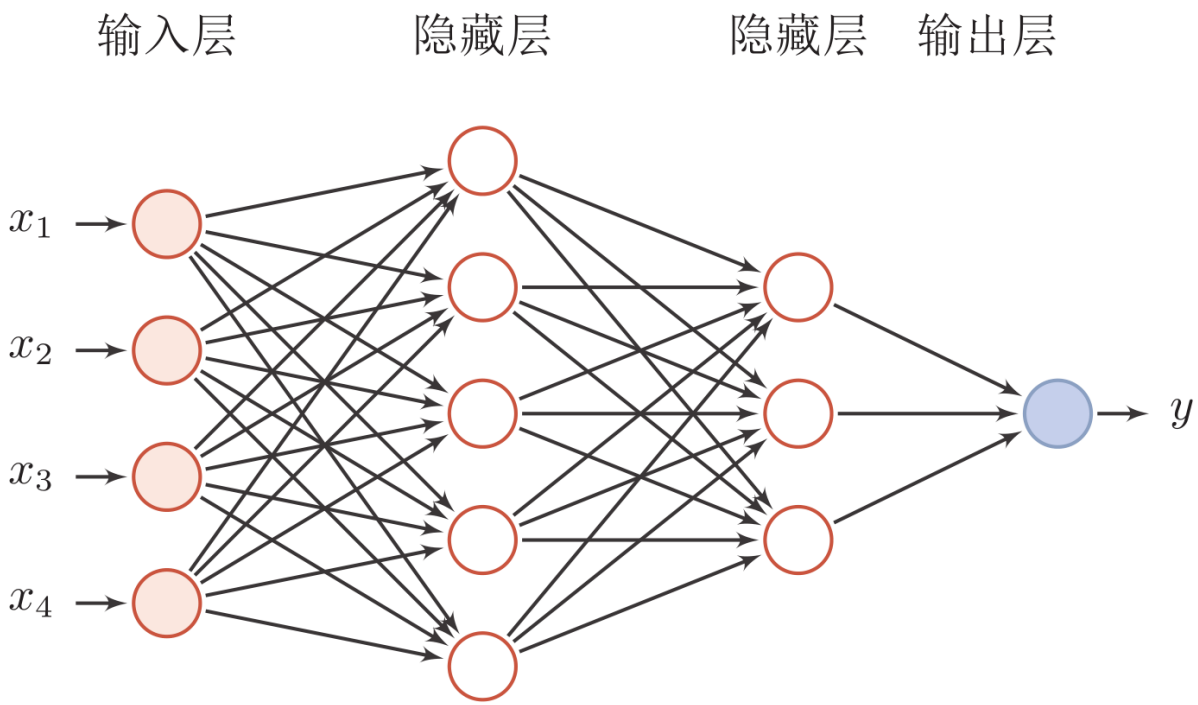


(c) 图网络

前馈神经网络

网络结构

- 在前馈神经网络中，各神经元分别属于不同的层。整个网络中无反馈，信号从输入层向输出层单向传播，可用一个有向无环图表示。



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。

一个两层的神经网络可以模拟任何函数。

应用到机器学习

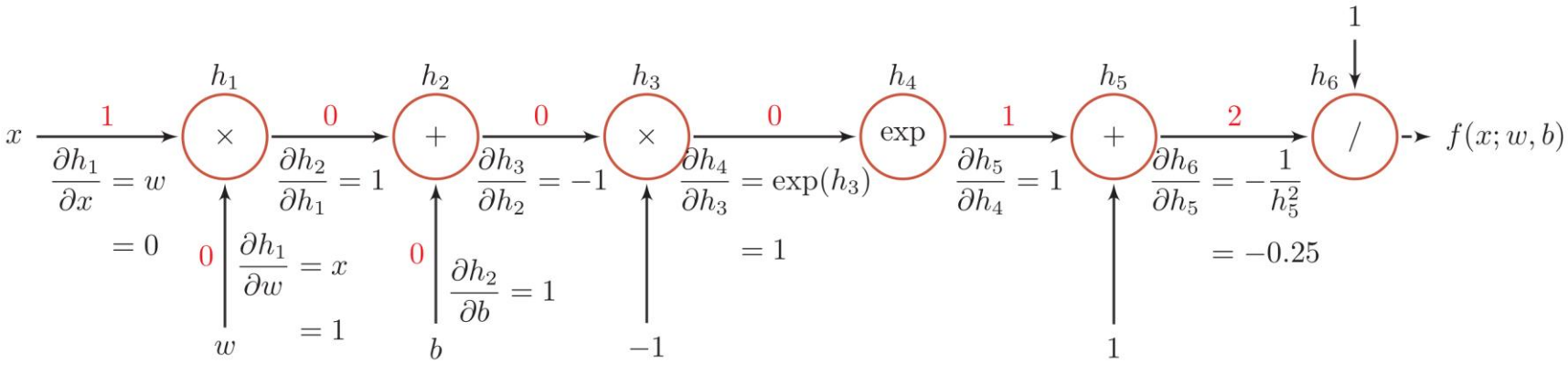
- 模型
  - $y = f^5(f^4(f^3(f^2(f^1(x)))))$
- 学习准则
  - $\mathrm{L}(y, y^*)$
- 优化
  - 梯度下降

链式法则，可以自动计算！

$$\frac{\partial \mathrm{L}(y, y^*)}{\partial f^1} = \frac{\partial f^2}{\partial f^1} \times \frac{\partial f^3}{\partial f^2} \times \frac{\partial f^4}{\partial f^3} \times \frac{\partial f^5}{\partial f^4} \times \frac{\partial \mathrm{L}(y, y^*)}{\partial f^5}$$

计算图与自动微分

- 复合函数f(x;w,b) = σ(wx + b)的计算图

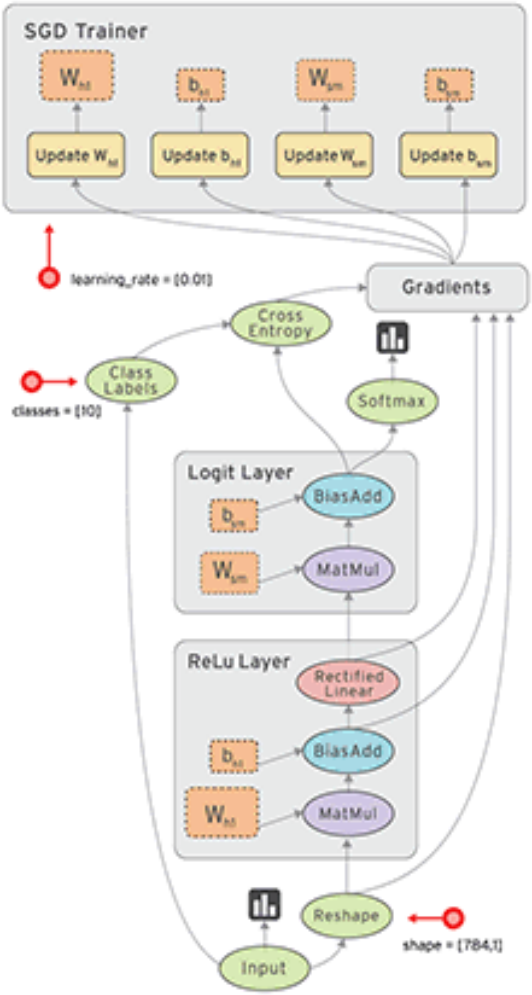


- 链式法则 $$\frac{\partial f(x;w,b)}{\partial w} = \frac{\partial f(x;w,b)}{\partial h_6}\frac{\partial h_6}{\partial h_5}\frac{\partial h_5}{\partial h_4}\frac{\partial h_4}{\partial h_3}\frac{\partial h_3}{\partial h_2}\frac{\partial h_2}{\partial h_1}\frac{\partial h_1}{\partial w}$$

反向传播算法只是自动微分的一种特殊形式。

TensorFlow中的计算图示例

- 计算图是一个用来描述数学计算的有向图(有向无环图)
- 节点代表操作(Operation)
- 边代表Tensor



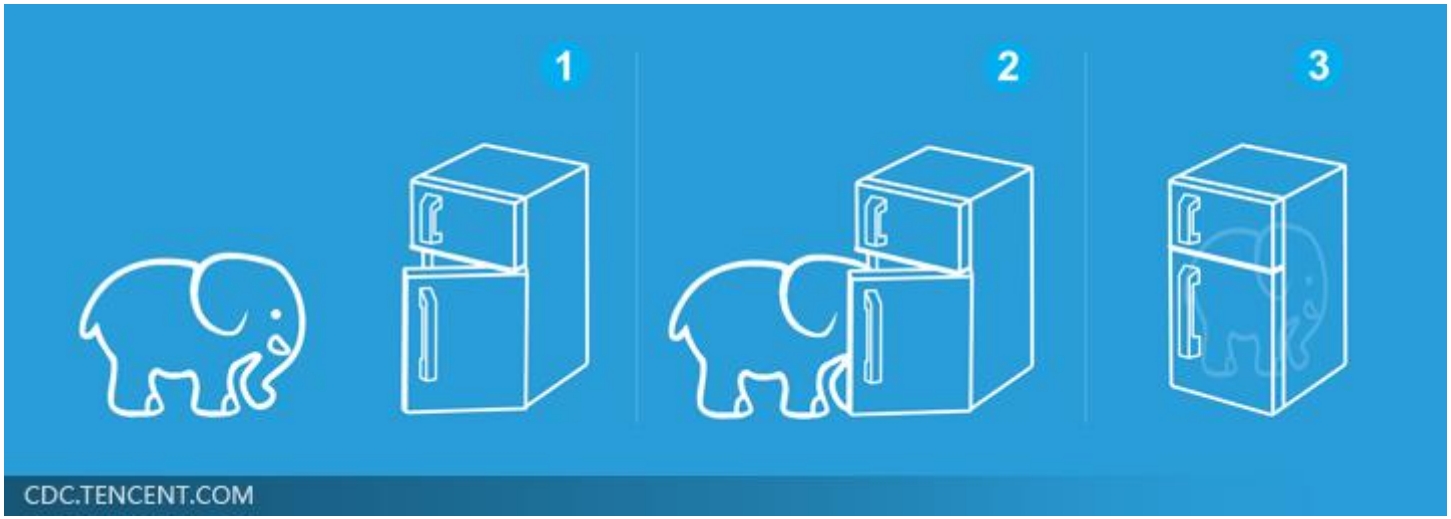
常用的深度学习框架

1. 简易和快速的原型设计
2. 自动梯度计算
3. 无缝CPU和GPU切换

深度学习的三个步骤

来源：李宏毅《1天搞懂深度学习》

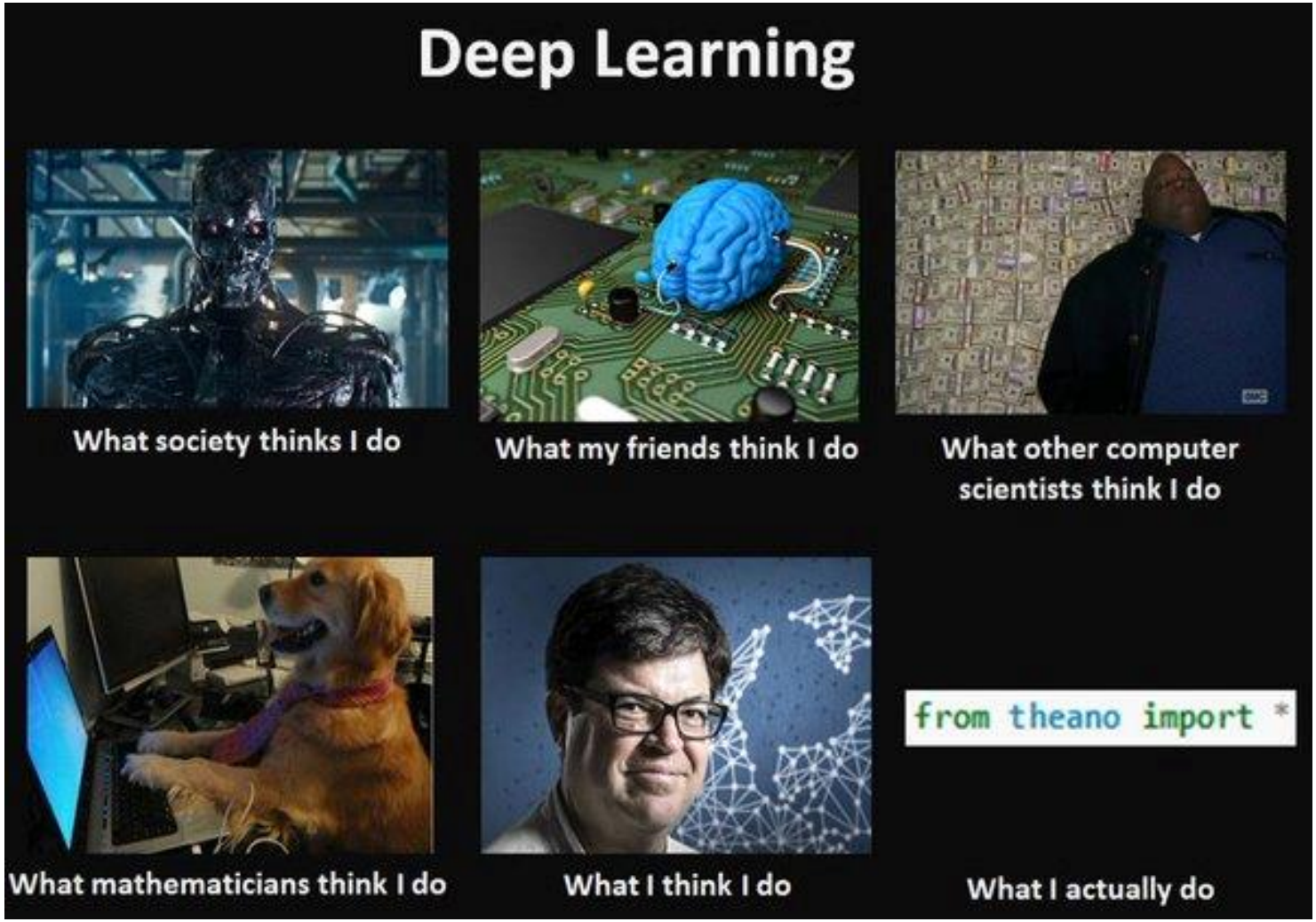
Deep Learning is so simple ……



Getting started: 30 seconds to Keras

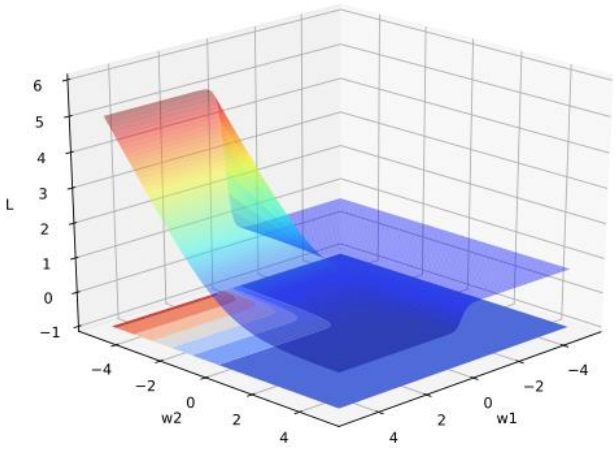
```
from keras.models import Sequential
from keras.layers import Dense, Activation
from keras.optimizers import SGD

model = Sequential()
model.add(Dense(output_dim=64, input_dim=100))
model.add(Activation("relu"))
model.add(Dense(output_dim=10))
model.add(Activation("softmax"))

model.compile(loss='categorical_crossentropy',
  optimizer='sgd', metrics=['accuracy'])

model.fit(X_train, Y_train, nb_epoch=5, batch_size=32)

loss = model.evaluate(X_test, Y_test, batch_size=32)
```

实现非常简单!

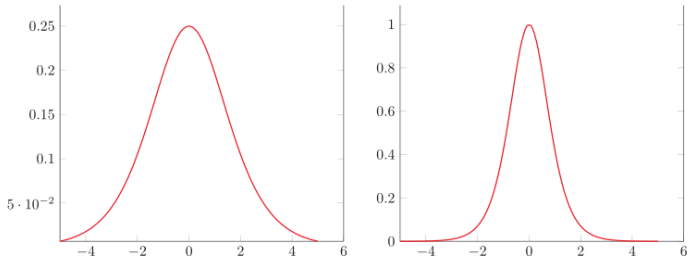

Deep Learning
What society thinks I do
What my friends think I do
What other computer scientists think I do
What mathematicians think I do
What I think I do
from theano import *
What I actually do

优化问题

- 非凸优化问题
  - $y = \sigma(w_2 \sigma(w_1 x))$的损失函数



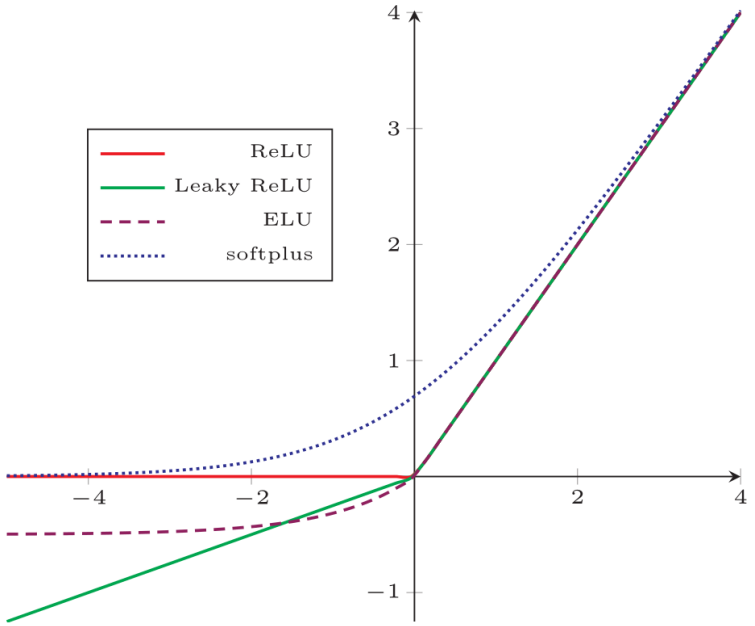
- 梯度消失问题
  - 在每一层都要乘以该层的激活函数的导数



(a) logistic 函数的导数　(b) tanh 函数的导数

激活函数

有效减轻梯度消失问题!



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

卷积神经网络

卷积神经网络

- 全连接网络
  - 权重矩阵的参数非常多
- 卷积神经网络
  - 生物学上感受野

- 卷积神经网络有三个结构上的特性：
  - 局部连接
  - 权重共享

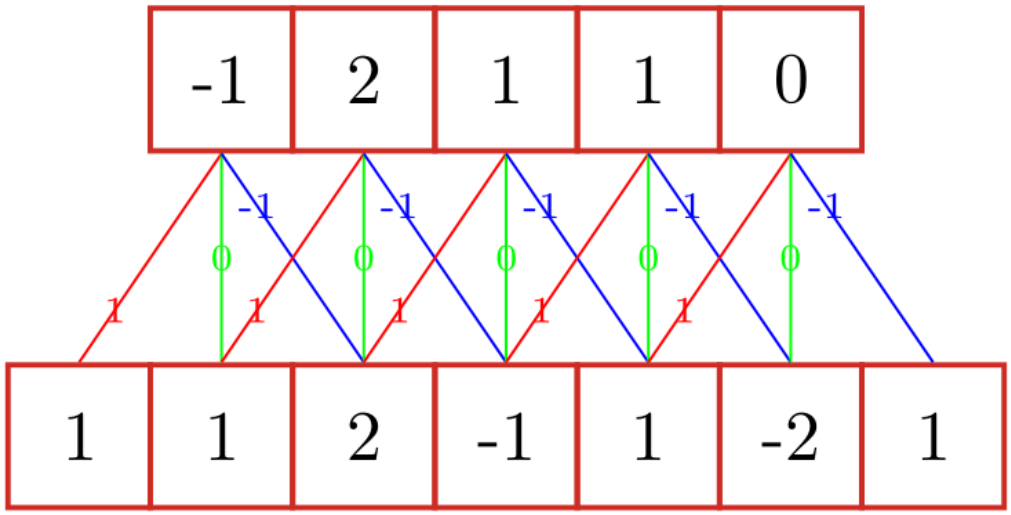
卷积

- 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

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

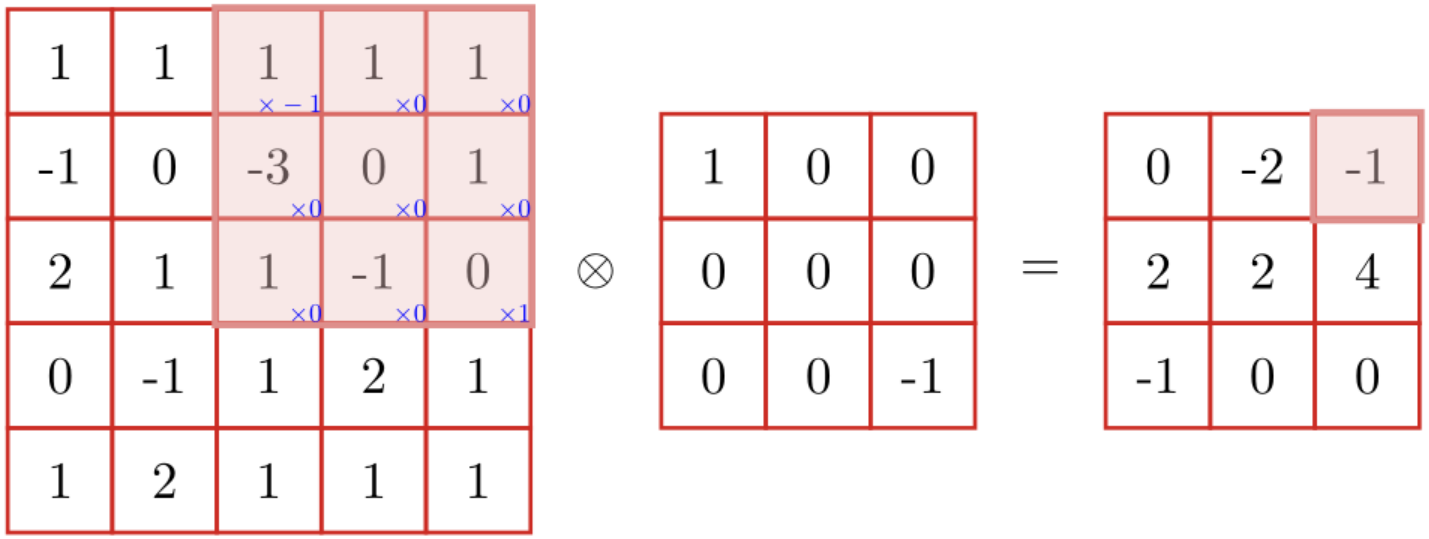
滤波器（filter）或卷积核（convolution kernel）

一维卷积

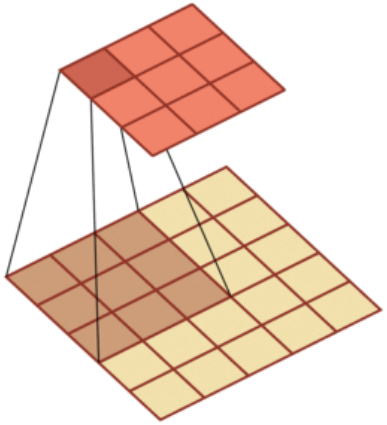


滤波器：[−1,0,1]

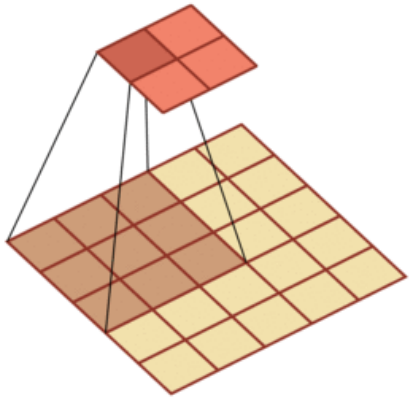
两维卷积


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
×−1 ×0 ×0
×0 ×0 ×0
×0 ×0 ×1
⊗
1 0 0
0 0 0
0 0 -1
=
0 -2 -1
2 2 4
-1 0 0

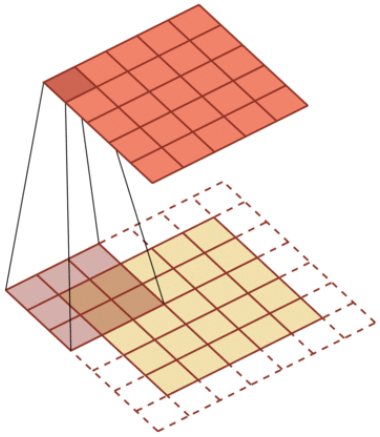
二维卷积



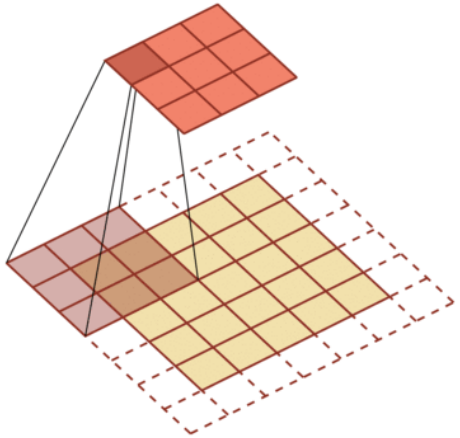
步长1，零填充0



步长2，零填充0

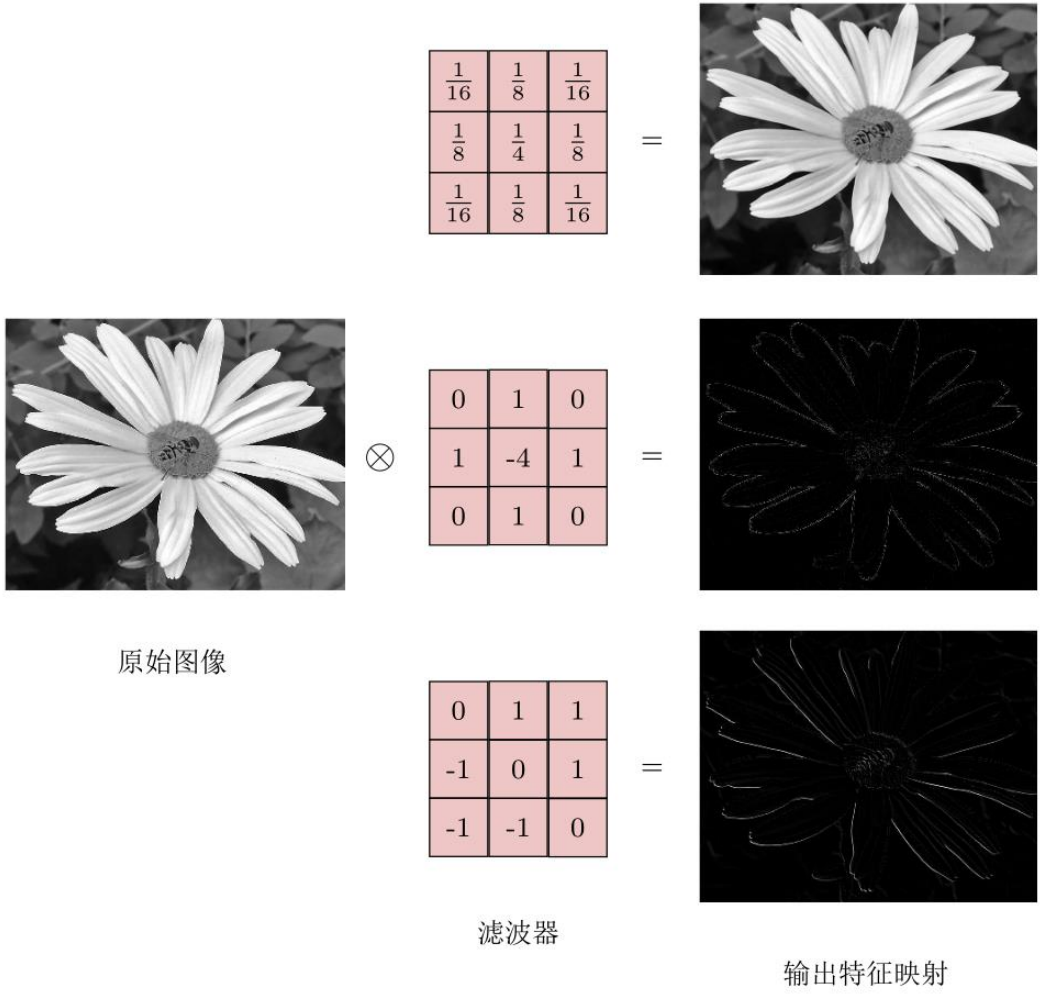


步长1，零填充1

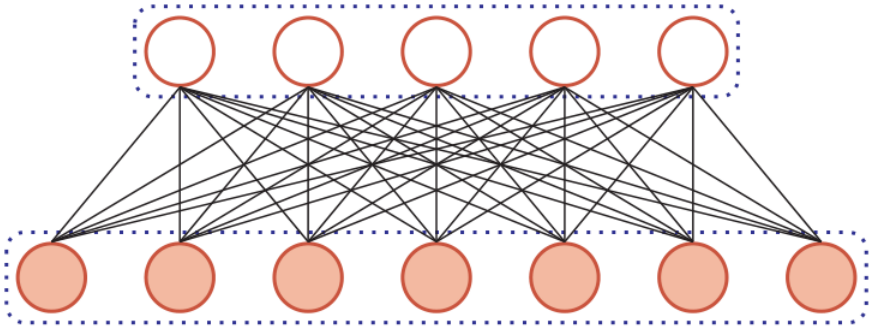


步长2，零填充1

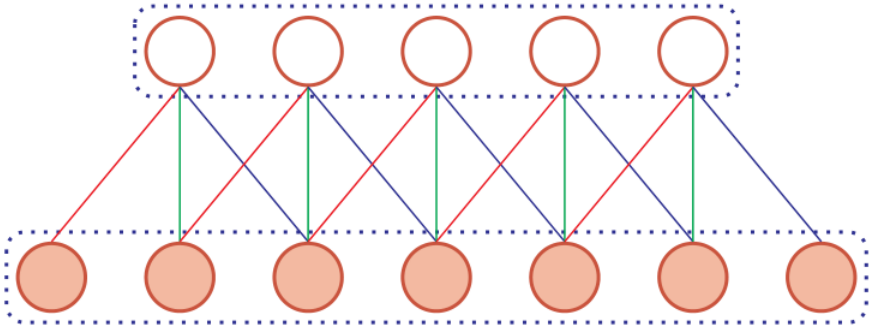
卷积作为特征提取器


$\frac{1}{16}$ $\frac{1}{8}$ $\frac{1}{16}$
$\frac{1}{8}$ $\frac{1}{4}$ $\frac{1}{8}$
$\frac{1}{16}$ $\frac{1}{8}$ $\frac{1}{16}$
=
⊗
0 1 0
1 -4 1
0 1 0
=
原始图像
0 1 1
-1 0 1
-1 -1 0
=
滤波器
输出特征映射

用卷积来代替全连接



(a) 全连接层

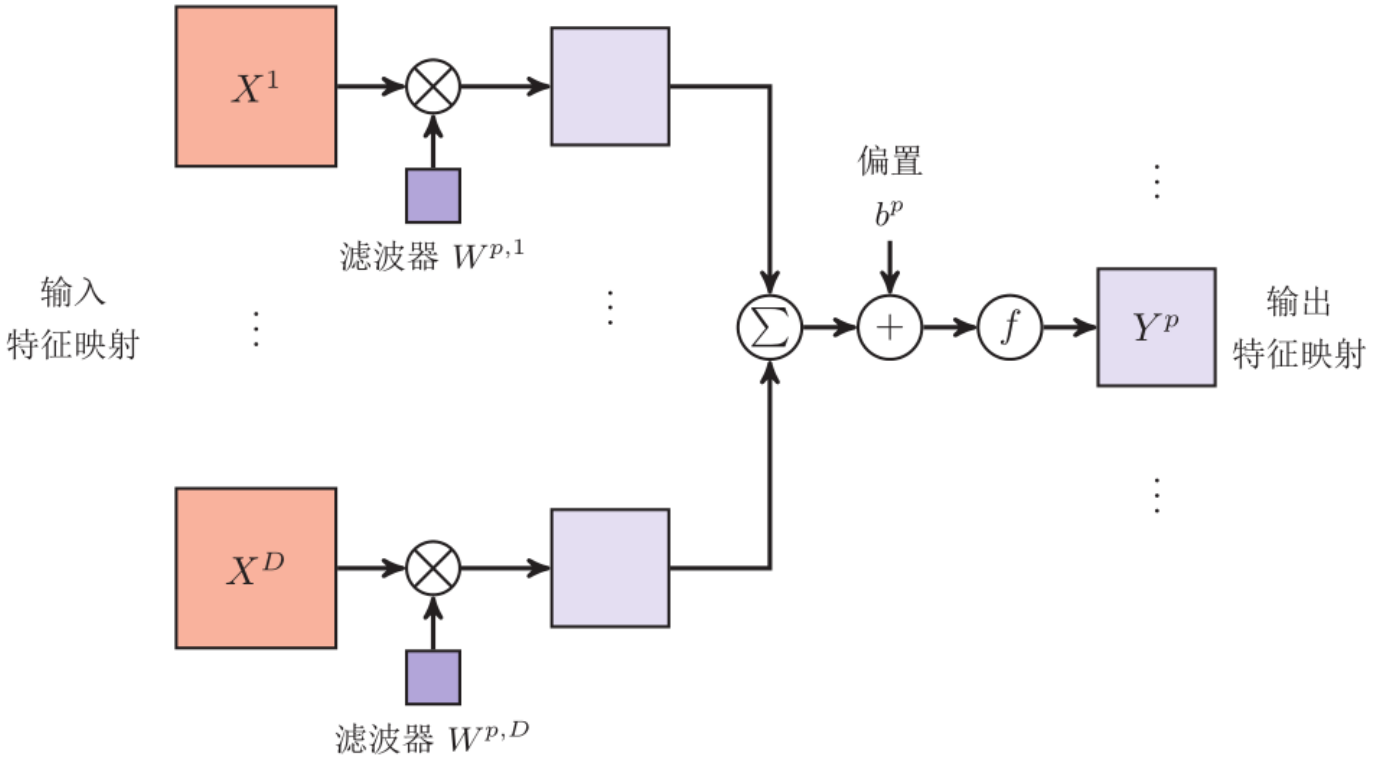


(b) 卷积层

参数数量?

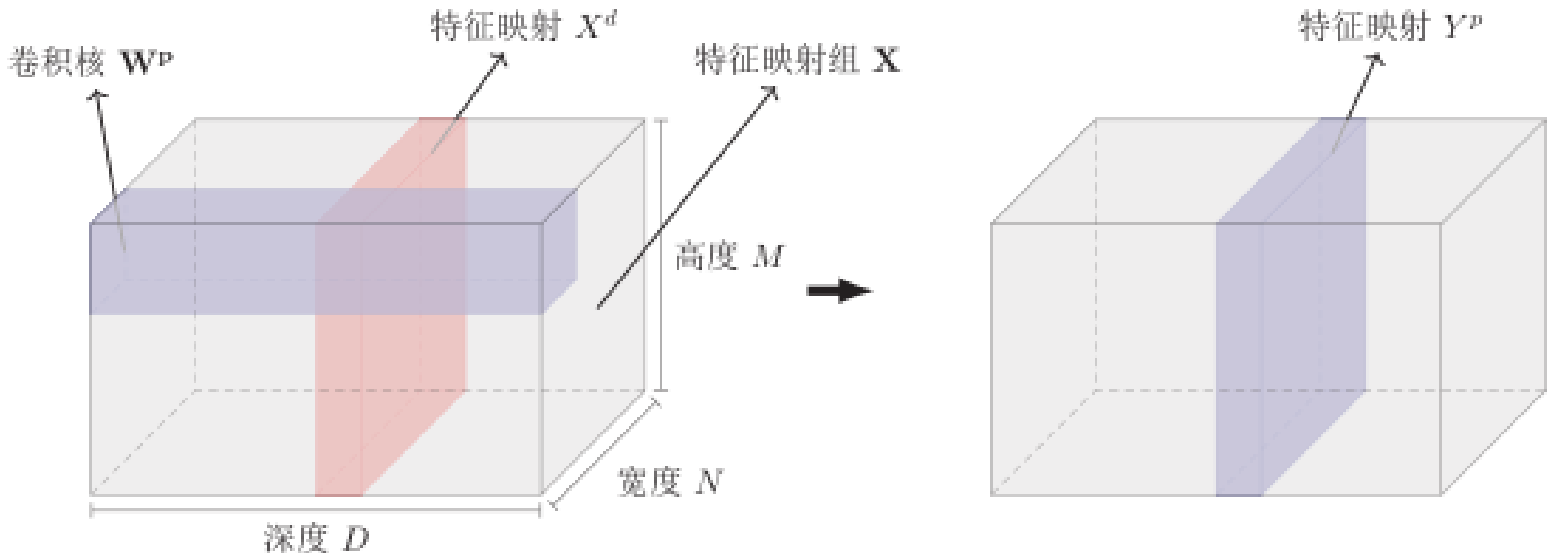
引入多组滤波器

以两维卷积为例



卷积层

- 典型的卷积层为3维结构

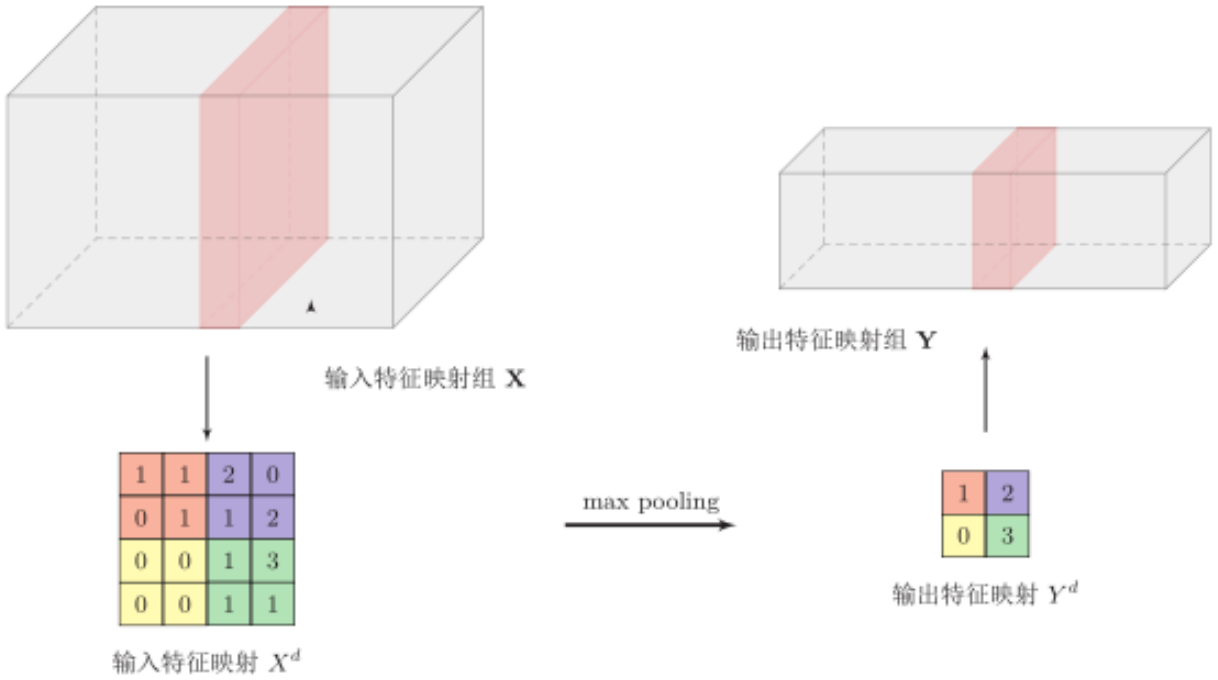


$$Z^p = \mathbf{W}^p \otimes \mathbf{X} + b^p = \sum_{d=1}^{D} W^{p,d} \otimes X^d + b^p,$$

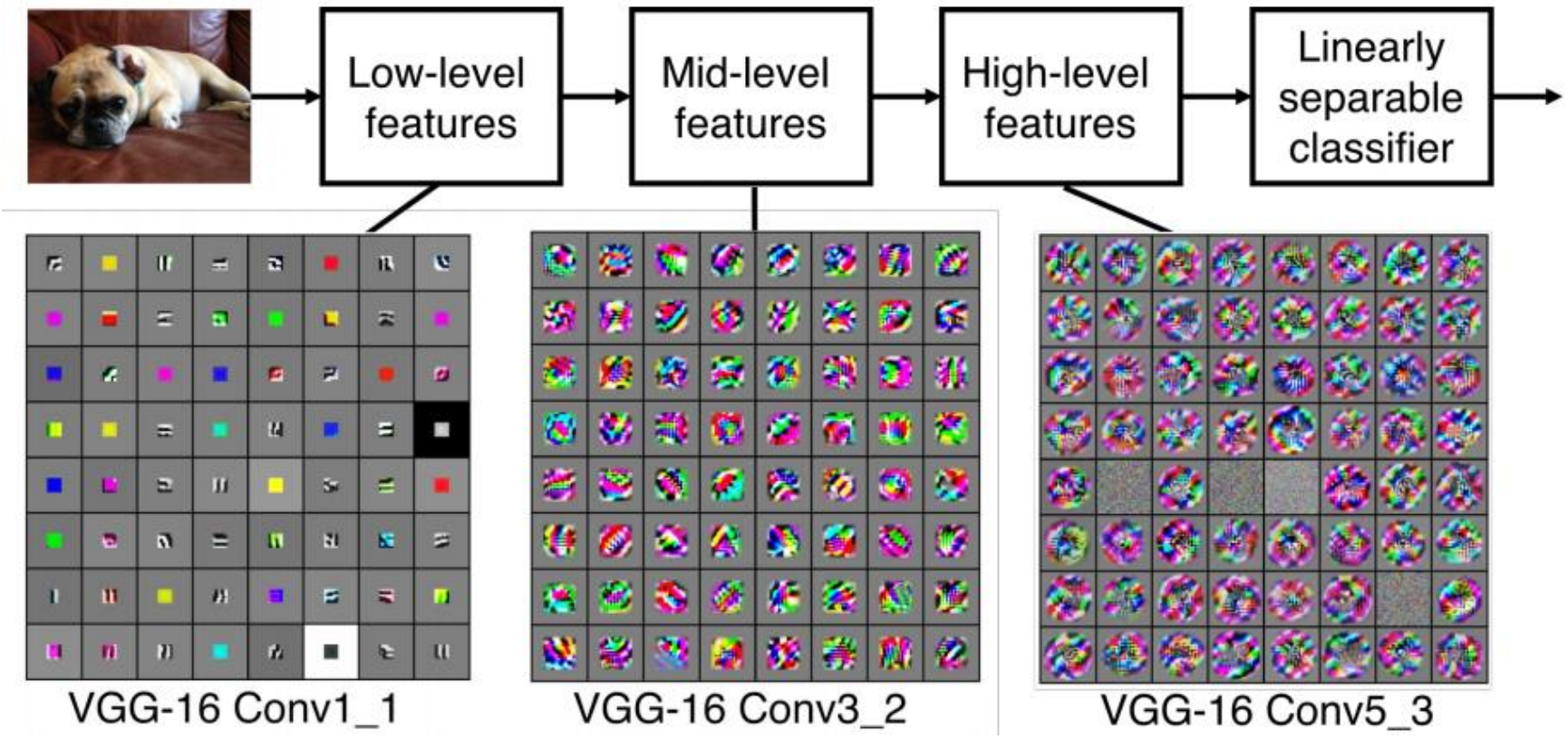
$$Y^p = f(Z^p).$$

汇聚层

- 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。

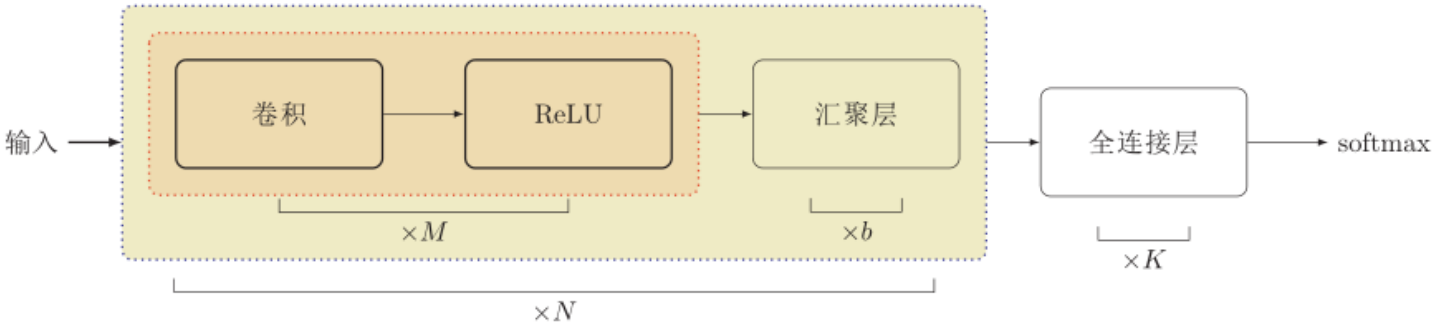


表示学习


Low-level features
Mid-level features
High-level features
Linearly separable classifier
VGG-16 Conv1_1
VGG-16 Conv3_2
VGG-16 Conv5_3

卷积网络结构

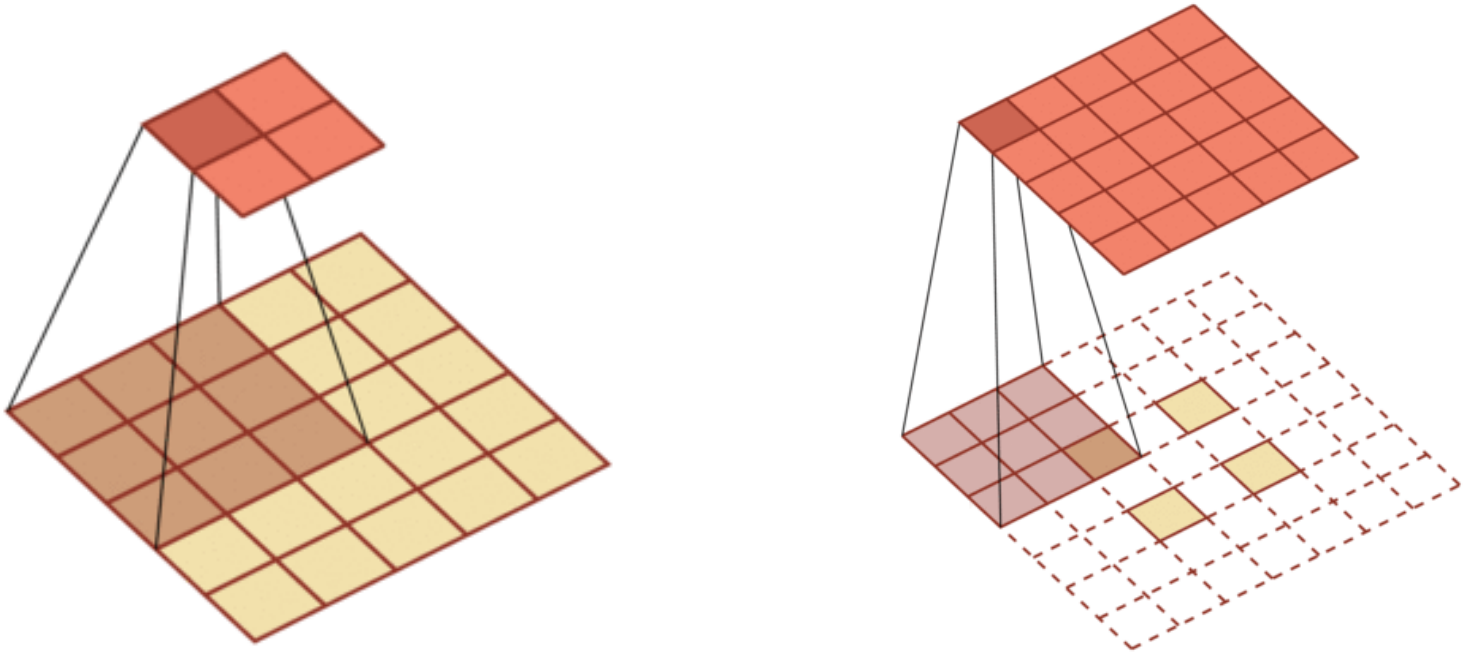
- 卷积网络是由卷积层、子采样层、全连接层交叉堆叠而成。
  - 趋向于小卷积、大深度
  - 趋向于全卷积
- 典型结构



  - 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，然后在接着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）。

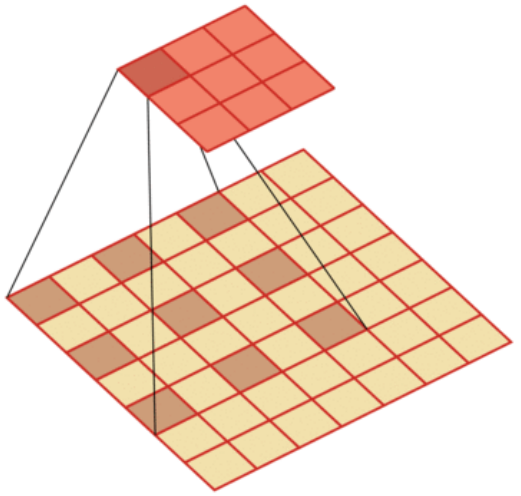
转置卷积/微步卷积

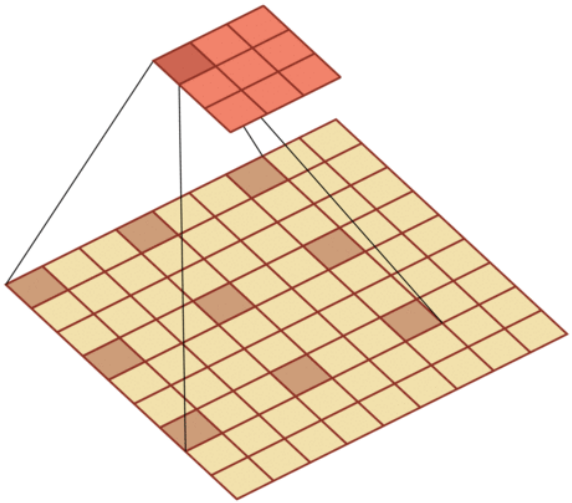
- 低维特征映射到高维特征



空洞卷积

- 如何增加输出单元的感受野
  - 增加卷积核的大小
  - 增加层数来实现
  - 在卷积之前进行汇聚操作
- 空洞卷积
  - 通过给卷积核插入“空洞”来变相地增加其大小。





典型的卷积网络

Large Scale Visual Recognition Challenge

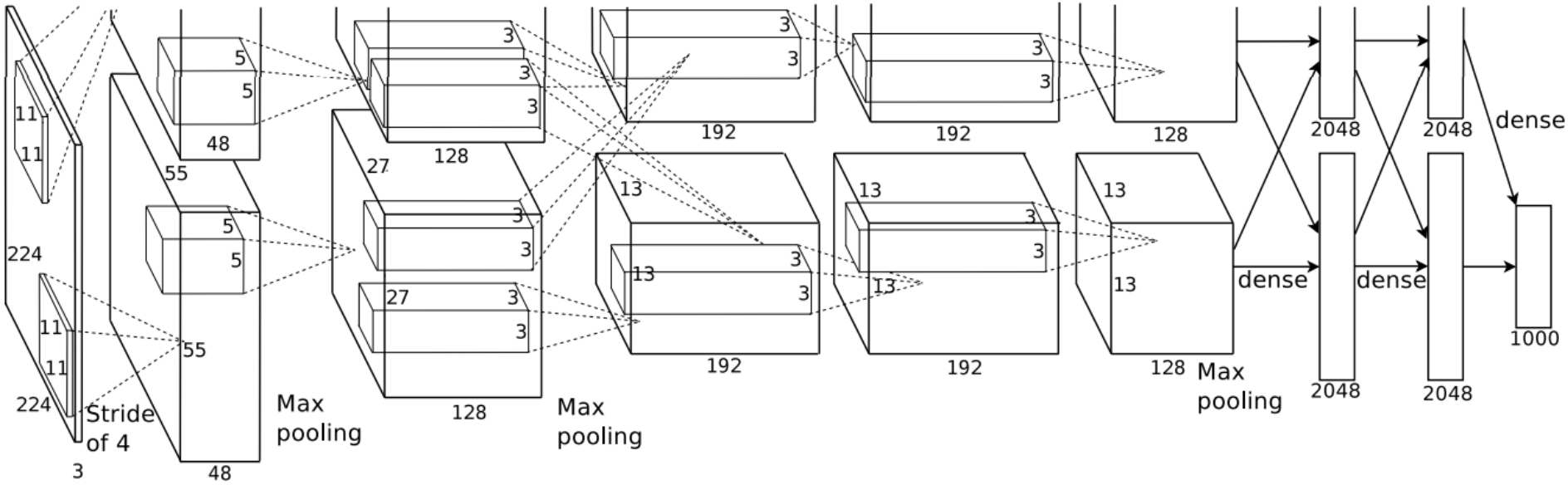
2012 Teams	%error
Supervision (Toronto)	15.3
ISI (Tokyo)	26.1
VGG (Oxford)	26.9
XRCE/INRIA	27.0
UvA (Amsterdam)	29.6
INRIA/LEAR	33.4

2013 Teams	%error
Clarifai (NYU spinoff)	11.7
NUS (singapore)	12.9
Zeiler-Fergus (NYU)	13.5
A. Howard	13.5
OverFeat (NYU)	14.1
UvA (Amsterdam)	14.2
Adobe	15.2
VGG (Oxford)	15.2
VGG (Oxford)	23.0

2014 Teams	%error
GoogLeNet	6.6
VGG (Oxford)	7.3
MSRA	8.0
A. Howard	8.1
DeeperVision	9.5
NUS-BST	9.7
TTIC-ECP	10.2
XYZ	11.2
UvA	12.1

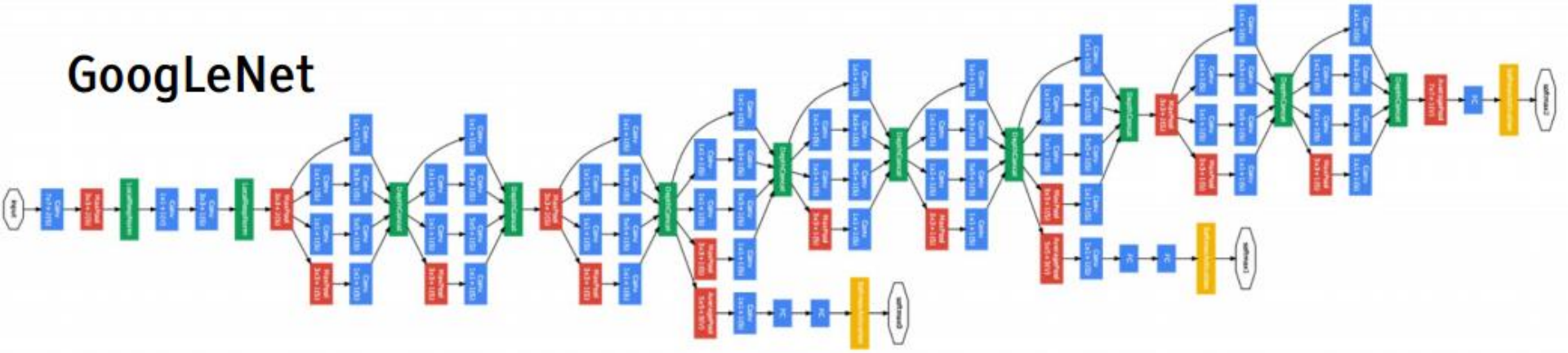
AlexNet

- 2012 ILSVRC winner
  - （top 5 error of 16\% compared to runner-up with 26\% error ）
  - 共有8层，其中前5层卷积层，后边3层全连接层

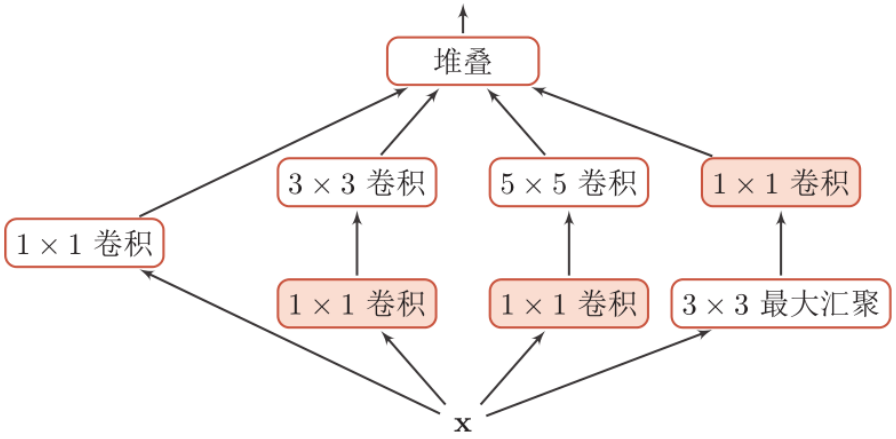


Inception网络

- 2014 ILSVRC winner （22层）
  - 参数：GoogLeNet：4M VS AlexNet：60M
  - 错误率：6.7%



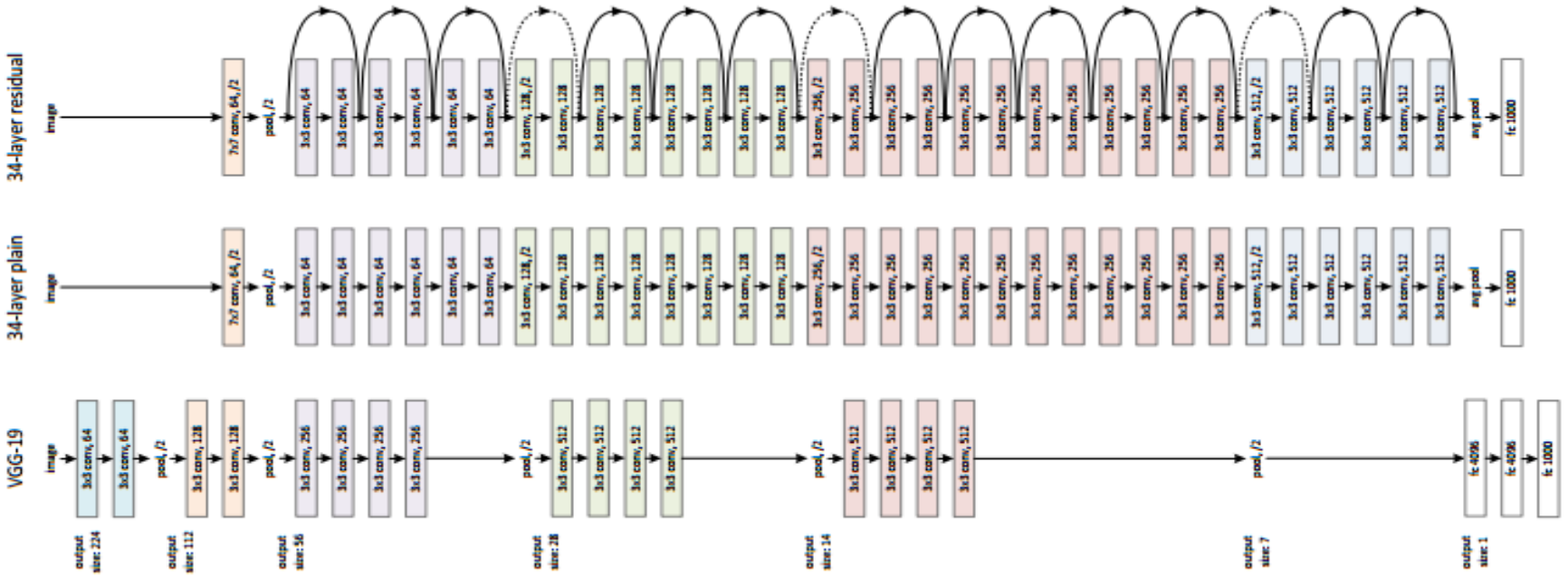
Inception v1的模块结构



卷积和最大汇聚都是等宽的。

ResNet

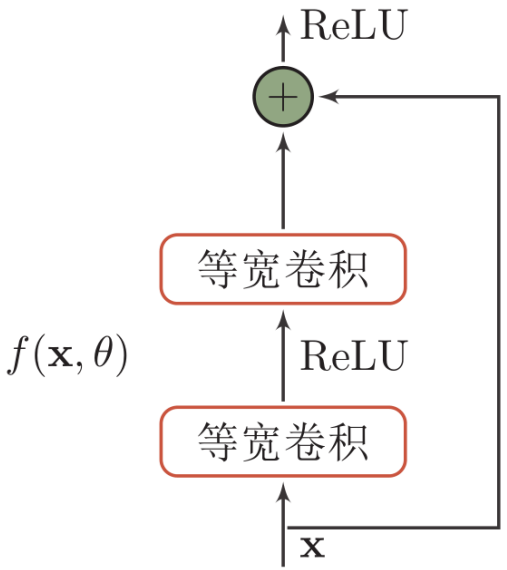
- 2015 ILSVRC winner （152层）
  - 错误率： 3.57%



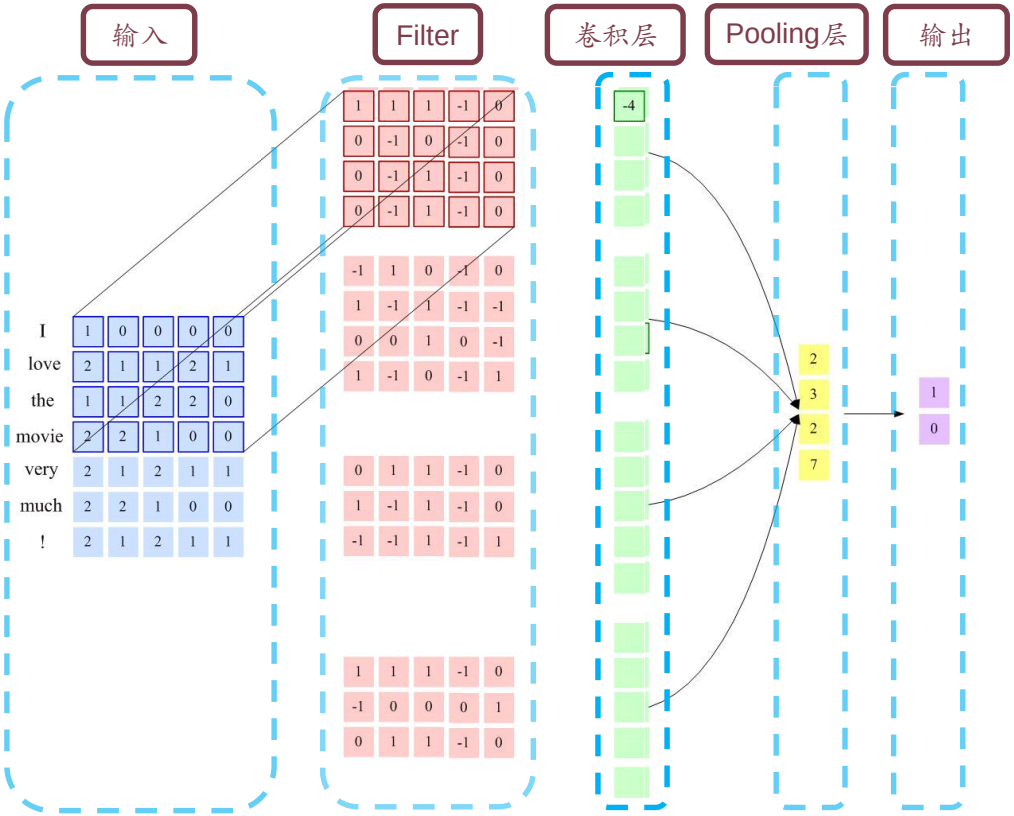
一个简单的残差单元结构

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

$$f(\mathbf{x}, \theta)$$

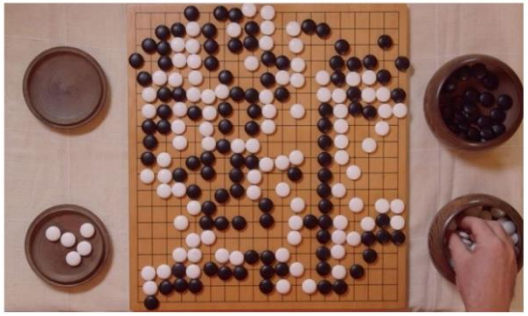


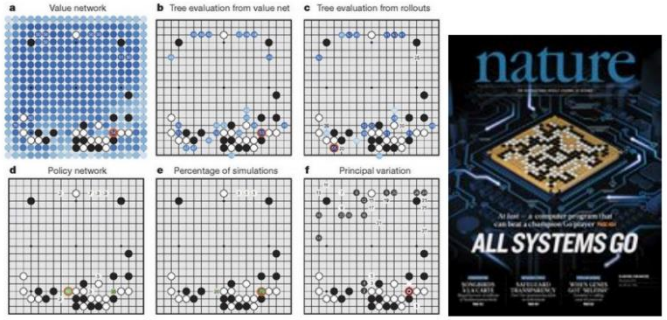
文本序列的卷积模型



卷积的应用

AlphaGo





The input to the policy network is a 19 × 19 × 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.

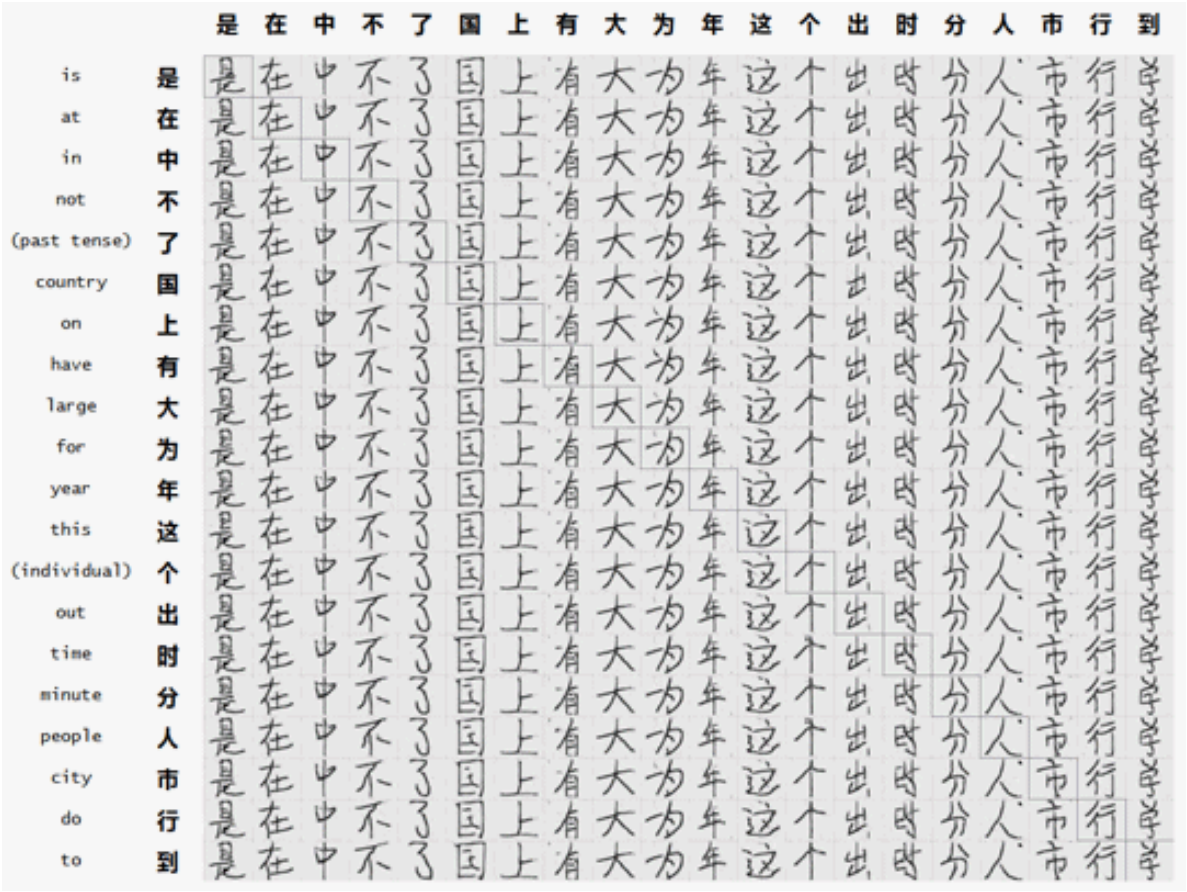
policy network:
[19x19x48] Input
CONV1: 192 5x5 filters , stride 1, pad 2 => [19x19x192]
CONV2..12: 192 3x3 filters, stride 1, pad 1 => [19x19x192]
CONV: 1 1x1 filter, stride 1, pad 0 => [19x19] *(probability map of promising moves)*

- 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

Mask RCNN

Figure 4. More results of **Mask R-CNN** on COCO test images, using ResNet-101-FPN and running at 5 fps, with 35.7 mask AP (Table 1).

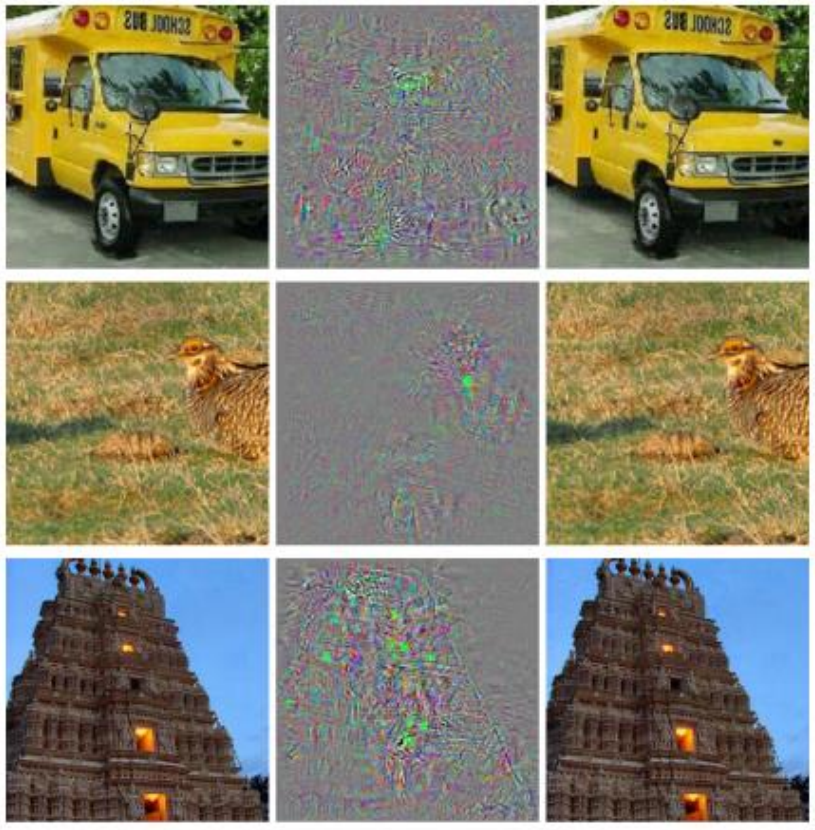
图像生成


是 在 中 不 了 国 上 有 大 为 年 这 个 出 时 分 人 市 行 到
is 是
at 在
in 中
not 不
(past tense) 了
country 国
on 上
have 有
large 大
for 为
year 年
this 这
(individual) 个
out 出
time 时
minute 分
people 人
city 市
do 行
to 到

Deep Dream

画风迁移

对抗样本



(a)

(b)

循环神经网络

前馈网络的一些不足

- 连接存在层与层之间，每层的节点之间是无连接的。（无循环）
- 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。
- 假设每次输入都是独立的，也就是说每次网络的输出只依赖于当前的输入。

- 如何用FNN去模拟一个有限状态自动机?

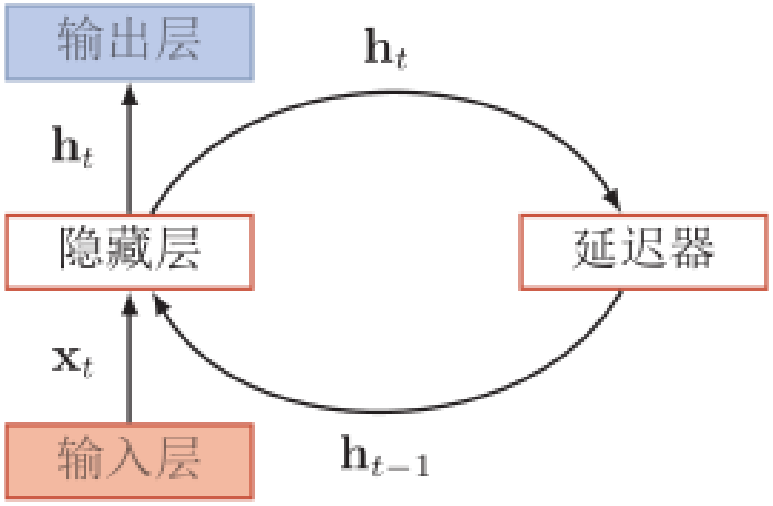
循环神经网络

- 循环神经网络
  - 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序列。
  - 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。
  - 循环神经网络已经被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。

循环神经网络

给定一个输入序列$\mathbf{x}_{1:T} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t, \ldots, \mathbf{x}_T)$，循环神经网络通过下面公式更新带反馈边的隐藏层的**活性值**$\mathbf{h}_t$：

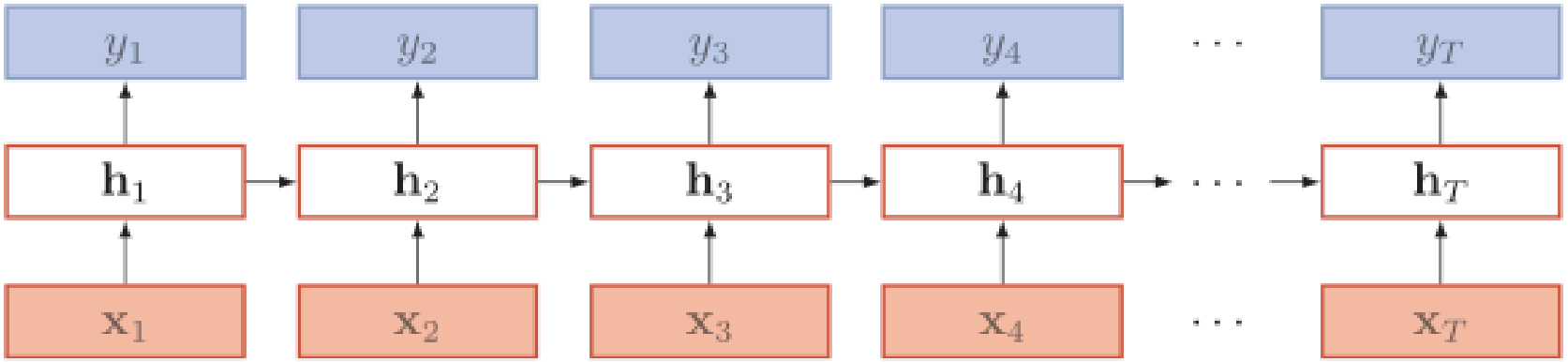
$$\mathbf{h}_t = \begin{cases} 0 & t = 0 \\ f(\mathbf{h}_{t-1}, \mathbf{x}_t) & \text{otherwise} \end{cases}$$



简单循环网络

- 状态更新：

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



RNN是图灵完全等价的 (Siegelmann and Sontag, 1995)

FNN：模拟任何函数
RNN：模拟任何程序（计算过程）。

长期依赖问题

- 改进方法
  - 循环边改为线性依赖关系

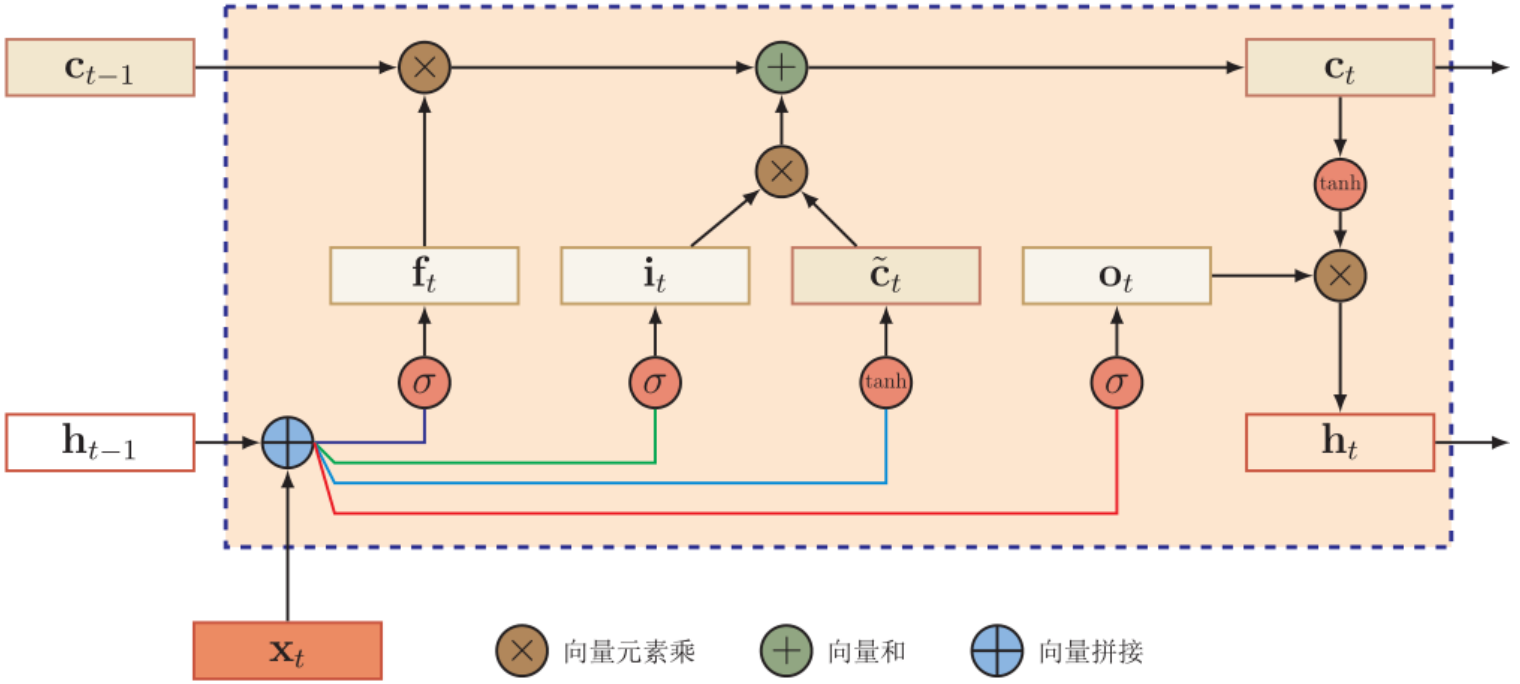
$$\mathbf{h}_t = \mathbf{h}_{t-1} + g(\mathbf{x}_t; \theta),$$

  - 增加非线性

$$\mathbf{h}_t = \mathbf{h}_{t-1} + g(\mathbf{x}_t, \mathbf{h}_{t-1}; \theta),$$

残差网络?

长短时记忆神经网络：LSTM



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

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

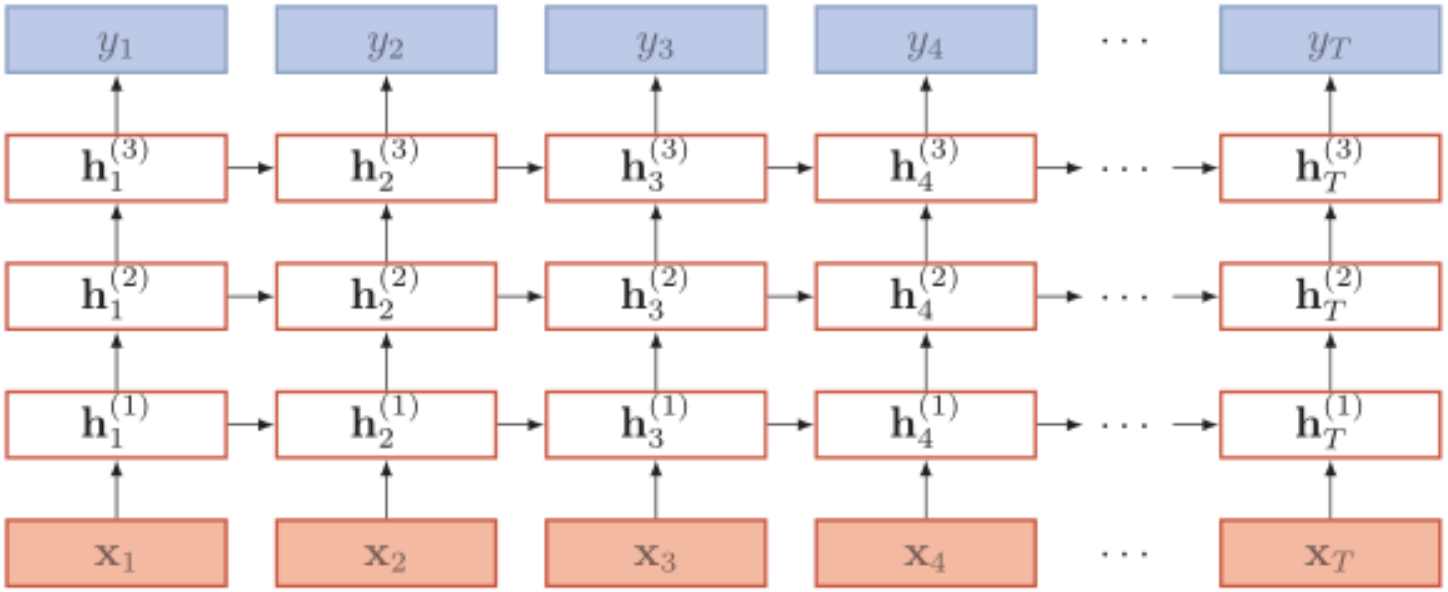
$$\mathbf{o}_t = \sigma(W_o\mathbf{x}_t + U_o\mathbf{h}_{t-1} + \mathbf{b}_o),$$

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

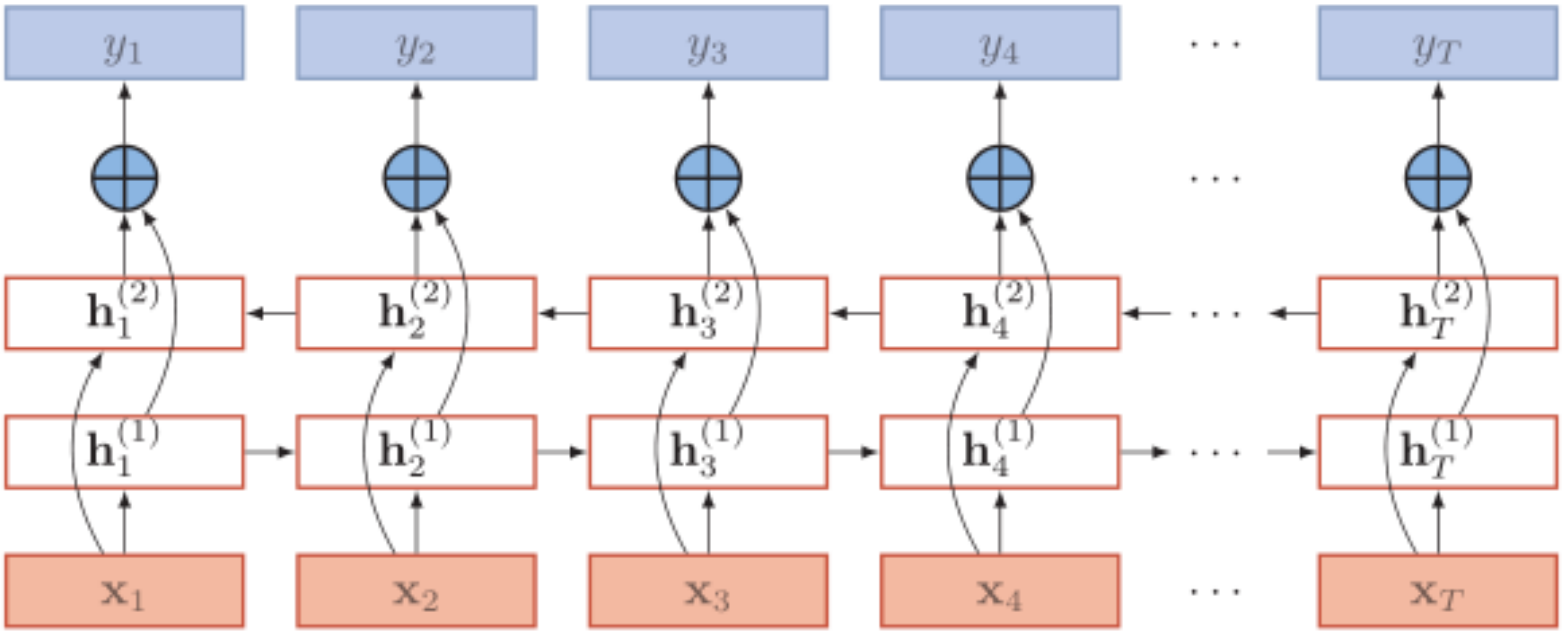
$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t,$$

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

堆叠循环神经网络


y_1 y_2 y_3 y_4 $\cdots$ y_T
$\mathbf{h}_1^{(3)}$ $\mathbf{h}_2^{(3)}$ $\mathbf{h}_3^{(3)}$ $\mathbf{h}_4^{(3)}$ $\cdots$ $\mathbf{h}_T^{(3)}$
$\mathbf{h}_1^{(2)}$ $\mathbf{h}_2^{(2)}$ $\mathbf{h}_3^{(2)}$ $\mathbf{h}_4^{(2)}$ $\cdots$ $\mathbf{h}_T^{(2)}$
$\mathbf{h}_1^{(1)}$ $\mathbf{h}_2^{(1)}$ $\mathbf{h}_3^{(1)}$ $\mathbf{h}_4^{(1)}$ $\cdots$ $\mathbf{h}_T^{(1)}$
$\mathbf{x}_1$ $\mathbf{x}_2$ $\mathbf{x}_3$ $\mathbf{x}_4$ $\cdots$ $\mathbf{x}_T$

双向循环神经网络


y_1
y_2
y_3
y_4
$\cdots$
y_T
$\mathbf{h}_1^{(2)}$
$\mathbf{h}_2^{(2)}$
$\mathbf{h}_3^{(2)}$
$\mathbf{h}_4^{(2)}$
$\mathbf{h}_T^{(2)}$
$\mathbf{h}_1^{(1)}$
$\mathbf{h}_2^{(1)}$
$\mathbf{h}_3^{(1)}$
$\mathbf{h}_4^{(1)}$
$\mathbf{h}_T^{(1)}$
$\mathbf{x}_1$
$\mathbf{x}_2$
$\mathbf{x}_3$
$\mathbf{x}_4$
$\mathbf{x}_T$

循环神经网络的扩展

- 递归神经网络

- 图网络

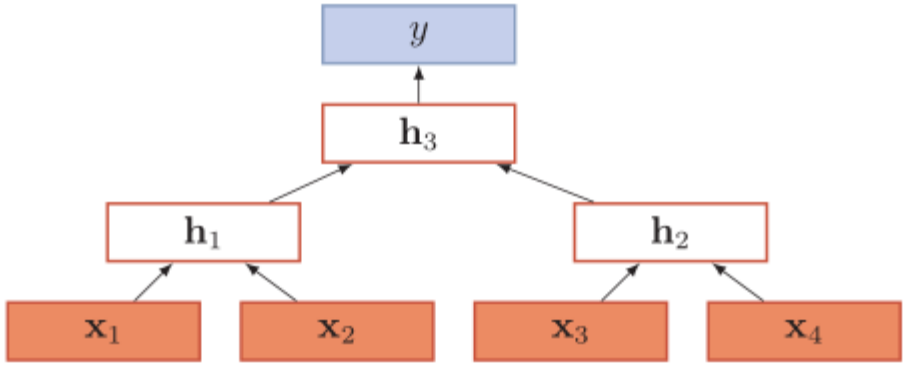
递归神经网络
Recursive Neural Network

- 递归神经网络实在一个有向图无循环图上共享一个组合函数

$$\mathbf{h}_1 = f(W \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} + \mathbf{b}),$$

$$\mathbf{h}_2 = f(W \begin{bmatrix} \mathbf{x}_3 \\ \mathbf{x}_4 \end{bmatrix} + \mathbf{b}),$$

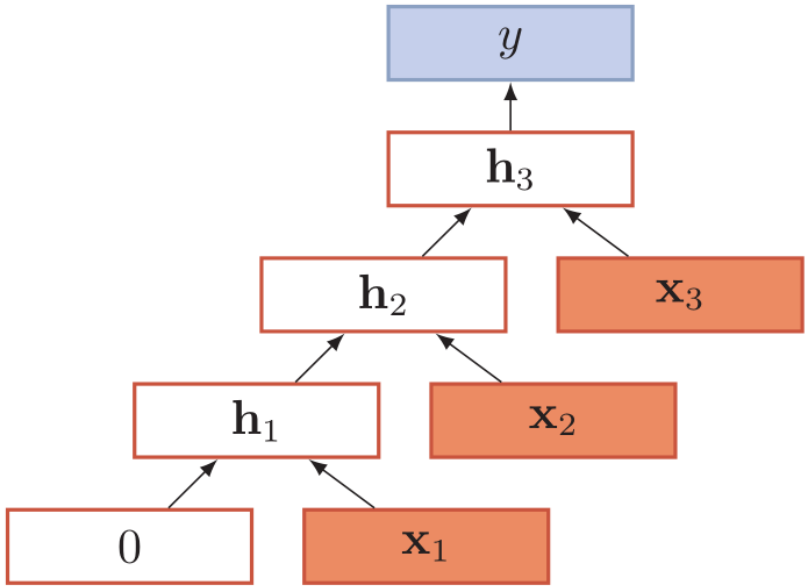
$$\mathbf{h}_3 = f(W \begin{bmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \end{bmatrix} + \mathbf{b}),$$



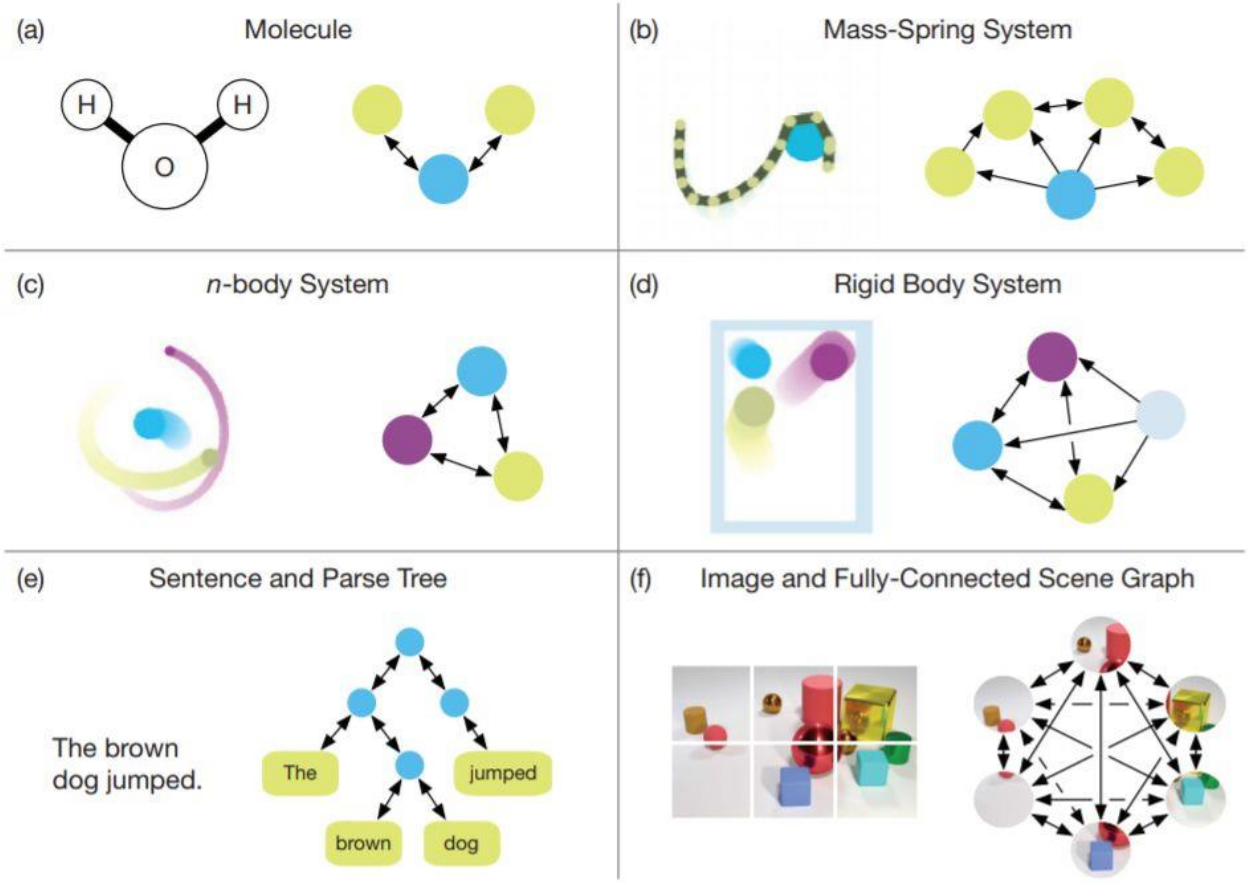
递归神经网络

- 退化为循环神经网络

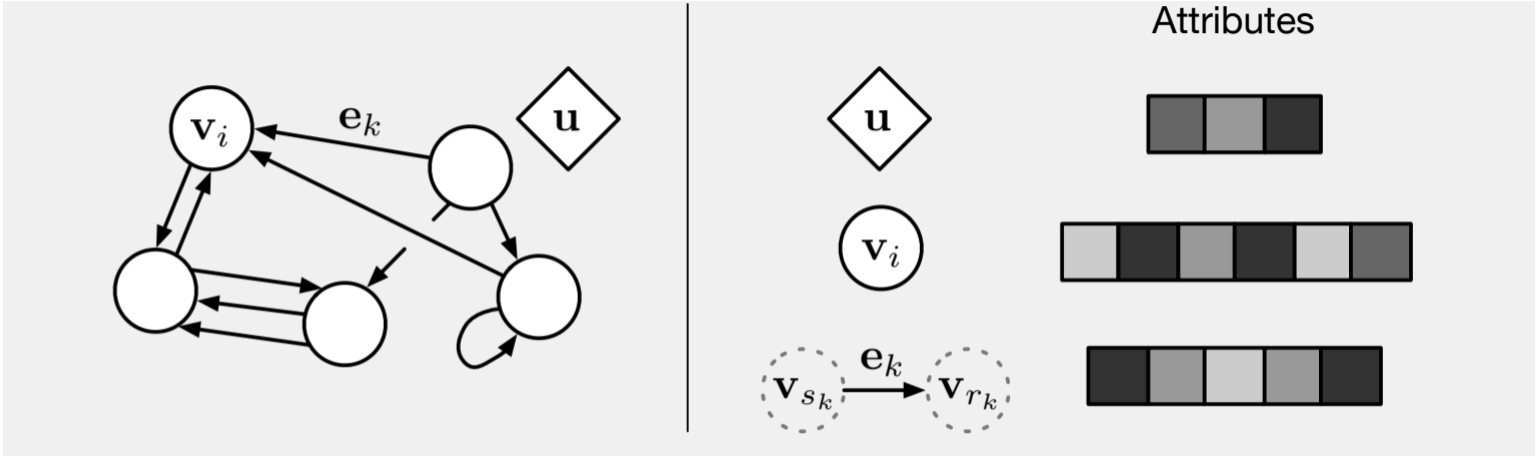
$$\mathbf{h}_t = f(\mathbf{h}_{t-1}, \mathbf{x}_t),$$

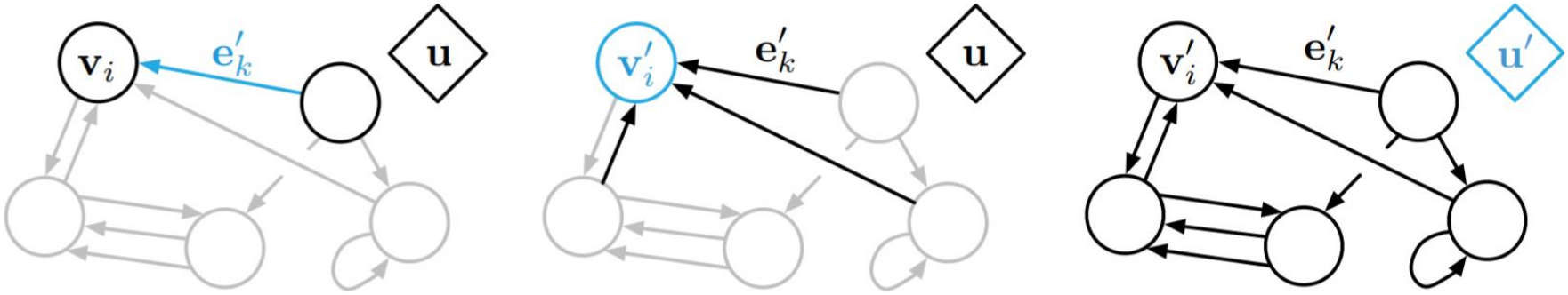


图网络


(a)
Molecule
H
H
O
(b)
Mass-Spring System
(c)
n-body System
(d)
Rigid Body System
(e)
Sentence and Parse Tree
The brown dog jumped.
The
jumped
brown
dog
(f)
Image and Fully-Connected Scene Graph

图网络





(a) Edge update (b) Node update (c) Global update

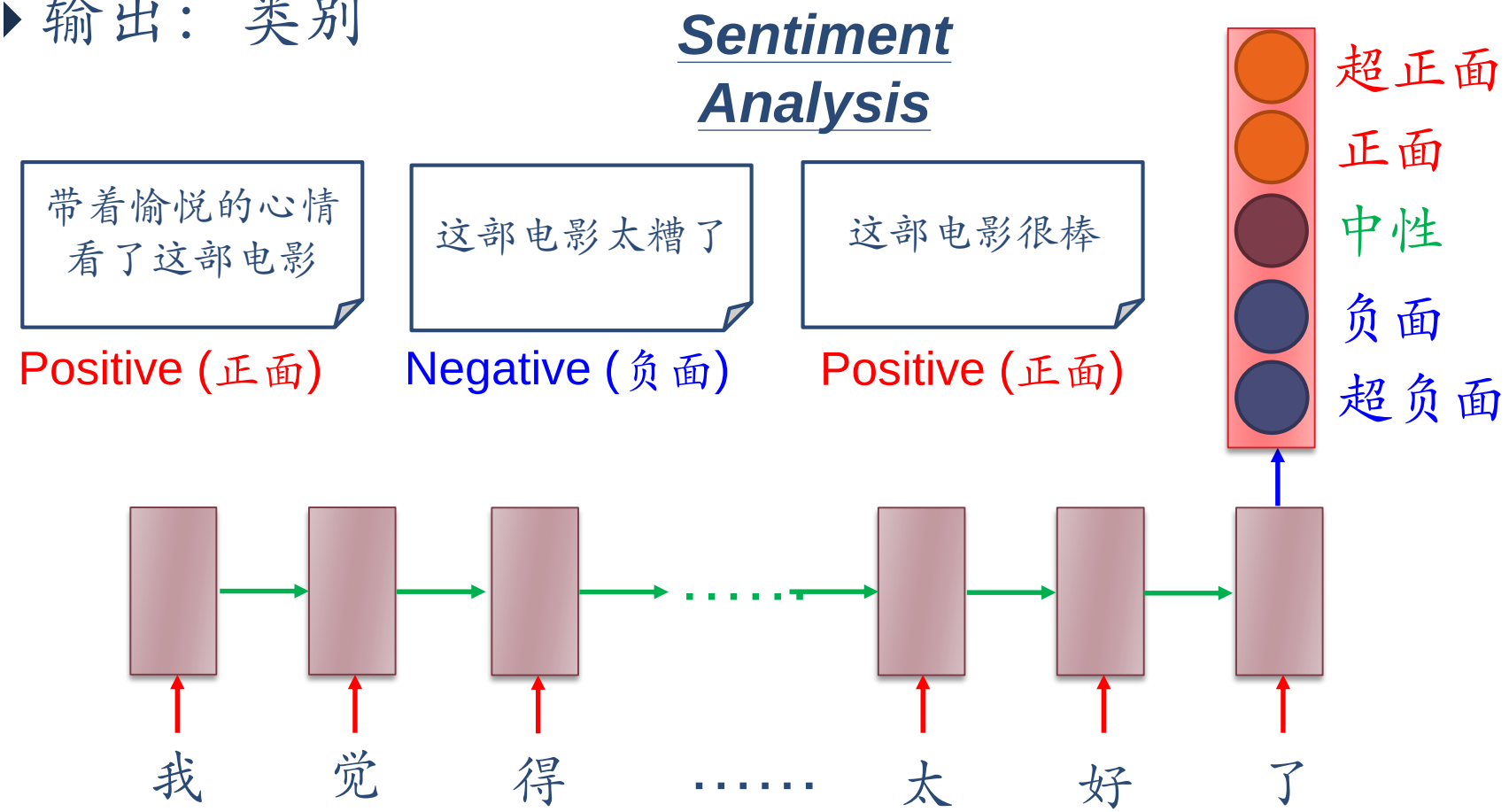
Relational inductive biases, deep learning, and graph networks

循环网络应用

序列到类别

来源：李宏毅《1天搞懂深度学习》

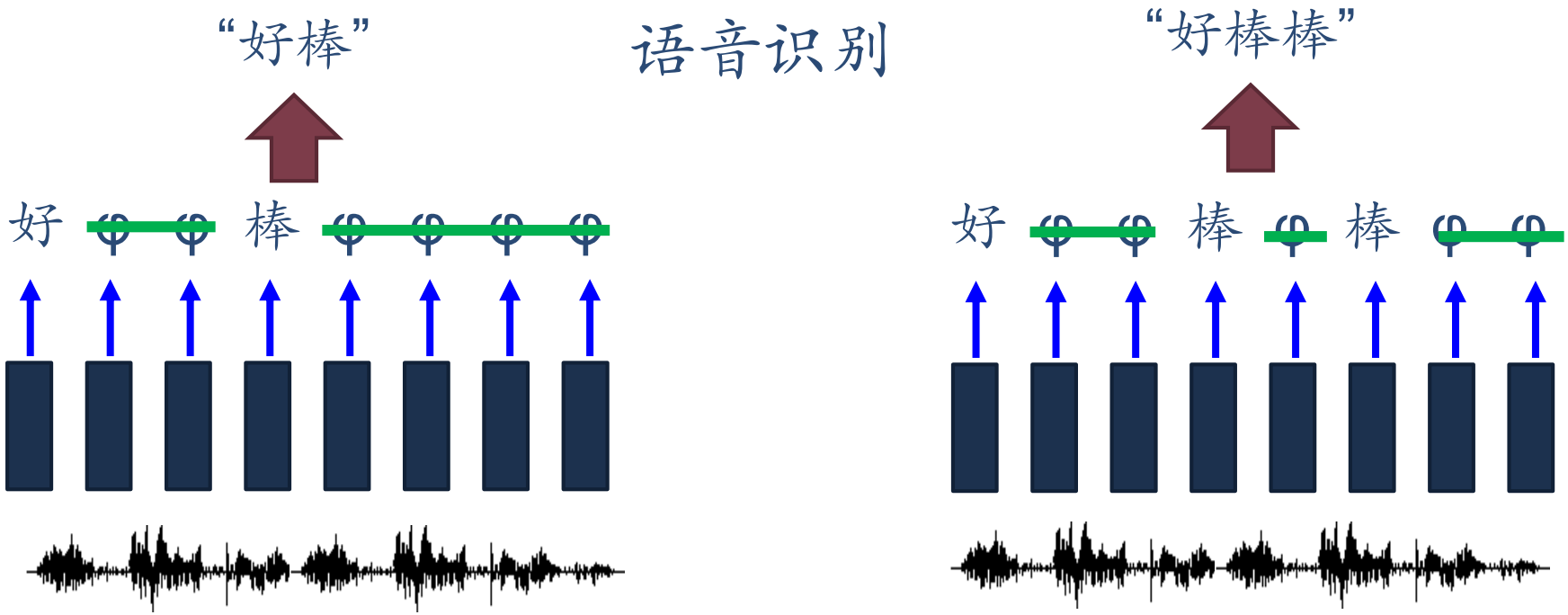
- 输入：序列
- 输出：类别



同步的序列到序列模式

来源：李宏毅《1天搞懂深度学习》

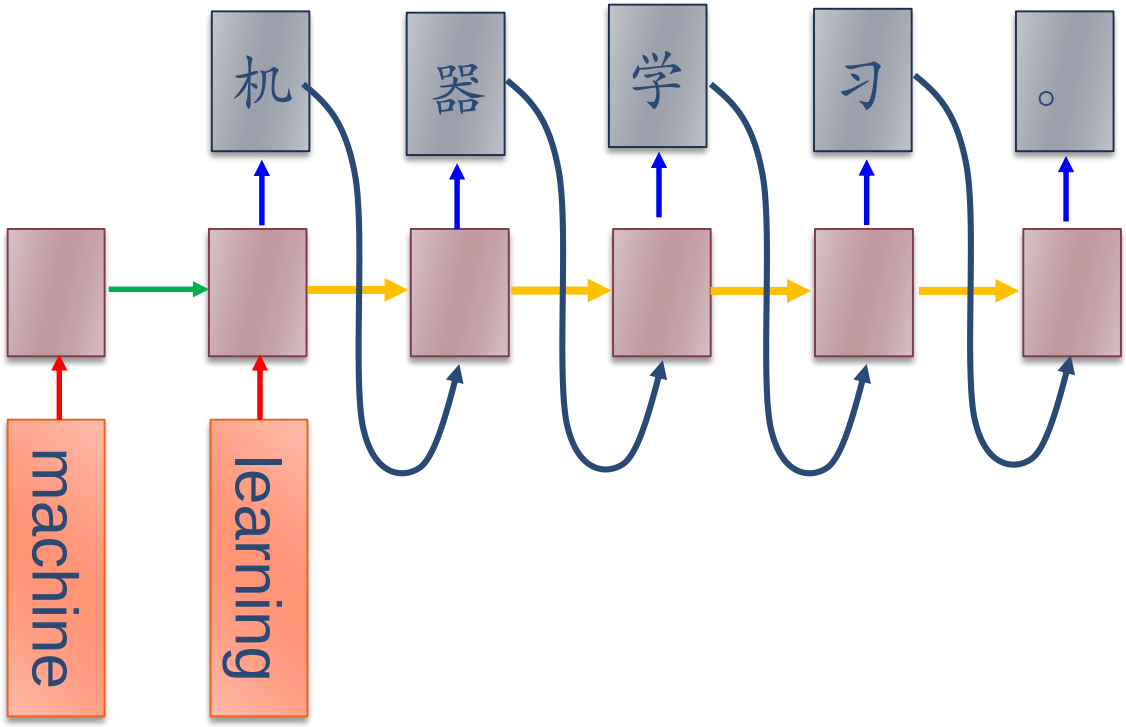
- Connectionist Temporal Classification (CTC) [Alex Graves, ICML’06][Alex Graves, ICML’14][Haşim Sak, Interspeech’15][Jie Li, Interspeech’15][Andrew Senior, ASRU’15]



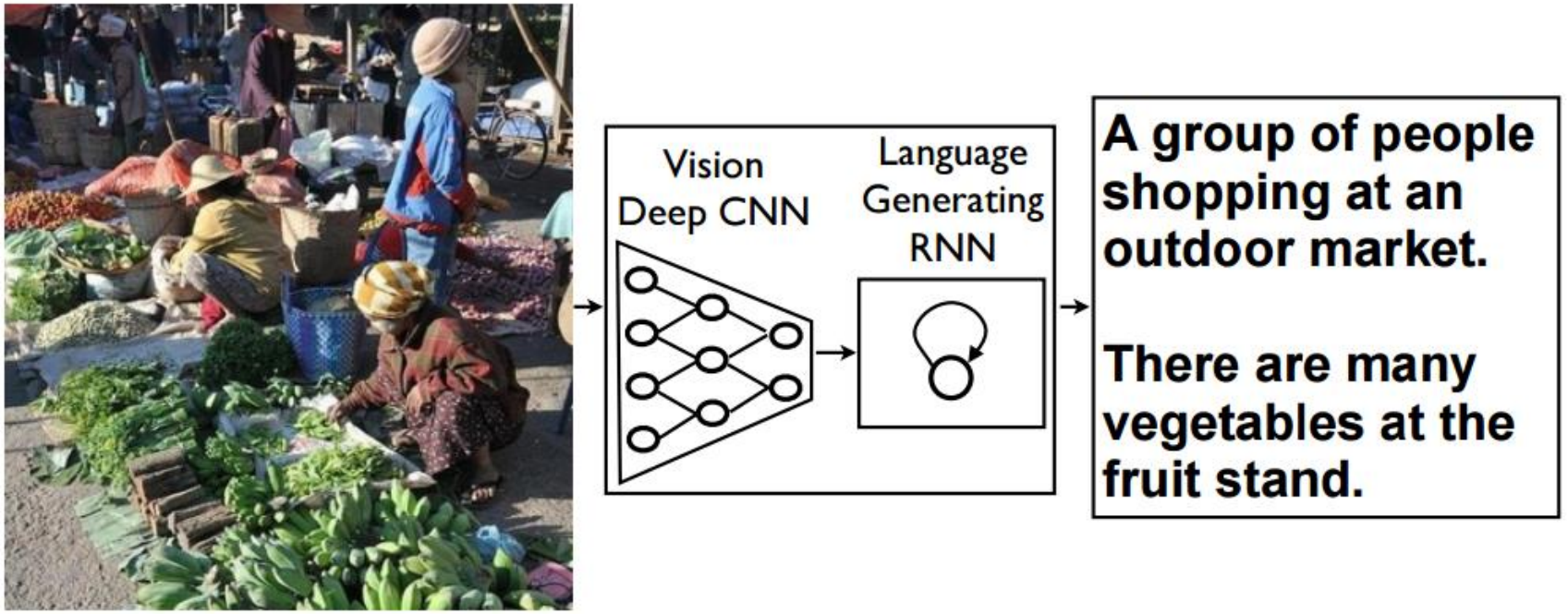
异步的序列到序列模式

来源：李宏毅《1天搞懂深度学习》

- 机器翻译



看图说话


Vision
Deep CNN
Language
Generating
RNN
A group of people shopping at an outdoor market.
There are many vegetables at the fruit stand.

看图说话

Figure 5. A selection of evaluation results, grouped by human rating.

生成LINUX内核代码

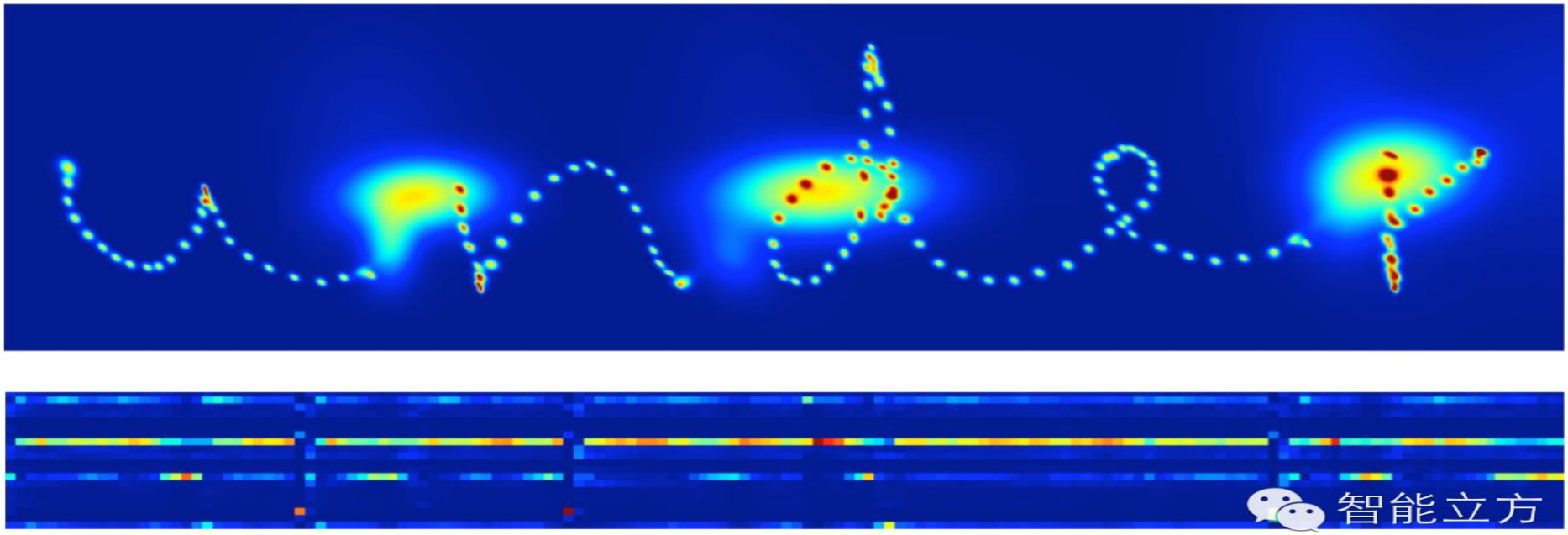
```
/*
 * If this error is set, we will need anything right after that BSD.
 */
static void action_new_function(struct s_stat_info *wb)
{
  unsigned long flags;
  int lel_idx_bit = e->edd, *sys & ~((unsigned long) *FIRST_COMPAT);
  buf[0] = 0xFFFFFFFF & (bit << 4);
  min(inc, slist->bytes);
  printk(KERN_WARNING "Memory allocated %02x/%02x, "
    "original MLL instead\n"),
    min(min(multi_run - s->len, max) * num_data_in),
    frame_pos, sz + first_seg);
  div_u64_w(val, inb_p);
  spin_unlock(&disk->queue_lock);
  mutex_unlock(&s->sock->mutex);
  mutex_unlock(&func->mutex);
  return disassemble(info->pending_bh);
}

static void num_serial_settings(struct tty_struct *tty)
{
  if (tty == tty)
    disable_single_st_p(dev);
  pci_disable_spool(port);
```

作诗

白鹭窥鱼立, Egrets stood, peeping fishes. 青山照水开. Water was still, reflecting mountains. 夜来风不动, The wind went down by nightfall, 明月见楼台. as the moon came up by the tower.	满怀风月一枝春, Budding branches are full of romance. 未见梅花亦可人. Plum blossoms are invisible but adorable. 不为东风无此客, With the east wind comes Spring. 世间何处是前身. Where on earth do I come from?

写字

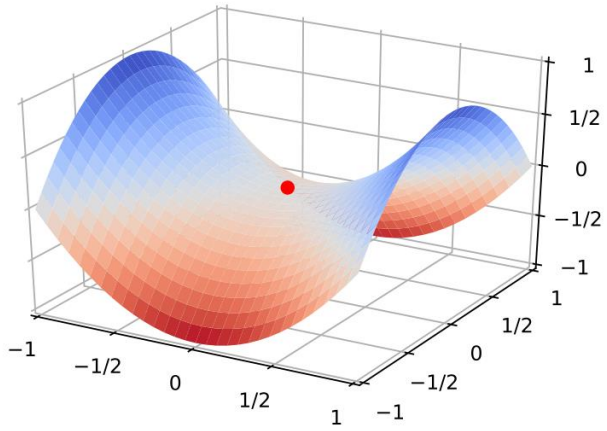
- 把一个字母的书写轨迹看作是一连串的点。一个字母的“写法”其实是每一个点相对于前一个点的偏移量，记为(offset x, offset y)。再增加一维取值为0或1来记录是否应该“提笔”。

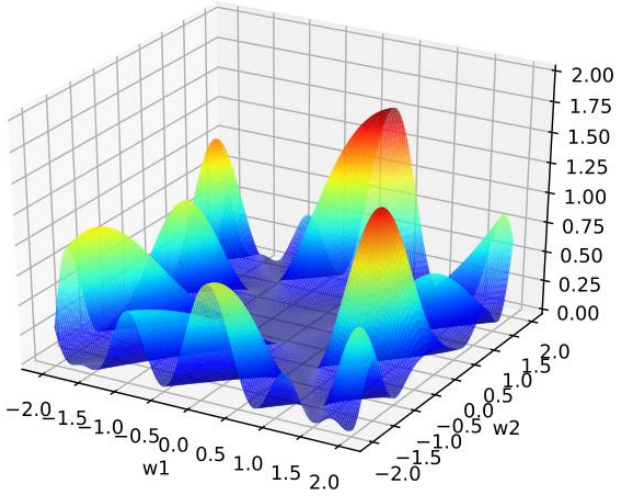


优化与正则化

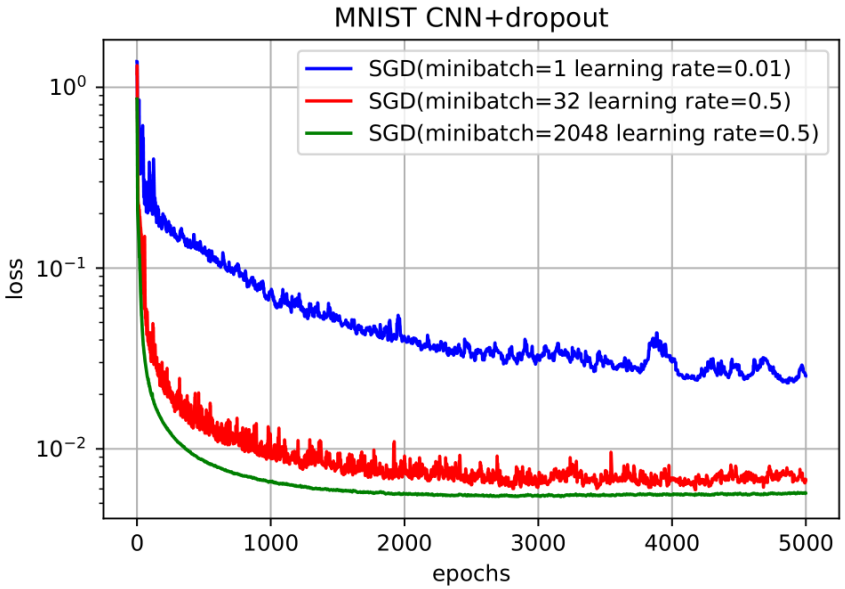
神经网络的参数学习

- 非凸优化问题
  - 参数初始化
  - 逃离局部最优
- 高维的非凸优化
  - 鞍点
  - 平摊底部

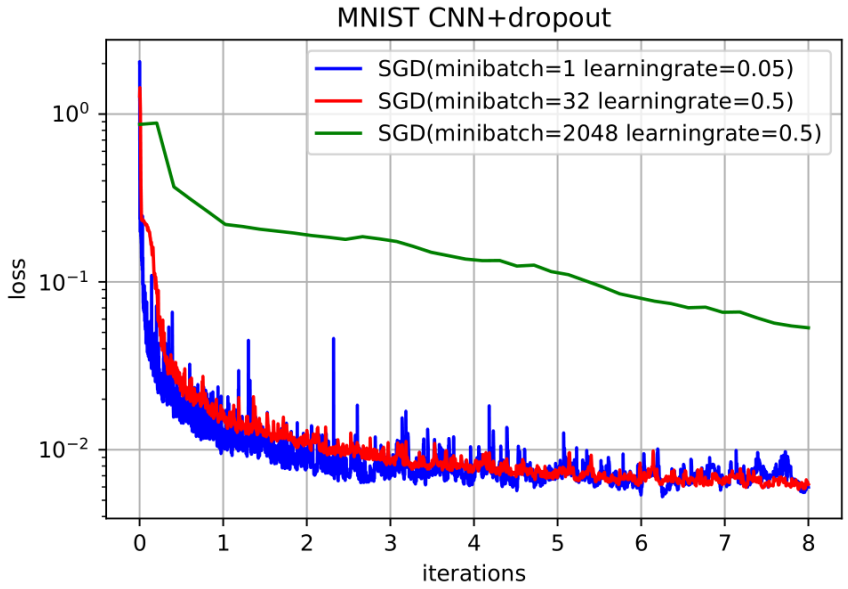




随机梯度下降



(a) 按每次小批量更新的损失变化



(b) 按整个数据集迭代的损失变化

小批量梯度下降中，每次选取样本数量对损失下降的影响。

$$\Delta\theta_t = -\alpha \mathbf{g}_t$$

如何改进?

Reference:
1. An overview of gradient descent optimization algorithms
2. Optimizing the Gradient Descent

- 标准的（小批量）梯度下降

$$\Delta\theta_t = -\alpha \mathbf{g}_t$$

实际更新方向　　梯度方向

- 学习率
  - 学习率衰减
  - Adagrad
  - Adadelta
  - RMSprop

$$\Delta\theta_t = -\frac{\alpha}{\sqrt{G_t + \epsilon}} \odot \mathbf{g}_t$$

Adam

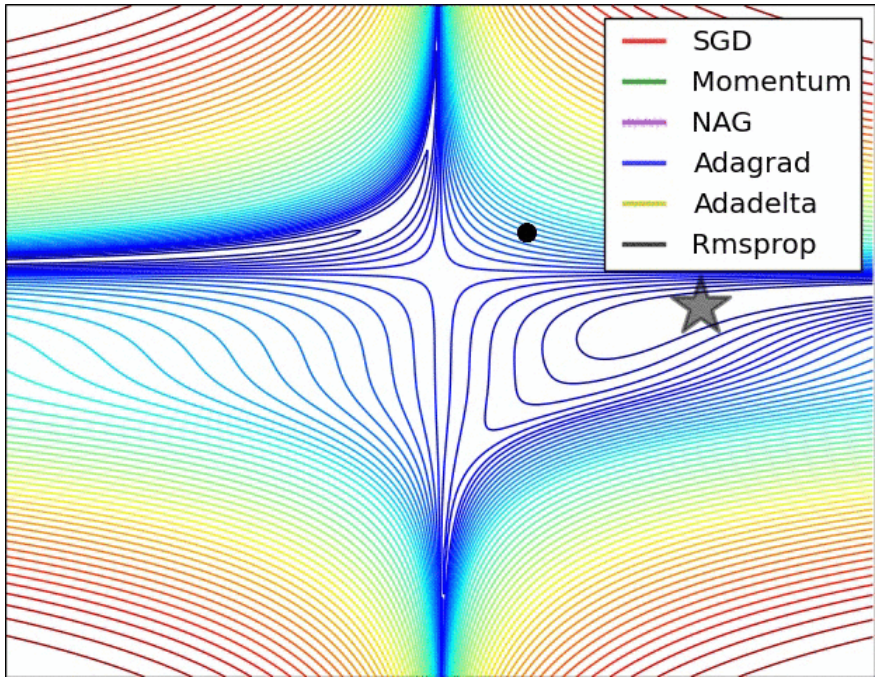
- 梯度
  - Momentum

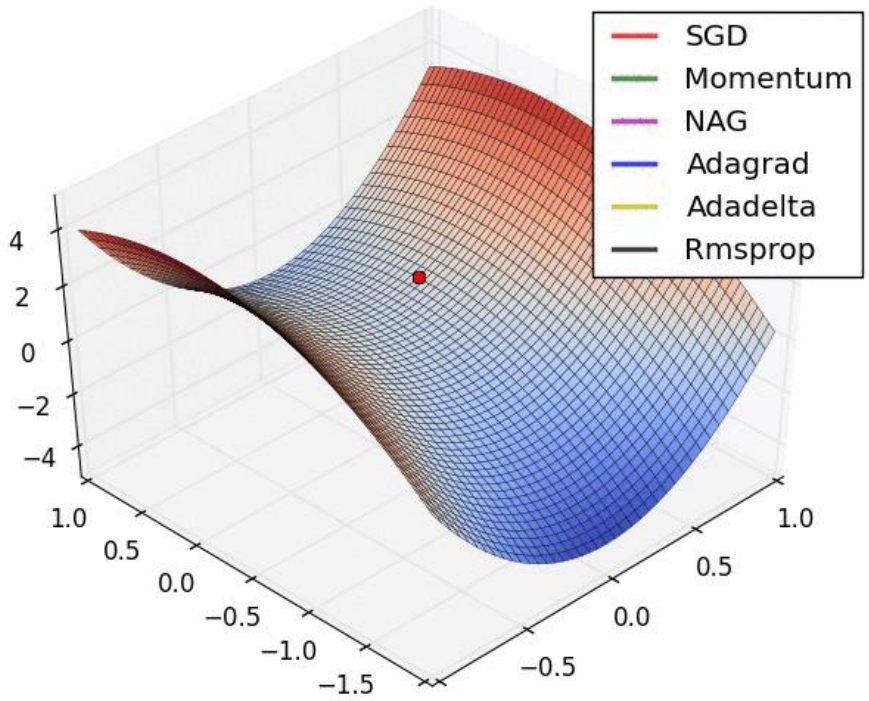
$$\Delta\theta_t = \rho\Delta\theta_{t-1} - \alpha \mathbf{g}_t$$

Adam is better choice!

    - 计算负梯度的“加权移动平均”作为参数的更新方向
  - Nesterov accelerated gradient
  - 梯度截断

优化


SGD
Momentum
NAG
Adagrad
Adadelta
Rmsprop


SGD
Momentum
NAG
Adagrad
Adadelta
Rmsprop
4
2
0
−2
−4
1.0
0.5
0.0
−0.5
−1.0
−1.5
1.0
0.5
0.0
−0.5

鞍点

超参数优化

- 超参数
  - 层数
  - 每层神经元个数
  - 激活函数
  - 学习率（以及动态调整算法）
  - 正则化系数
  - mini-batch 大小
- 优化方法
  - 网格搜索
  - 随机搜索
  - 贝叶斯优化
  - 动态资源分配
  - 神经架构搜索

重新思考泛化性

- 神经网络
  - 过度参数化
  - 拟合能力强

Zhang C, Bengio S, Hardt M, et al. Understanding deep learning requires rethinking generalization[J]. arXiv preprint arXiv:1611.03530, 2016.

正则化（regularization）

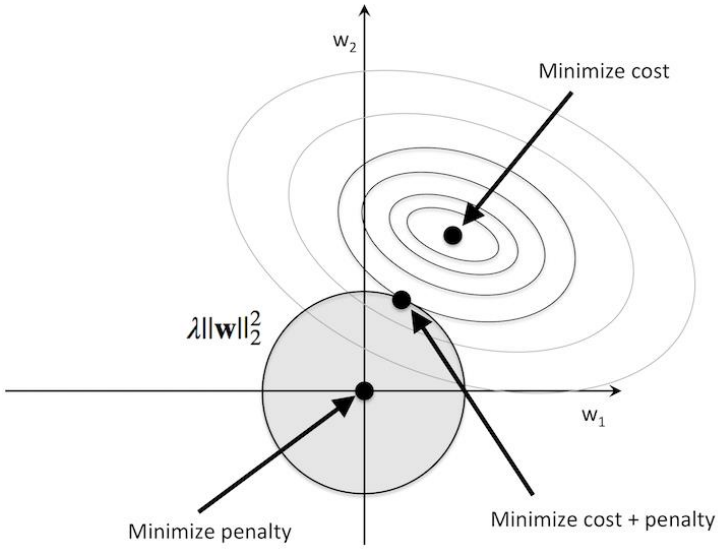
所有损害优化的方法都是正则化。

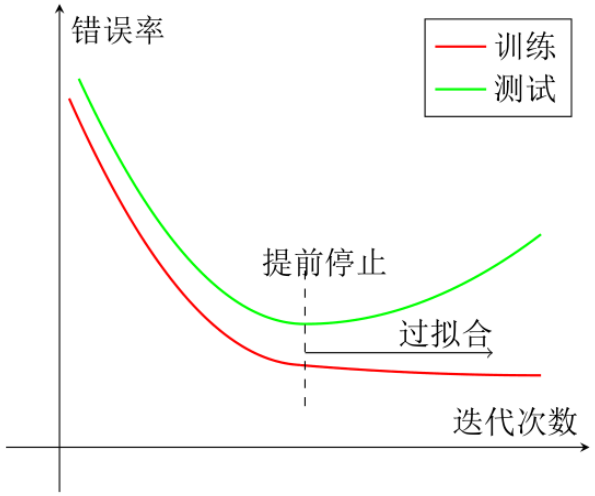
增加优化约束

L1/L2约束、数据增强

干扰优化过程

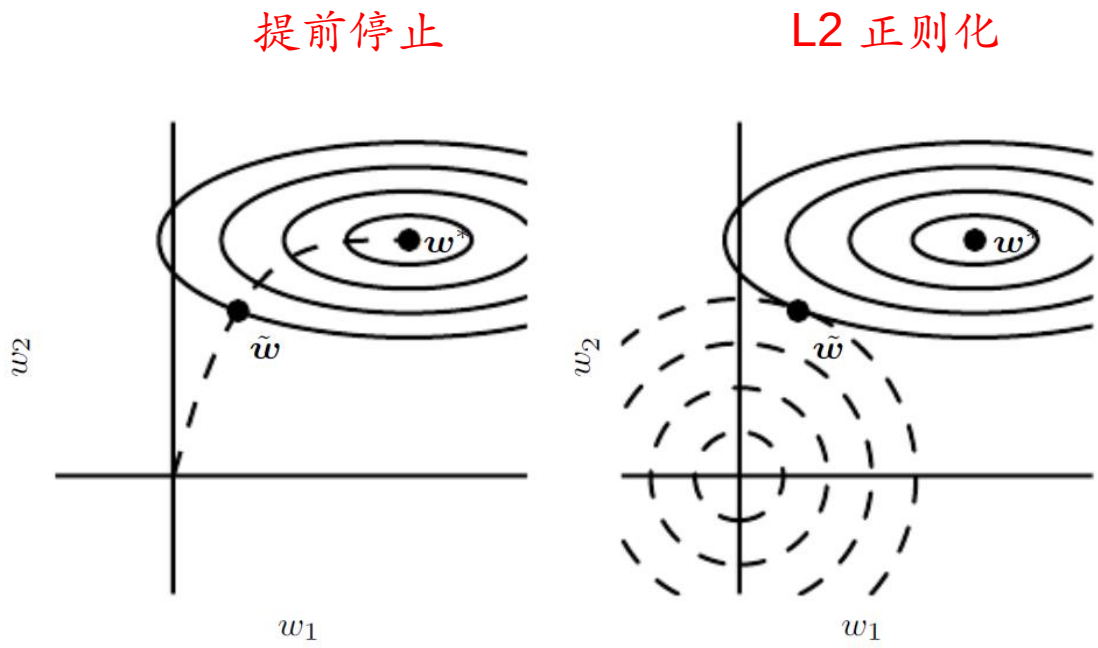
权重衰减、随机梯度下降、提前停止





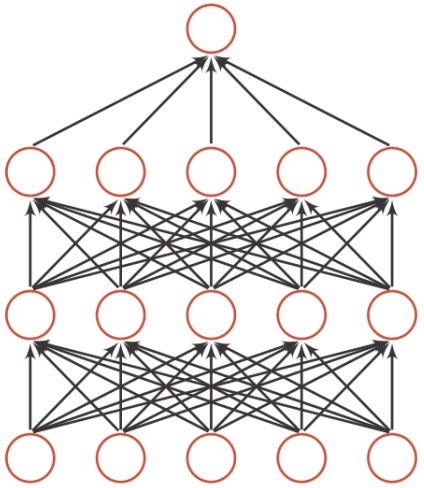
正则化

- 如何提高神经网络的泛化能力
  - 增加优化约束
    - 数据增强
    - 标签平滑
    - L1/L2 正则化
  - 干扰优化过程
    - 提前停止 early stop
    - 权重衰减
    - SGD
    - Dropout



Dropout

- 集成学习的解释
  - 原始网络可以近似看作是不同子网络的组合模型。

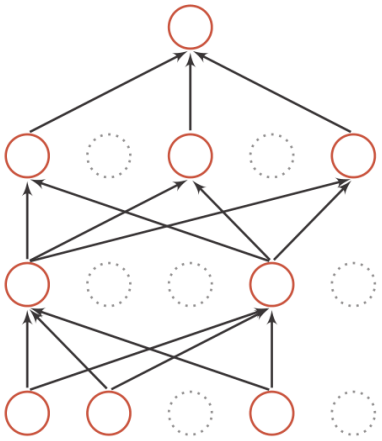


(a) 标准网络

- 贝叶斯学习的解释
  - 参数θ为随机向量，并且先验分布为q(θ)

$$\mathbb{E}_{q(\theta)}[y] = \int_q f(\mathbf{x}, \theta) q(\theta) d\theta$$

$$\approx \frac{1}{M} \sum_{m=1}^{M} f(\mathbf{x}, \theta_m),$$



(b) Dropout 后的网络

经验

- 用 ReLU 作为激活函数
- 分类时用交叉熵作为损失函数
- SDG+mini-batch
- 每次迭代都重新随机排序
- 数据预处理（标准归一化）
- 动态学习率（越来越小）
- 用 L1 或 L2 正则化（跳过前几轮）
- 逐层归一化
- dropout
- 数据增强

注意力与记忆机制

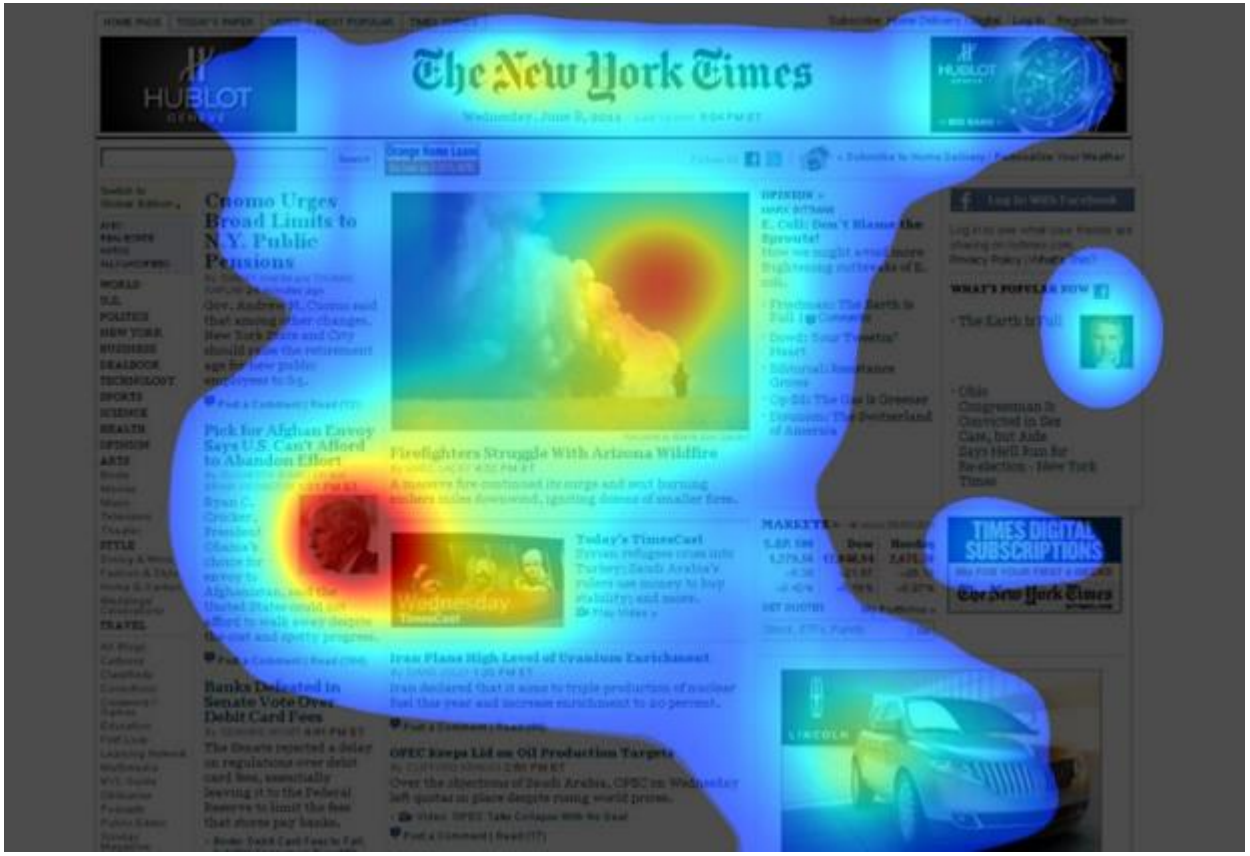
通用近似定理

- 由于优化算法和计算能力的限制，神经网络在实践中很难达到通用近似的能力。
  - 网络不能太复杂（参数太多）
- 如何提高网络能力
  - 局部连接
  - 权重共享
  - 汇聚操作
  - ？

大脑中的注意力

- 人脑每个时刻接收的外界输入信息非常多，包括来源于视觉、听觉、触觉的各种各样的信息。
- 但就视觉来说，眼睛每秒钟都会发送千万比特的信息给视觉神经系统。
- 人脑通过注意力来解决信息超载问题。

注意力示例


The New York Times
HUBLOT
Cuomo Urges Broad Limits to N.Y. Public Pensions
Pick for Afghan Envoy Says U.S. Can't Afford to Abandon Effort
Banks Defeated in Senate Vote Over Debit Card Fees
Firefighters Struggle With Arizona Wildfire
Today's TimesCast
Iran Plans High Level of Uranium Enrichment
OPEC Keeps Lid on Oil Production Targets
TIMES DIGITAL SUBSCRIPTIONS
LINCOLN

如何实现?

- 自下而上

汇聚（pooling）

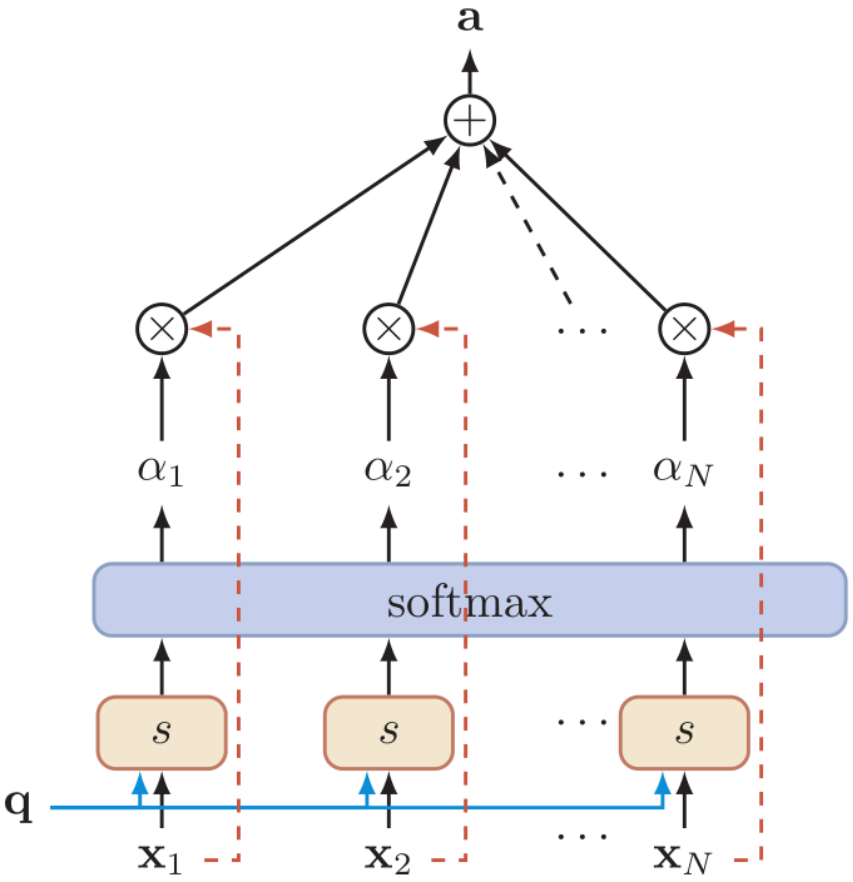
- 自上而下

会聚（focus）

注意力分布

- 给定查询q和输入信息$\mathbf{x}_{1:N}$

$$\begin{aligned}\alpha_i &= p(z=i|\mathbf{x}_{1:N},\mathbf{q}) \\ &= \text{softmax}\left(s(\mathbf{x}_i,\mathbf{q})\right) \\ &= \frac{\exp\left(s(\mathbf{x}_i,\mathbf{q})\right)}{\sum_{j=1}^{N}\exp\left(s(\mathbf{x}_j,\mathbf{q})\right)},\end{aligned}$$



- s($\mathbf{x}_i$,q)为注意力打分函数
  - 加性模型 $s(\mathbf{x}_i,\mathbf{q}) = \mathbf{v}^{\mathrm{T}}\tanh(W\mathbf{x}_i + U\mathbf{q}),$
  - 乘法模型 $s(\mathbf{x}_i,\mathbf{q}) = \mathbf{x}_i^{\mathrm{T}}\mathbf{q},$

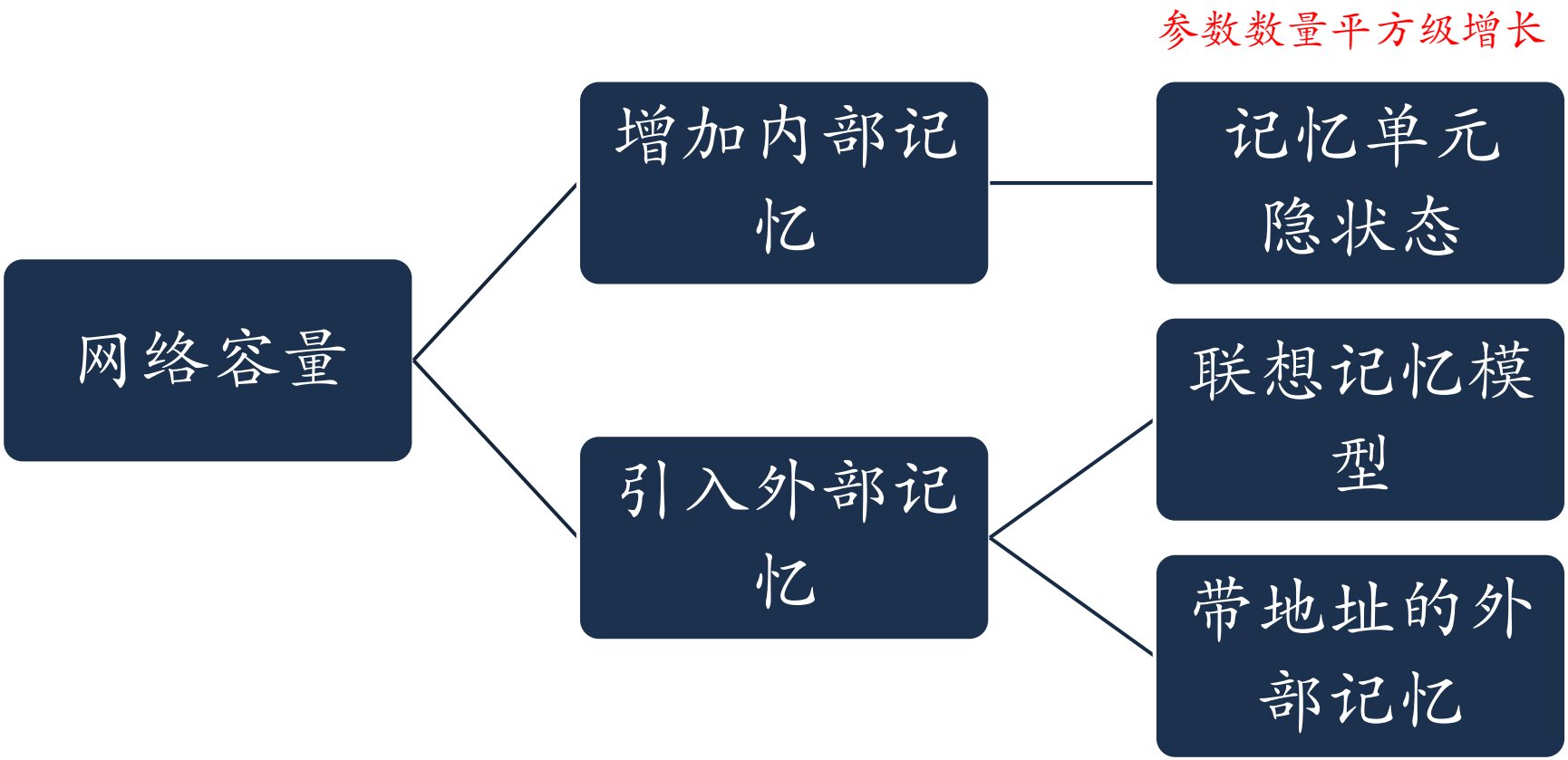
$$s(\mathbf{x}_i,\mathbf{q}) = \mathbf{x}_i^{\mathrm{T}}W\mathbf{q},$$

注意力的变种

- 多头注意力Multi-Head Attention
- 硬注意力 Hard Attention
- 结构化注意力 Structure Attention
- 指针网络 Pointer Network
- 双向注意力Bi-Directional Attention
- 键值对注意力 Key-Value Attention
- 自注意力 Self/Intra Attention
- …

如何增加网络容量?

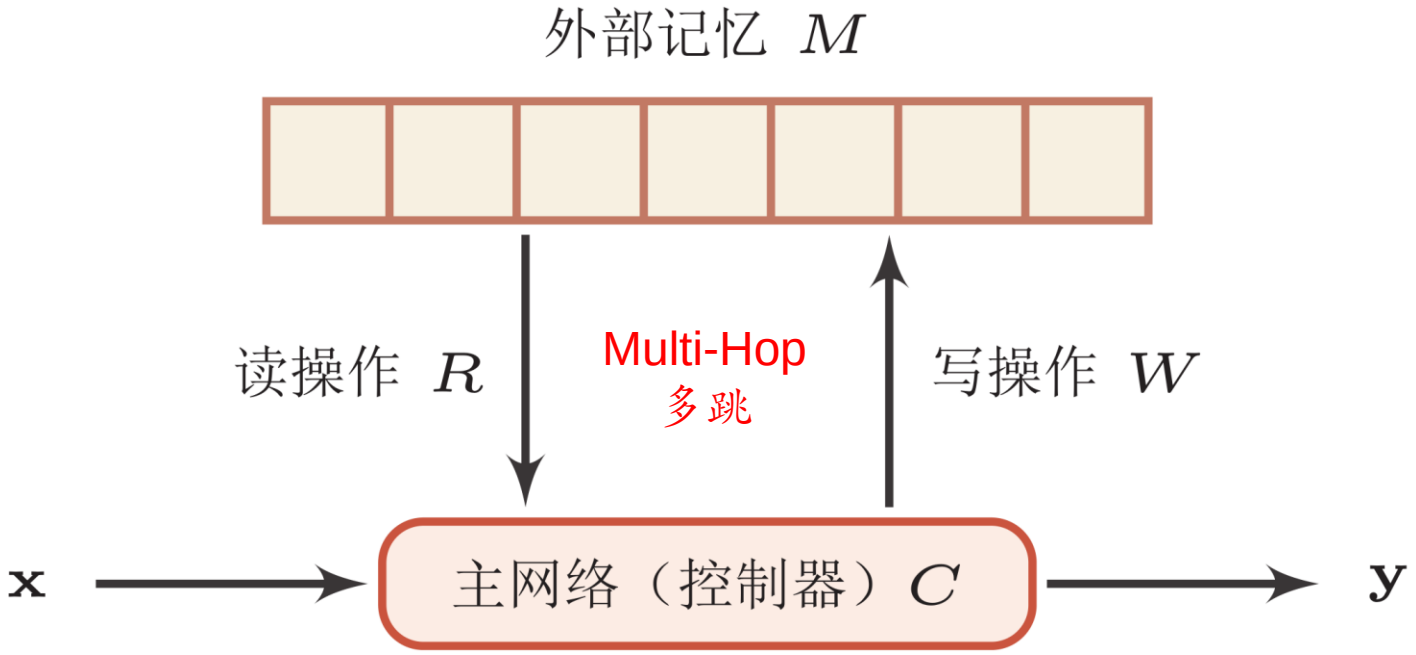
- 以LSTM为例，



外部记忆

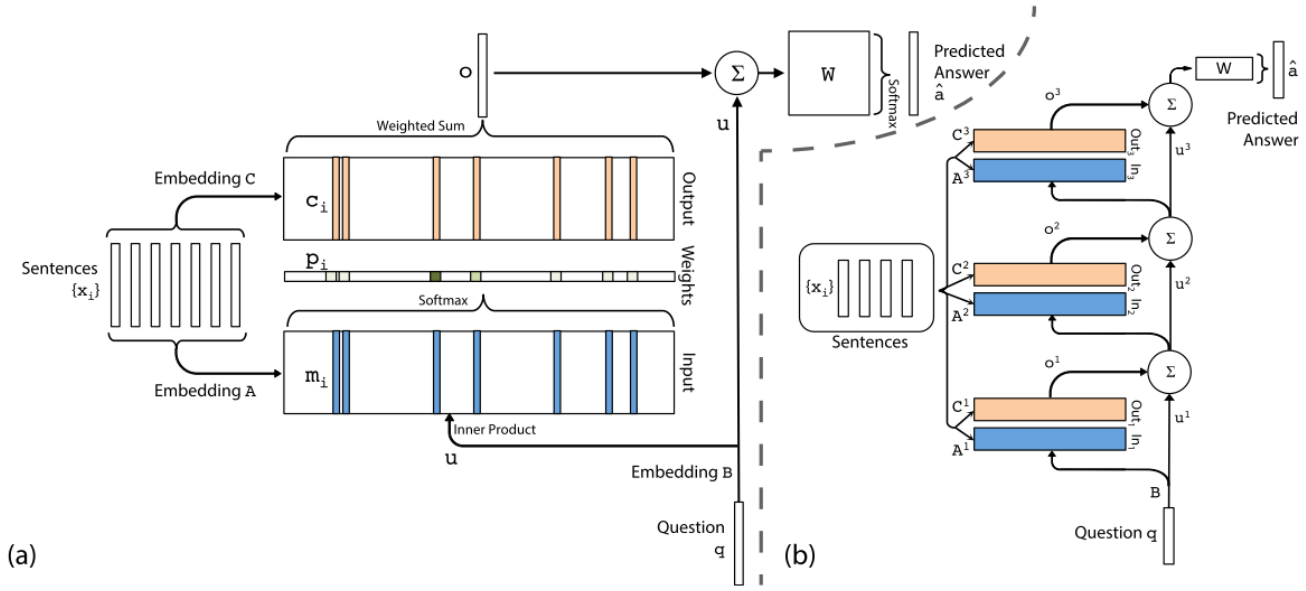
- 外部记忆定义为矩阵$\mathrm{M} \in \mathrm{R}^{\mathrm{d} \times k}$
  - k 是记忆片段的数量，d 是每个记忆片段的大小
- 外部记忆类型
  - 只读
    - Memory Network
    - RNN中的h_t
  - 可读写
    - NTM

记忆网络的结构



按内容寻址：通常利用注意力机制来完成。

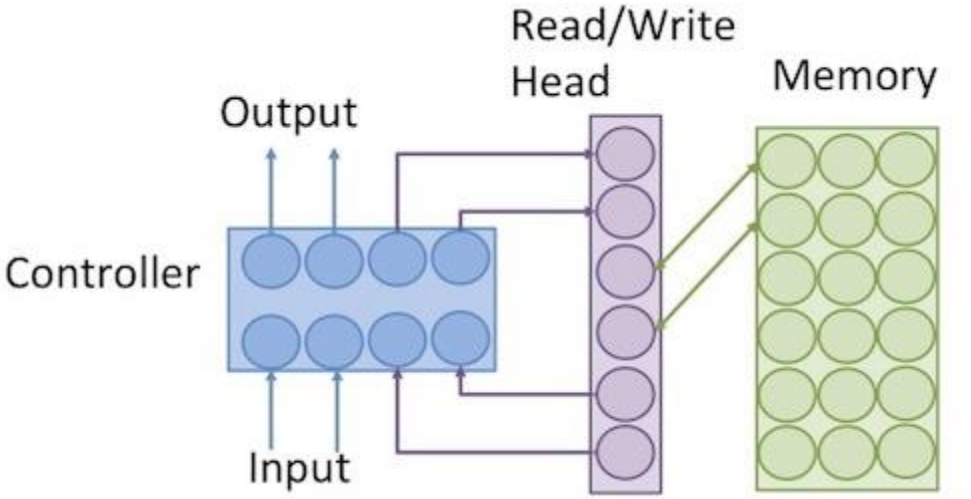
端到端记忆网络



Sukhbaatar, S., Szlam, A., Weston, J., & Fergus, R. (2015). End-To-End Memory Networks, 1–11. http://arxiv.org/abs/1503.08895

神经图灵机

- 组件
  - 控制器
  - 外部记忆
  - 读写操作
- 整个架构可微分



图片来源：
http://cpmarkchang.logdown.com/posts/279710-neural-network-neural-turing-machine

Graves, A., Wayne, G., & Danihelka, I. (2014). Neural Turing Machines. Arxiv, 1–26. http://arxiv.org/abs/1410.5401

不严格的类比

记忆周期	计算机	人脑	神经网络
短期	寄存器	工作记忆	隐状态
中期	内存	情景记忆	外部记忆
长期	外存	结构记忆	可学习参数

无监督学习

无监督学习

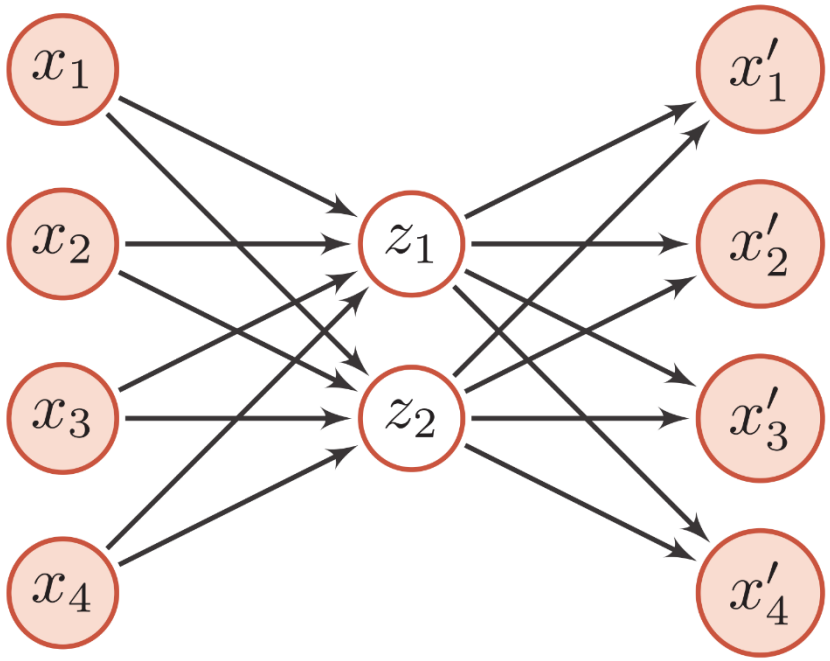
- 密度估计
  - 有参模型
    - 玻尔兹曼机、深度信念网络、深度生成模型
  - 无参模型
- 特征学习
  - 主成分分析
  - 稀疏编码
  - 自编码器
- 聚类

自编码器

- 最简单的自编码器是两层的神经网络，输入层到隐藏层用来编码，隐藏层到输出层用来解码。

- 模型
  - 编码器（encoder）
  - 解码器（decoder）

- 学习准则
  - 最小化重构错误



概率图模型

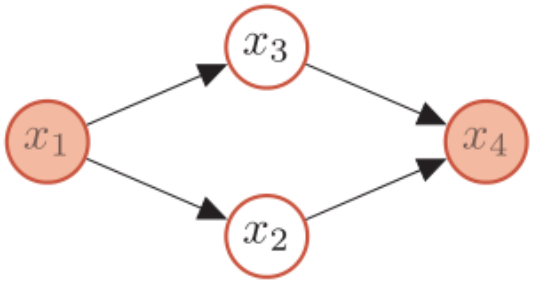
概率图模型

- 概率图模型是指一种用图结构来描述多元随机变量之间条件独立关系的概率模型。

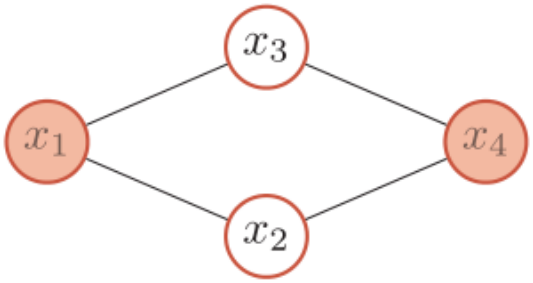
- 图中的每个节点都对应一个随机变量，可以是观察变量，隐变量或是未知参数等；每个连接表示两个随机变量之间具有依赖关系。

概率图模型

- 模型表示（图结构）
  - 有向图
  - 无向图



(a) 有向图：贝叶斯网络



(b) 无向图：马尔可夫随机场

- 推断
  - 给定部分变量，推断另一部分变量的后验概率。

- （参数）学习
  - 给定一组训练样本，求解网络参数

贝叶斯网络

- 对于K 随机变量$X = X_1, \cdots, X_K$和一个有向非循环图G，并定义X_{π_k}表示变量X_k的所有父节点变量集合。
- 如果X的联合概率分布可以分解为每个随机变量X_k的局部条件概率的连乘形式，那么(G,X)构成了一个贝叶斯网络。

$$p(x_1, x_2, \cdots, x_k) = \prod_{k=1}^{K} p(x_k | x_{\pi_k})$$

局部马尔可夫性质

- 利用图模型的局部马尔可夫性，我们可以对多元变量的联合概率进行简化，从而降低建模的复杂度。
- 以贝叶斯网络为例，

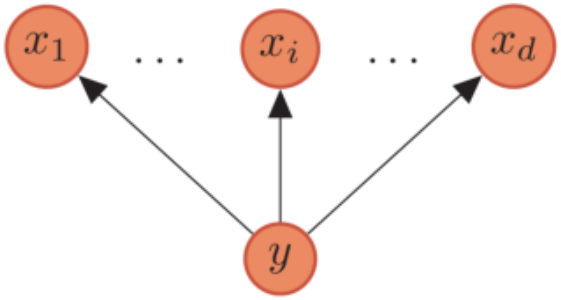
$$p(x_1, x_2, x_3, x_4) = p(x_1)p(x_2|x_1)p(x_3|x_1)p(x_4|x_2, x_3),$$

  - 是4个局部条件概率的乘积，这样只需要1 + 2 + 2 + 4 = 9个独立参数。

常见的有向图模型

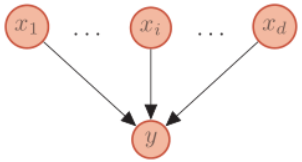
- 朴素贝叶斯分类器
  - 给定一个有d维特征的样本x和类别y，类别的后验概率为

$$p(y|\mathbf{x}) \propto p(y) \prod_{i=1}^{d} p(x_i|y)$$

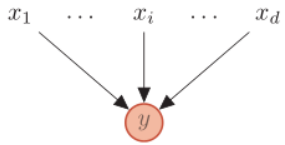


- Sigmoid信念网络
  - Sigmoid信念网络网络中的变量为二值变量，取值为 {0,1}。

$$p(x_k = 1|x_{\pi_k}) = \sigma(w_0 + \sum_{x_i \in \pi_k} w_i x_i),$$



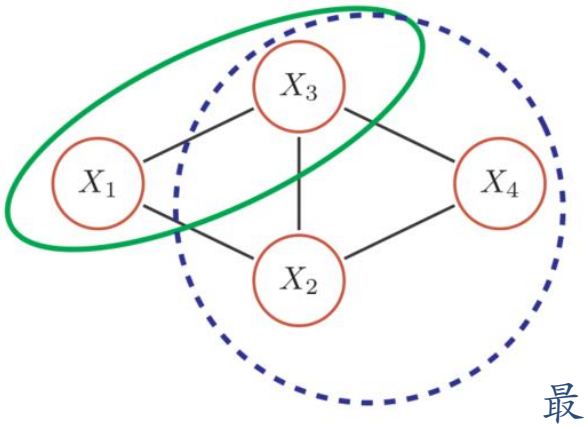
(a) Sigmoid信念网络



(b) Logistic回归

马尔可夫随机场

- 马尔可夫随机场，也称无向图模型，是一类用无向图来表示一组具有马尔可夫性质的随机变量X的联合概率分布模型。



最大团

马尔可夫网络

- 马尔可夫网络的联合分布可以表示为

$$P(\mathbf{X}=\mathbf{x}) = \frac{1}{Z}\prod_{c\in\mathcal{C}}\exp(-E(X_c))$$
$$= \frac{1}{Z}\exp(\sum_{c\in\mathcal{C}}-E(X_c))$$

  - 其中 $E(X_c)$ 为能量函数，Z 是配分函数。

常见的无向图模型

- 对数线性模型
  - 势能函数的一般定义为

$$\phi_c(\mathbf{x}_c|\theta_c) = \exp\left(\theta_c^{\mathrm{T}} f_c(\mathbf{x}_c)\right)$$

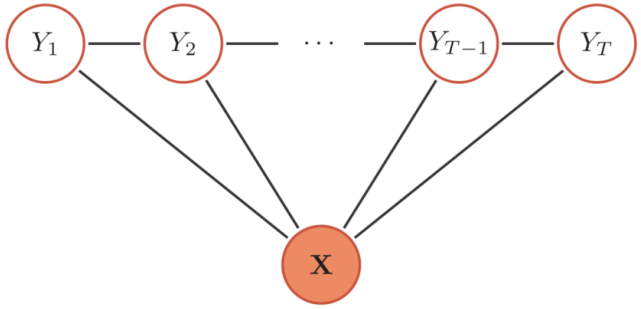
  - 联合概率p(x)的对数形式为

$$\log p(\mathbf{x}|\theta) = \sum_{c\in C} \theta_c^{\mathrm{T}} f_c(\mathbf{x}_c) - \log Z(\theta)$$

  - 也称为最大熵模型
- 条件随机场
  - y一般为随机向量
  - 条件概率p(y|x)

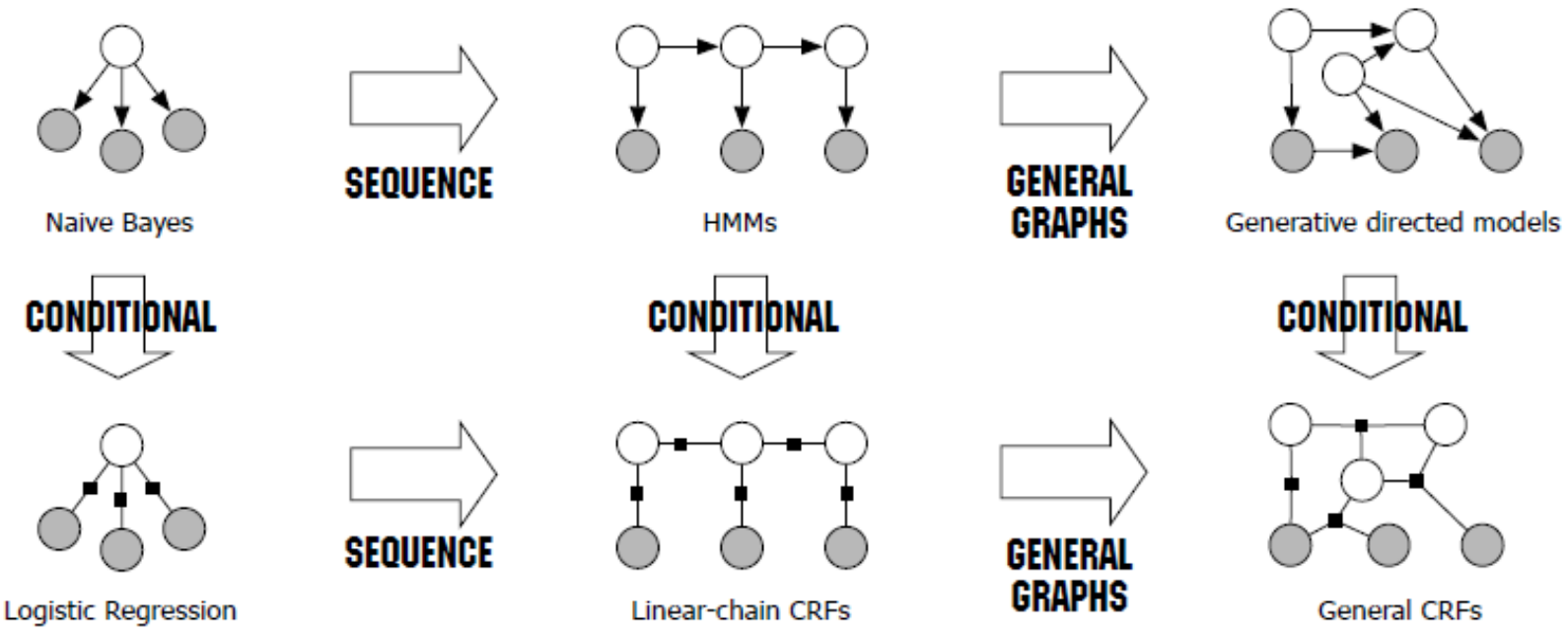
$$p(\mathbf{y}|\mathbf{x},\theta) = \frac{1}{Z(\mathbf{x},\theta)} \exp\left(\sum_{c\in\mathcal{C}} \theta_c^{\mathrm{T}} f_c(\mathbf{x},\mathbf{y}_c)\right)$$

(a) 最大熵模型



(b) 线性链的条件随机场

模型对比


Naive Bayes
SEQUENCE
HMMs
GENERAL GRAPHS
Generative directed models
CONDITIONAL
CONDITIONAL
CONDITIONAL
Logistic Regression
SEQUENCE
Linear-chain CRFs
GENERAL GRAPHS
General CRFs

推断（inference）

- 推断
  - 指在观测到部分变量e时，计算其它变量z的某个子集q 的后验概率p(q|e)。

- 根据贝叶斯公式有

$$p(\mathbf{q}|\mathbf{e}) = \frac{p(\mathbf{q},\mathbf{e})}{p(\mathbf{e})} = \frac{\sum_{\mathbf{z}} p(\mathbf{q},\mathbf{e},\mathbf{z})}{\sum_{\mathbf{q},\mathbf{z}} p(\mathbf{q},\mathbf{e},\mathbf{z})}.$$

  - 图模型的推断问题可以转换为求任意一个变量子集的边际概率分布问题。

推断方法

- 常用的推断方法可以分为精确推断和近似推断：
  - 精确推断（Exact Inference）
    - 信念传播（Belief Propagation，BP）算法
      - 也称为消息传递（Message Passing）算法
  - 近似推断（Approximate Inference）
    - 环路信念传播（Loopy Belief Propagation，LBP）
    - 变方法（Variational Method）
    - 采样法（Sampling Method）

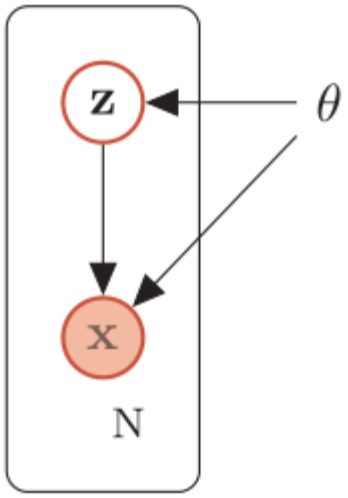
参数学习

- 在贝叶斯网络中，所有变量x的联合概率分布可以分解为每个随机变量x_k的局部条件概率的连乘形式。
- 假设每个局部条件概率$p(x_k \mid x_{\pi(k)})$的参数为θ_k，则x的对数似然函数为

$$\log p(\mathbf{x}, \Theta) = \sum_{k=1}^{K} \log p(x_k | x_{\pi_k}, \theta_k),$$

EM算法

- 假设有一组变量，有部分变量是是不可观测的，如何进行参数估计呢?
- 期望最大化算法
  - Expectation-Maximum，EM算法



$$
\begin{aligned}
\log p(\mathbf{x}|\theta) &= \sum_{\mathbf{z}} q(\mathbf{z}) \log p(\mathbf{x}|\theta) \\
&= \sum_{\mathbf{z}} q(\mathbf{z}) \Big(\log p(\mathbf{x}, \mathbf{z}|\theta) - \log p(\mathbf{z}|\mathbf{x}, \theta) \Big) \\
&= \sum_{\mathbf{z}} q(\mathbf{z}) \log \frac{p(\mathbf{x}, \mathbf{z}|\theta)}{q(\mathbf{z})} - \sum_{\mathbf{z}} q(\mathbf{z}) \log \frac{p(\mathbf{z}|\mathbf{x}, \theta))}{q(\mathbf{z})} \\
&= ELBO(q, \mathbf{x}|\theta) + D_{\mathrm{KL}}(q(\mathbf{z}) \| p(\mathbf{z}|\mathbf{x}, \theta)),
\end{aligned}
$$

EM算法

$$ELBO(q, \mathbf{x}|\theta) + D_{\mathrm{KL}}(q(\mathbf{z})\|p(\mathbf{z}|\mathbf{x}, \theta))$$

- E步

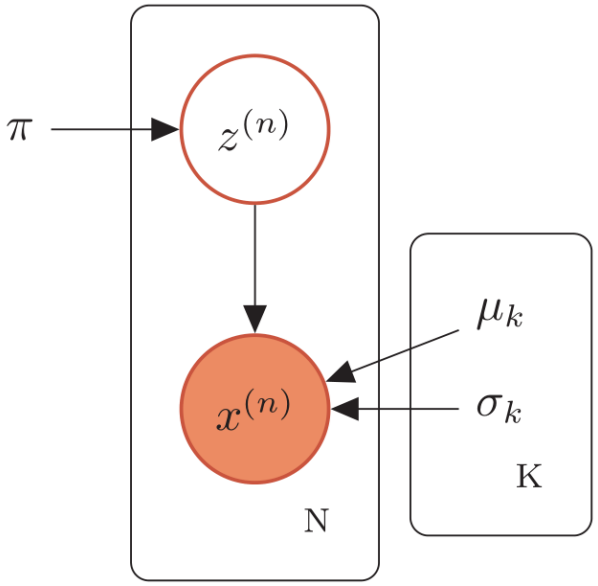
$$q(\mathbf{z}) = p(\mathbf{z}|\mathbf{x}, \theta_t)$$

- M步

$$\theta_{t+1} = \arg\max_{\theta} ELBO(q_{t+1}, \mathbf{x}|\theta)$$

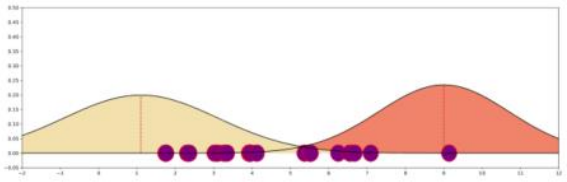
高斯混合模型

- 高斯混合模型（Gaussian Mixture Model，GMM）是由多个高斯分布组成的模型，其密度函数为多个高斯密度函数的加权组合。

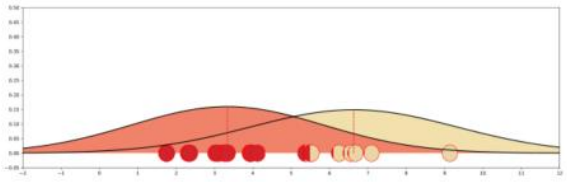


$$\mathcal{N}(x|\mu_k,\sigma_k)=\frac{1}{\sqrt{2\pi}\sigma_k}\exp\left(-\frac{(x-\mu_k)^2}{2\sigma_k^2}\right)$$

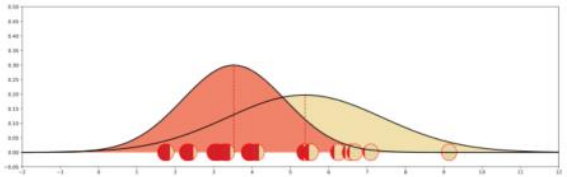
高斯混合模型的参数学习



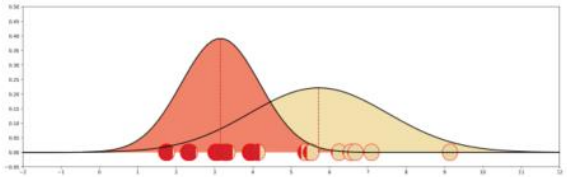
(a) 初始化



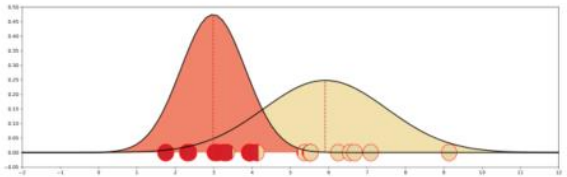
(b) 第 1 次迭代



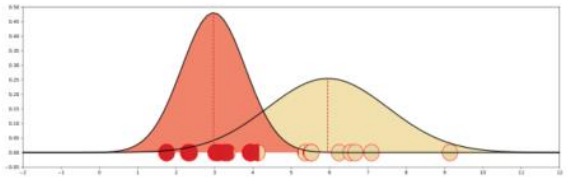
(c) 第 4 次迭代



(d) 第 8 次迭代

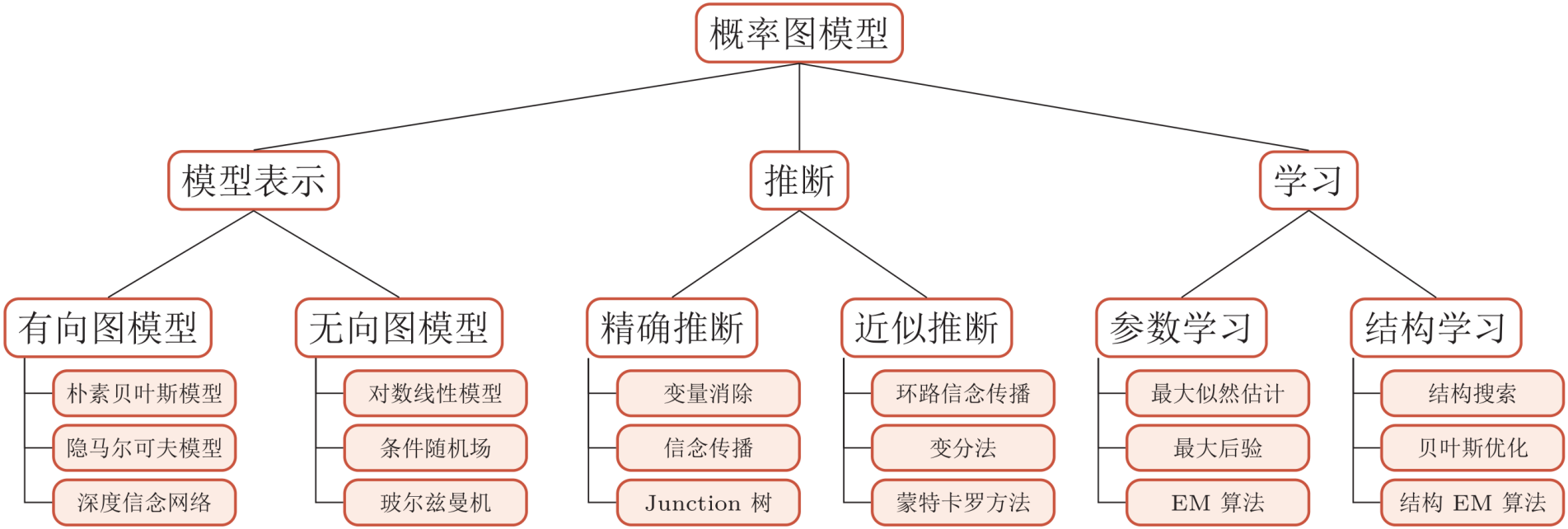


(e) 第 12 次迭代



(f) 第 16 次迭代

概率图模型


概率图模型
模型表示
推断
学习
有向图模型
无向图模型
精确推断
近似推断
参数学习
结构学习
朴素贝叶斯模型
隐马尔可夫模型
深度信念网络
对数线性模型
条件随机场
玻尔兹曼机
变量消除
信念传播
Junction 树
环路信念传播
变分法
蒙特卡罗方法
最大似然估计
最大后验
EM 算法
结构搜索
贝叶斯优化
结构 EM 算法

玻尔兹曼机

玻尔兹曼机（Boltzmann machine）

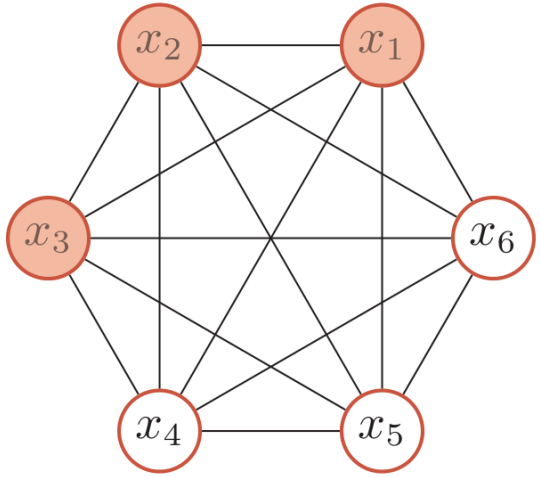
- 玻尔兹曼机是一个特殊的概率无向图模型。
  - 每个随机变量是二值的
  - 所有变量之间是全连接的
  - 整个能量函数定义为

$$E(\mathbf{x}) \triangleq E(\mathbf{X}=\mathbf{x})$$

$$= -\left(\sum_{i<j} w_{ij}\, x_i\, x_j + \sum_i b_i\, x_i\right)$$

  - P(X) 为玻尔兹曼分布

$$P(\mathbf{X}=\mathbf{x}) = \frac{1}{Z}\exp\left(\frac{-E(\mathbf{x})}{T}\right)$$



一个有六个变量的玻尔兹曼机

两个基本问题：
1. 推断p(h|v)
2. 参数学习W

玻尔兹曼机的推断

- 近似采样--Gibbs采样

$$P(X_i = 1|\mathbf{x}_{\backslash i}) = \sigma\left(\frac{\Delta E_i(\mathbf{x}_{\backslash i})}{T}\right)$$

- 模拟退火
  - 让系统刚开始在一个比较高的温度下运行，然后逐渐降低，直到系统在一个比较低的温度下达到热平衡。
  - 当系统温度非常高$T \to \infty$时，$p_i \to 0.5$，即每个变量状态的改变十分容易，每一种网络状态都是一样的，而从很快可以达到热平衡。
  - 当系统温度非常低$T \to 0$时，如果$\Delta E_i(x_{\backslash i}) > 0$则$p_i \to 1$，如果$\Delta E_i(x_{\backslash i}) < 0$则$p_i \to 0$。
    - 随机性方法变成确定性方法

玻尔兹曼机的参数学习

- 最大似然估计

$$\mathcal{LL}(\mathcal{W}) = \frac{1}{N}\sum_{i=1}^{N}\log p(\hat{\mathbf{v}}^{(n)})$$

- 采用梯度上升法

$$w_{ij} \leftarrow w_{ij} + \alpha\left(\frac{1}{N}\sum_{n=1}^{N}\mathbb{E}_{p(\mathbf{h}|\hat{\mathbf{v}}^{(n)})}[x_i x_j] - \mathbb{E}_{p(\mathbf{x})}[x_i x_j]\right)$$

玻尔兹曼机的参数学习

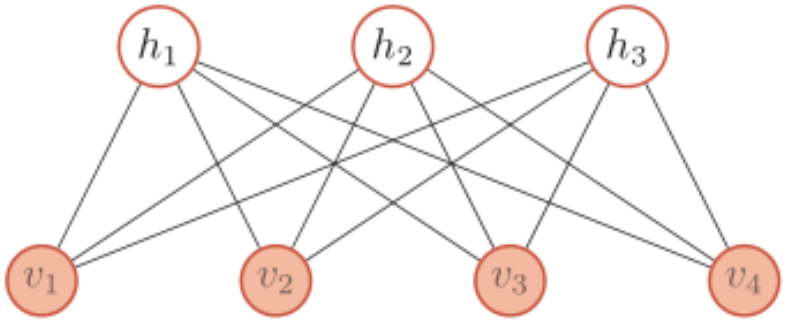
- 基于Gibbs采样来进行近似求解

$$w_{ij} \leftarrow w_{ij} + \alpha \left(\frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{p(\mathbf{h}|\hat{\mathbf{v}}^{(n)})}[x_i x_j] - \mathbb{E}_{p(\mathbf{x})}[x_i x_j] \right)$$

$$w_{ij} \leftarrow w_{ij} + \alpha \left(\langle x_i x_j \rangle_{\texttt{data}} - \langle x_i x_j \rangle_{\texttt{model}} \right)$$

受限玻尔兹曼机（Restricted Boltzmann Machines，RBM）

- 受限玻尔兹曼机是一个二分图结构的无向图模型。
  - 在受限玻尔兹曼机中，变量可以为两组，分别为隐藏层和可见层（或输入层）。
  - 节点变量的取值为0或1。
  - 和两层的全连接神经网络的结构相同。



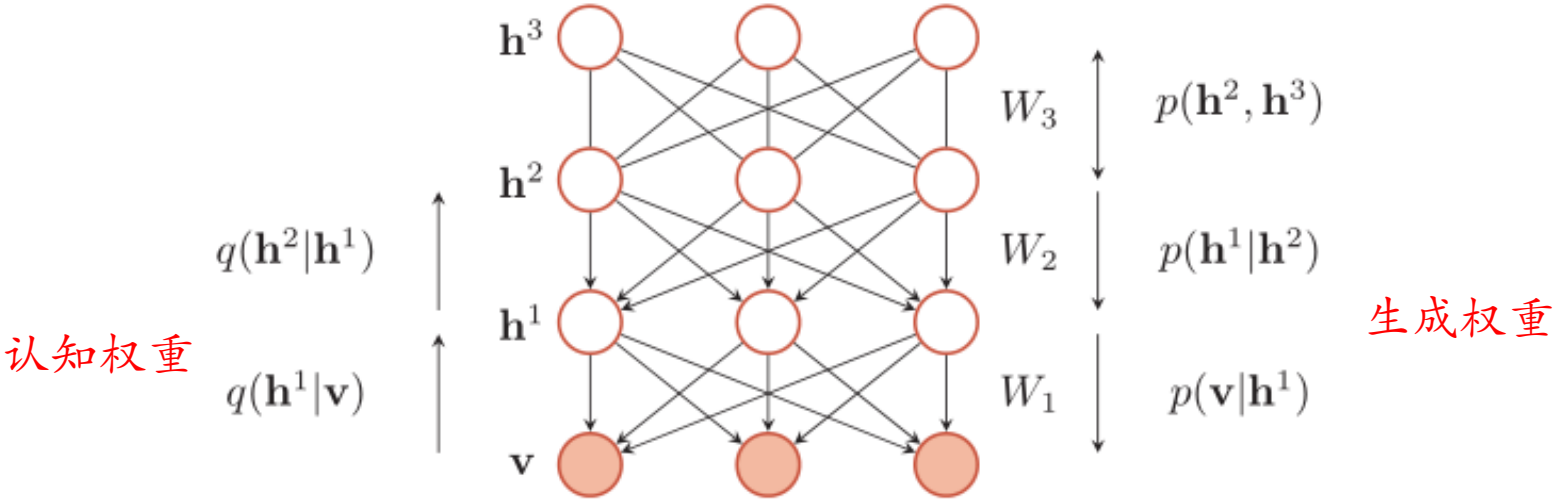
$$p(v_i = 1|\mathbf{h}) = \sigma\left(a_i + \sum_j w_{i,j} h_j\right),$$

$$p(h_j = 1|\mathbf{v}) = \sigma\left(b_j + \sum_i w_{i,j} v_i\right),$$

深度信念网络

深度信念网络（Deep Belief Network, DBN）

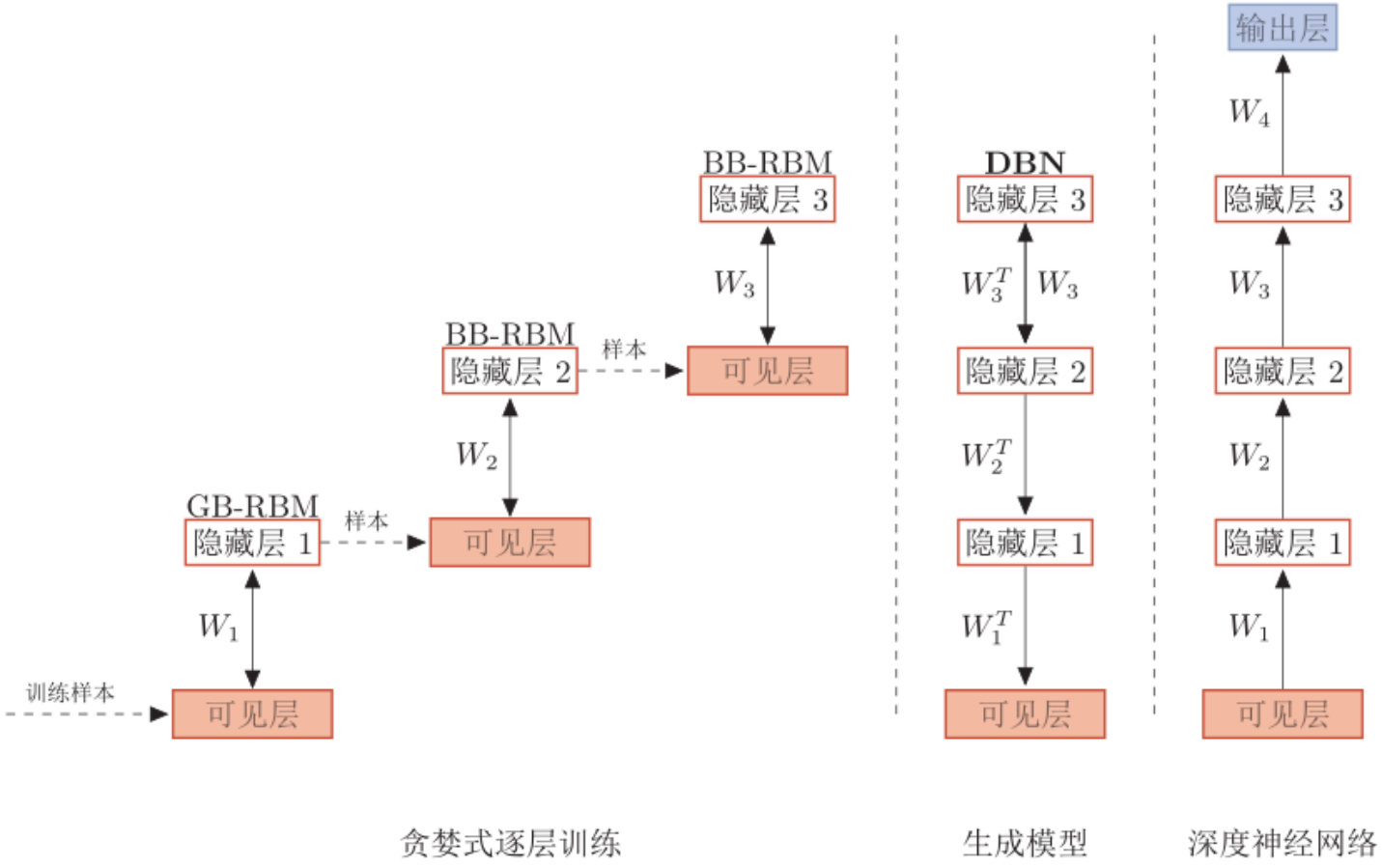
- 深度信念网络是深度的有向的概率图模型，其图结构由多层的节点构成。
  - 和全连接的神经网络结构相同。
  - 顶部的两层为一个无向图，可以看做是一个受限玻尔兹曼机。



$$p(\mathbf{h}^{(l)}|\mathbf{h}^{(l+1)}) = \sigma\left(\mathbf{b}^{(l+1)} + W^{(l+1)}\mathbf{h}^{(l+1)}\right)$$

训练深度信念网络-逐层训练

- 逐层训练是能够有效训练深度模型的最早的方法。



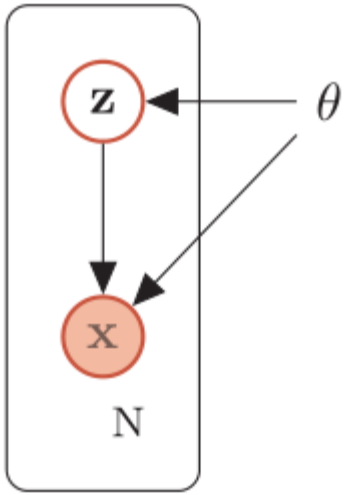
深度生成模型

深度生成模型

- 深度生成模型就是利用神经网络来建模条件分布 $p(x|z;\theta)$。
  - 对抗生成式网络（Generative Adversarial Network，GAN）[Goodfellow et al., 2014]
  - 变分自编码器（Variational Autoencoder，VAE）[Kingma and Welling, 2013, Rezende et al., 2014]。

生成模型

- 生成模型指一系列用于随机生成可观测数据的模型。
- 生成数据x的过程可以分为两步进行：
  - 根据隐变量的先验分布$p(z;\theta)$进行采样，得到样本z；
  - 根据条件分布$p(x|z;\theta)$进行采样，得到x。



变分自编码器

EM算法回顾

- 给定一个样本x，其对数边际似然$\log p(x|\theta)$可以分解为

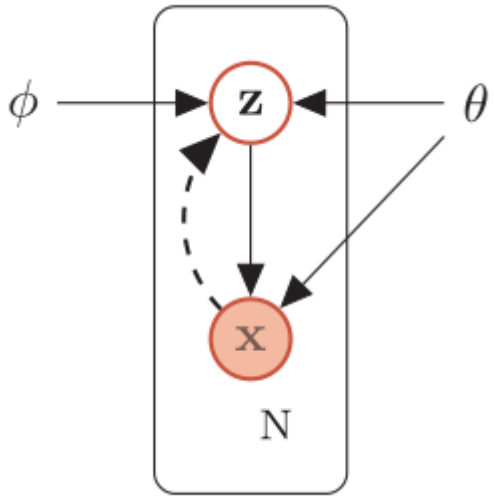
$$\begin{aligned}\log p(\mathbf{x}|\theta) &= \sum_{\mathbf{z}} q(\mathbf{z}) \log \frac{p(\mathbf{x},\mathbf{z}|\theta)}{q(\mathbf{z})} - \sum_{\mathbf{z}} q(\mathbf{z}) \log \frac{p(\mathbf{z}|\mathbf{x},\theta))}{q(\mathbf{z})} \\ &= ELBO(q,\mathbf{x}|\theta) + D_{\mathrm{KL}}(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x},\theta)),\end{aligned}$$

M step　　E step

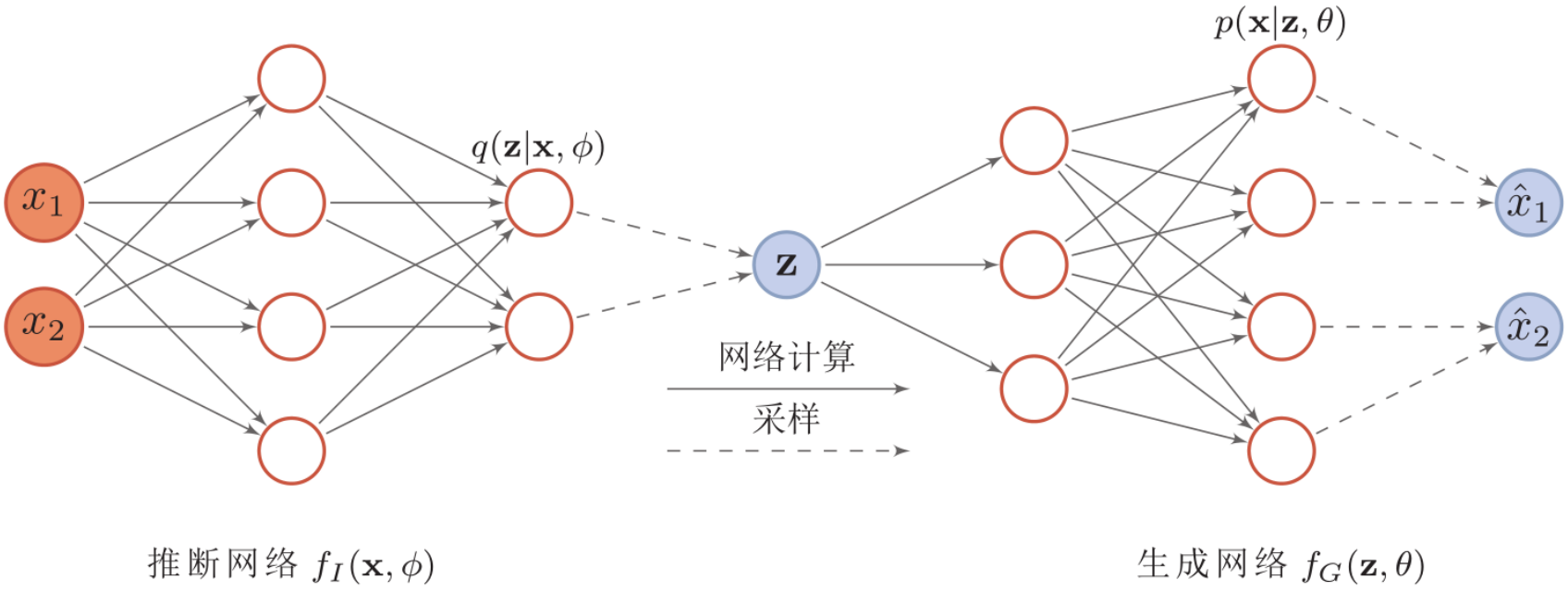
变分自编码器(VAE)

- 变分自编码器的模型结构可以分为两个部分：
  - 寻找后验分布$p(z|x;\theta)$的变分近似$q(z|x;\phi^*)$；
    - 变分推断：用简单的分布q去近似复杂的分$p(z|x;\theta)$
  - 在已知$q(z|x;\phi_*)$的情况下，估计更好的生成$p(x|z;\theta)$。

用神经网络来替代



变分自编码器


$p(\mathbf{x}|\mathbf{z},\theta)$
$q(\mathbf{z}|\mathbf{x},\phi)$
x_1
x_2
$\mathbf{z}$
$\hat{x}_1$
$\hat{x}_2$
网络计算
采样
推断网络 $f_I(\mathbf{x},\phi)$
生成网络 $f_G(\mathbf{z},\theta)$

模型汇总

$$\max_{\theta,\phi} ELBO(q, \mathbf{x}|\theta, \phi) = \max_{\theta,\phi} \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\phi)} \left[\log \frac{p(\mathbf{x}|\mathbf{z}, \theta)p(\mathbf{z}|\theta)}{q(\mathbf{z}|\phi)} \right]$$

$$\max_{\theta,\phi} \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x},\phi)} \Big[\log p(\mathbf{x}|\mathbf{z}, \theta) \Big] - D_{\mathrm{KL}} \Big(q(\mathbf{z}|\mathbf{x}, \phi) || p(\mathbf{z}|\theta) \Big)$$

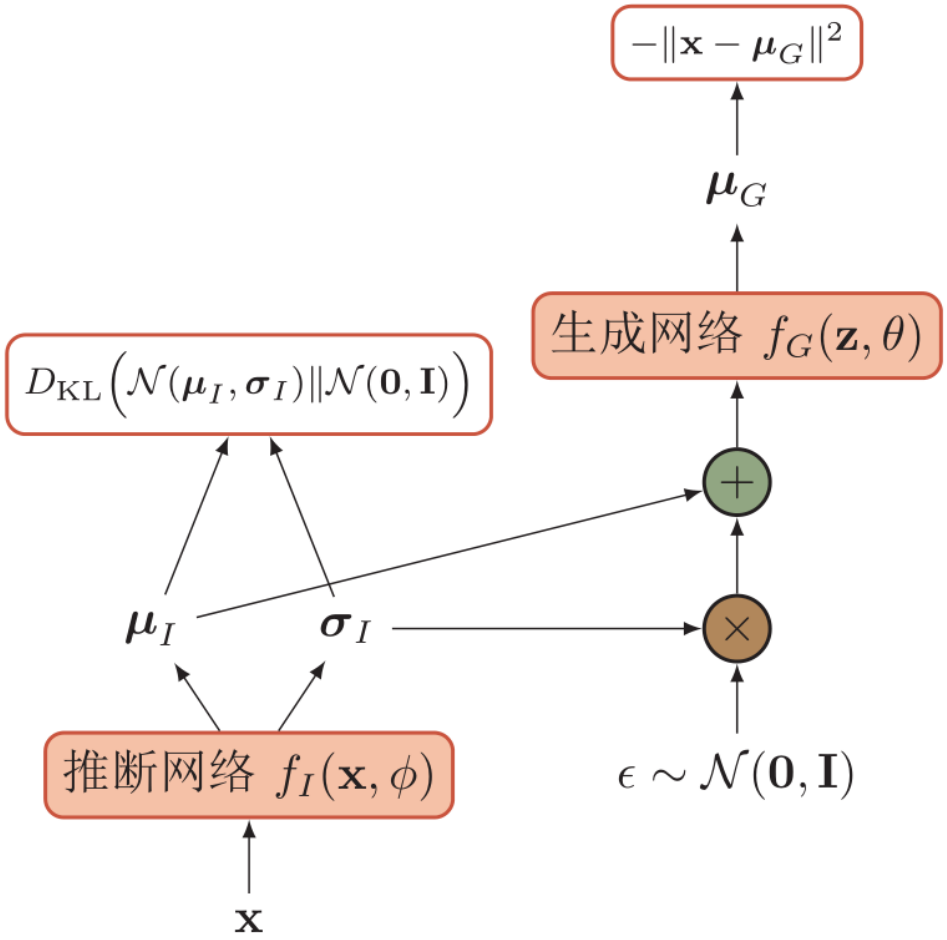
再参数化

- 分布$q(z|x,\phi)$依赖于参数ϕ

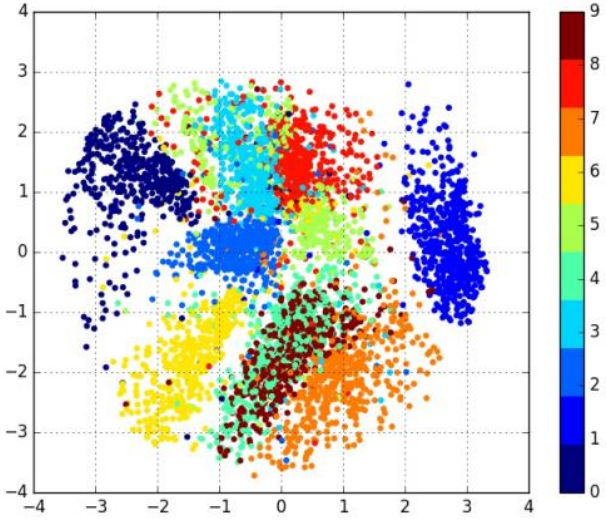
- 再参数化（reparameterization）是实现通过随机变量实现反向传播的一种重要手段

$$z \sim N(\mu_l, \sigma_l^2 I) \quad \Longrightarrow \quad \begin{aligned} &\epsilon \sim N(0, I) \\ &z = \mu_l + \sigma_l \odot \epsilon, \end{aligned}$$

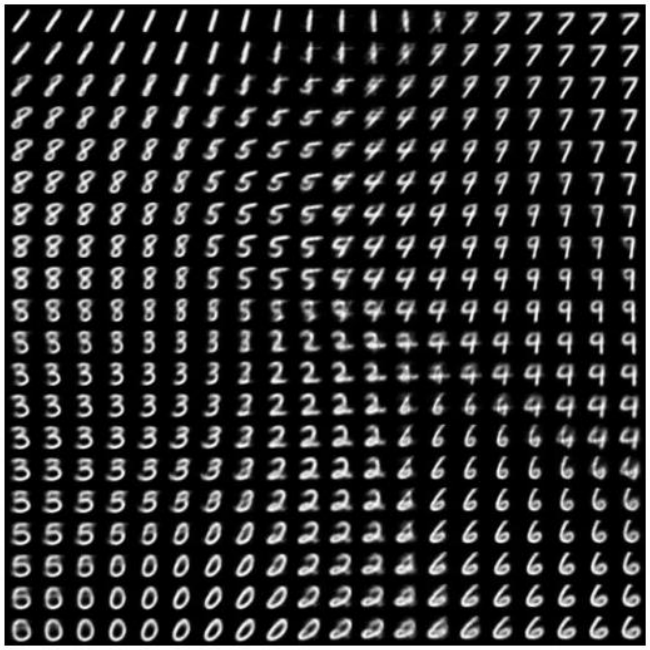
变分自编码器的训练过程



变分自编码器学习到的隐变量流形



(a) 训练集上所有样本在隐空间上的投影。

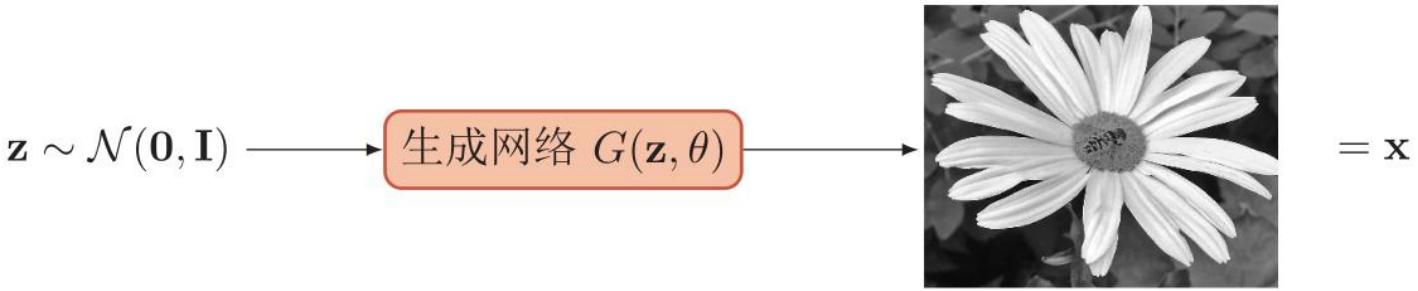


(b) 隐变量 $\mathbf{z}$ 在图像空间的投影。

生成对抗网络

显式密度模型和隐式密度模型

- 显式密度模型
  - 显示地构建出样本的密度函数$p(x|\theta)$，并通过最大似然估计来求解参数；
  - 变分自编码器、深度信念网络
- 隐式密度模型
  - 不显示地估计出数据分布的密度函数
  - 但能生成符合数据分布$p_{data}(x)$的样本
  - 无法用最大似然估计



生成对抗网络

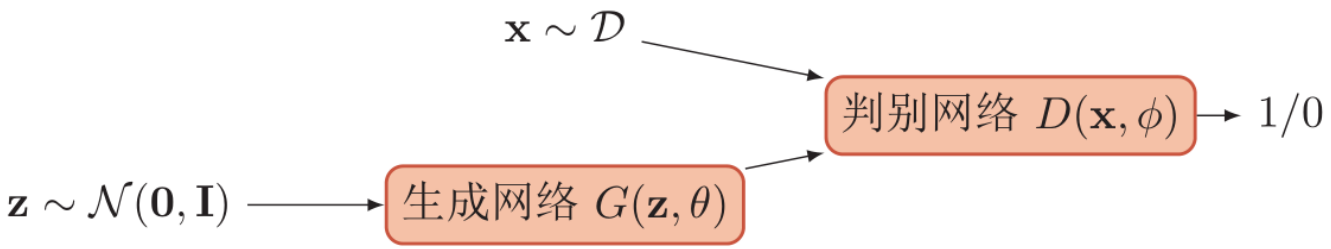
- 生成对抗网络由一个生成网络与一个判别网络组成。生成网络从潜在空间（latent space）中随机采样作为输入，其输出结果需要尽量模仿训练集中的真实样本。
- 判别网络的输入则为真实样本或生成网络的输出，其目的是将生成网络的输出从真实样本中尽可能分辨出来。

MinMax Game

- 生成网络要尽可能地欺骗判别网络。
- 判别网络将生成网络生成的样本与真实样本中尽可能区分出来。
- 两个网络相互对抗、不断调整参数，最终目的是使判别网络无法判断生成网络的输出结果是否真实。

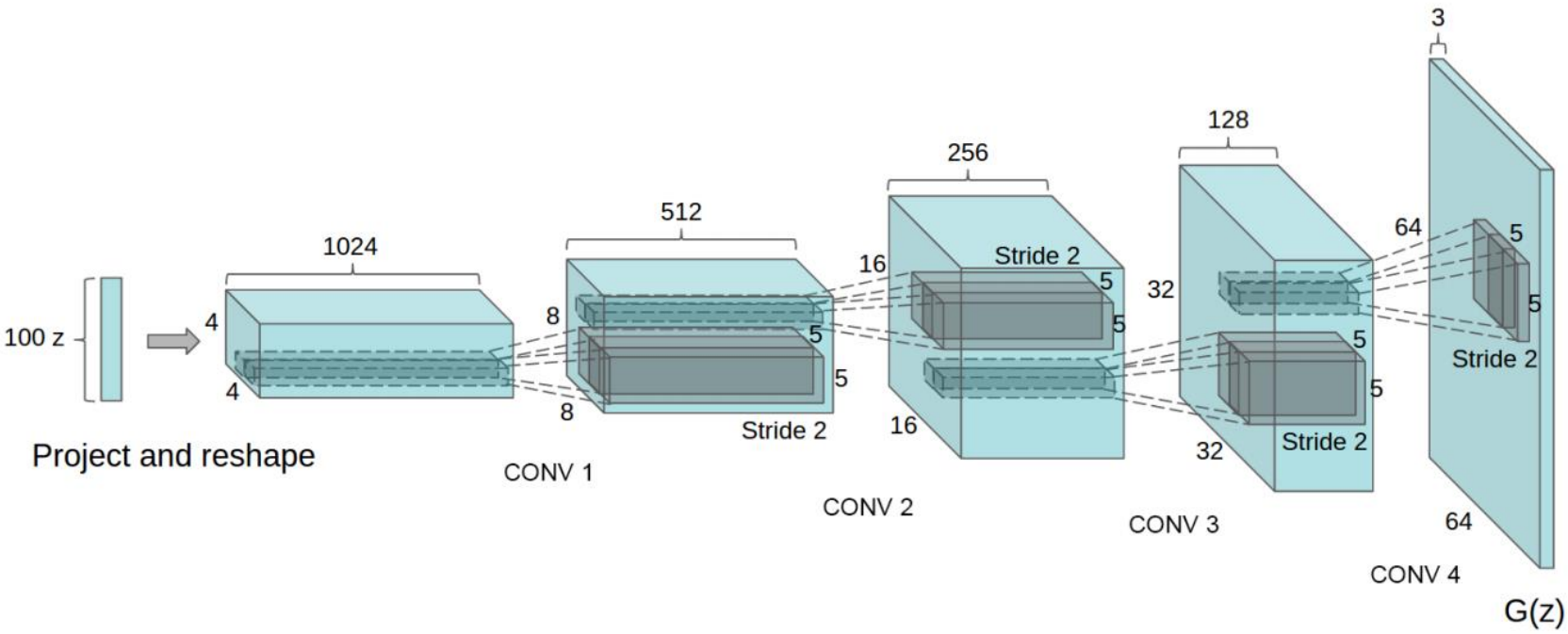
MinMax Game

$$\min_{\theta}\max_{\phi}\left(\mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}\Big[\log D(\mathbf{x},\phi)\Big]+\mathbb{E}_{\mathbf{z}\sim p(\mathbf{z})}\Big[\log(1-D\Big(G(\mathbf{z},\theta),\phi)\Big)\Big]\right)$$



一个具体的模型：DCGANs

- 判别网络是一个传统的深度卷积网络，但使用了带步长的卷积来实现下采样操作，不用最大汇聚（pooling）操作。
- 生成网络使用一个特殊的深度卷积网络来实现使用微步卷积来生成64×63大小的图像。



训练过程

算法 13.1: 生成对抗网络的训练过程

输入: 训练集 $\mathcal{D}$，对抗训练迭代次数 T，每次判别网络的训练迭代次数 K，小批量样本数量 M

随机初始化 θ, ϕ;

for $t \leftarrow 1$ **to** T **do**

  // 训练判别网络 $D(\mathbf{x}, \phi)$

  for $k \leftarrow 1$ **to** K **do**

    // 采集小批量训练样本

    从训练集 $\mathcal{D}$ 中采集 M 个样本 $\{\mathbf{x}^{(m)}\}, 1 \leq m \leq M$;

    从分布 $\mathcal{N}(\mathbf{0}, \mathbf{I})$ 中采集 M 个样本 $\{\mathbf{z}^{(m)}\}, 1 \leq m \leq M$;

    使用随机梯度上升更新 ϕ，梯度为

$$\frac{\partial}{\partial \phi}\left[\frac{1}{M}\sum_{m=1}^{M}\left(\log D(\mathbf{x}^{(m)}, \phi) + \log\left(1 - D\Big(G(\mathbf{z}^{(m)}, \theta), \phi\Big)\right)\right)\right];$$

  end

  // 训练生成网络 $G(\mathbf{z}, \theta)$

  从分布 $\mathcal{N}(\mathbf{0}, \mathbf{I})$ 中采集 M 个样本 $\{\mathbf{z}^{(m)}\}, 1 \leq m \leq M$;

  使用随机梯度上升更新 θ，梯度为

$$\frac{\partial}{\partial \theta}\left[\frac{1}{M}\sum_{m=1}^{M} D\Big(G(\mathbf{z}^{(m)}, \theta), \phi\Big)\right];$$

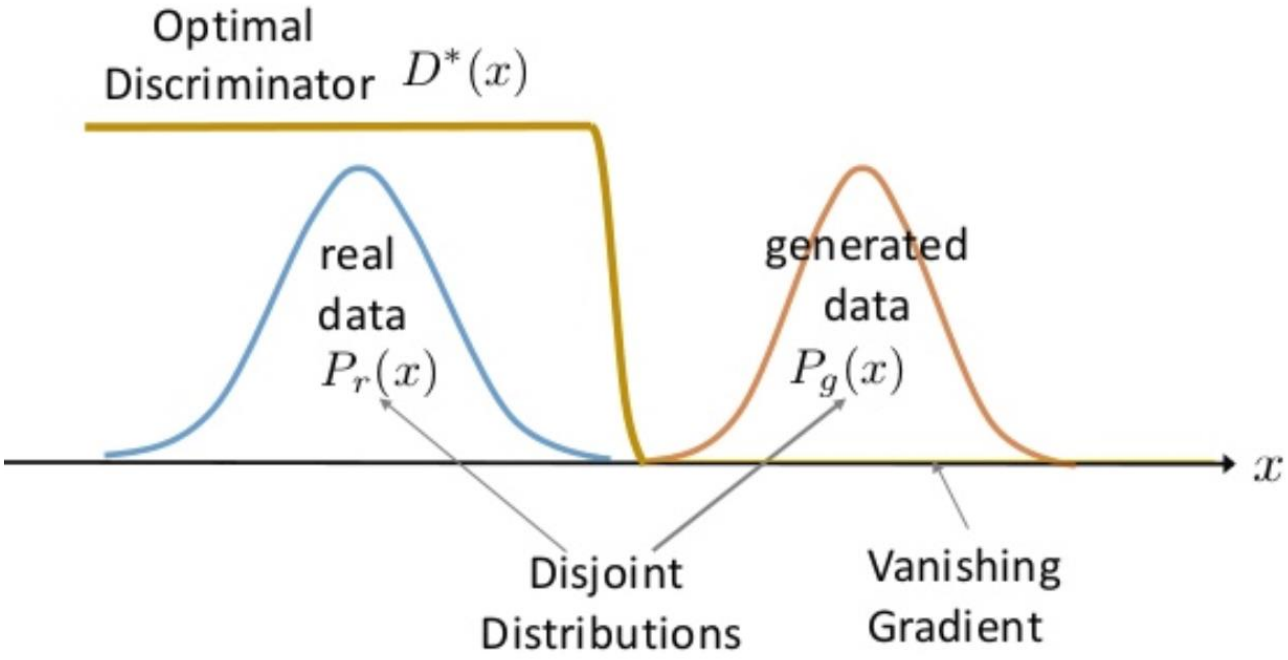
end

输出: 生成网络 $G(\mathbf{z}, \theta)$

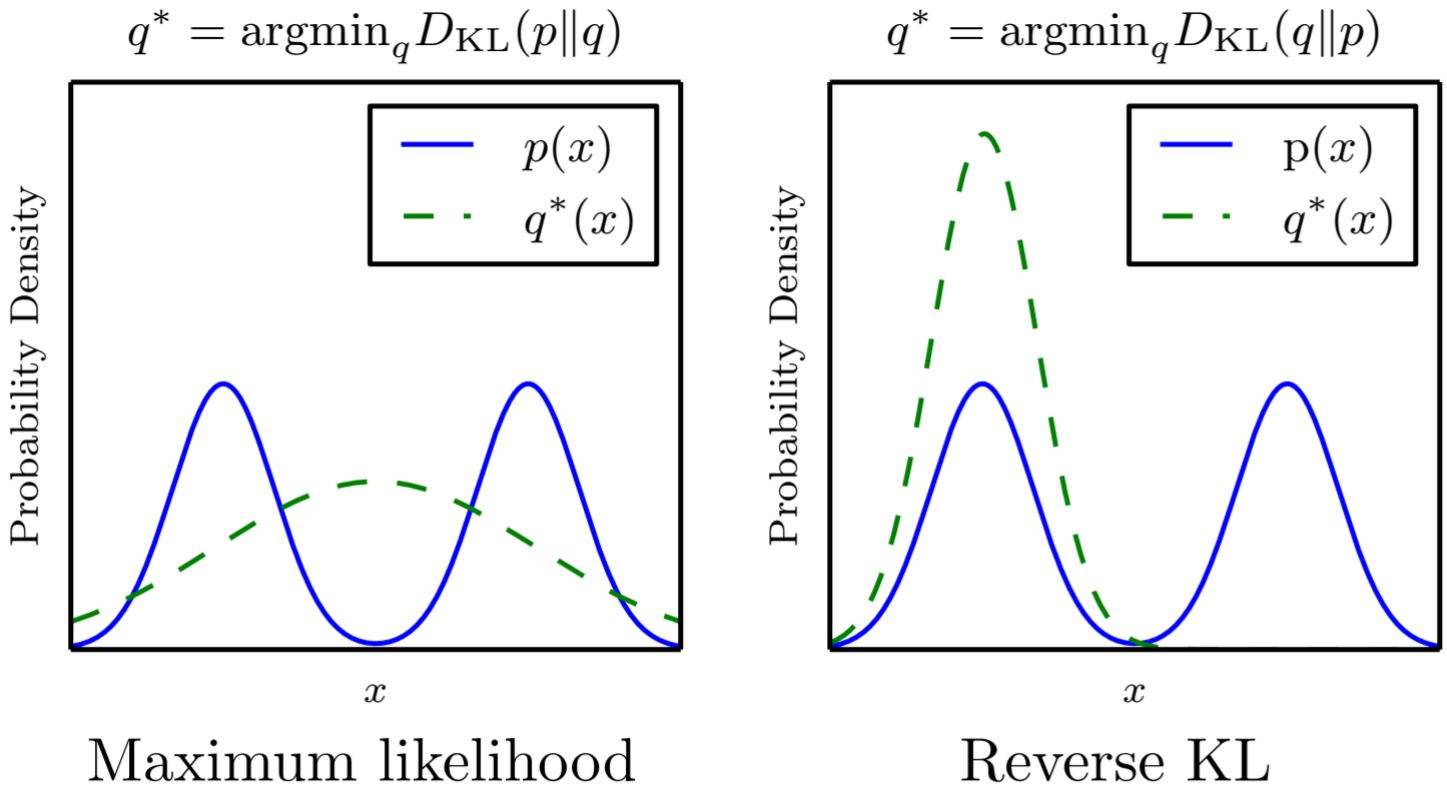
例子

不稳定性

$$\min_{\theta}\max_{\phi}\left(\mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}\Big[\log D(\mathbf{x},\phi)\Big]+\mathbb{E}_{\mathbf{z}\sim p(\mathbf{z})}\Big[\log(1-D\Big(G(\mathbf{z},\theta),\phi)\Big)\Big]\right)$$

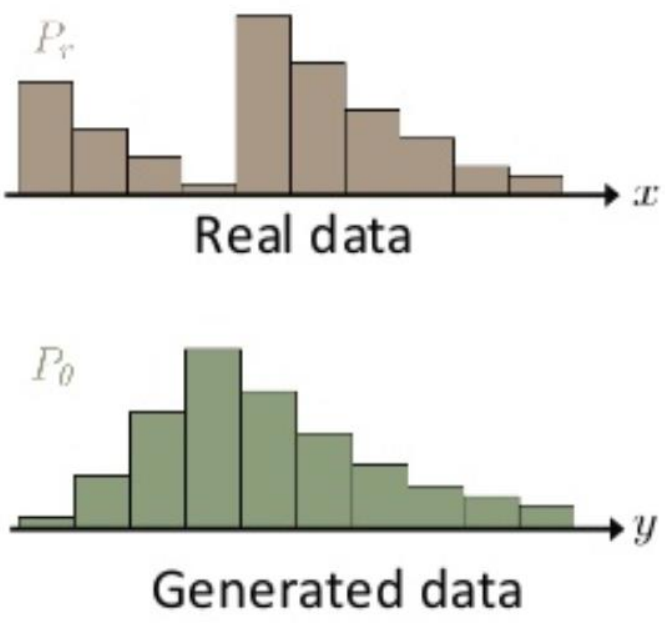


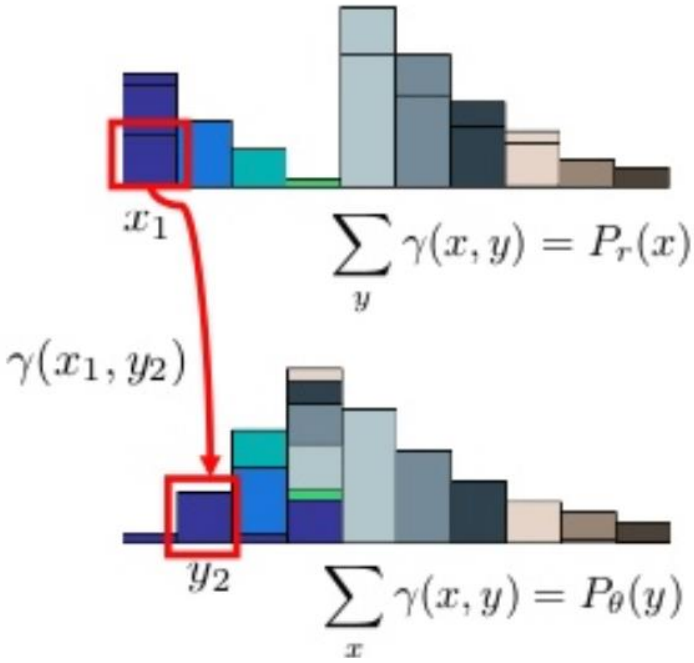
模型坍塌


$q^* = \text{argmin}_q D_{\text{KL}}(p\|q)$
$p(x)$
$q^*(x)$
Probability Density
x
Maximum likelihood
$q^* = \text{argmin}_q D_{\text{KL}}(q\|p)$
p(x)
$q^*(x)$
Probability Density
x
Reverse KL

Earth Mover's Distance

$$\mathrm{EMD}(P_r, P_\theta) = \inf_{\gamma \in \Pi(P_r, P_\theta)} \sum_{x,y} \|x - y\| \gamma(x, y) = \inf_{\gamma \in \Pi(P_r, P_\theta)} \mathbb{E}_{(x,y)\sim\gamma} \|x - y\|$$





Kantorovich-Rubinstein duality

This formula is highly intractable

$$\mathrm{EMD}(P_r, P_\theta) = \inf_{\gamma \in \Pi(P_r, P_\theta)} \sum_{x,y} \|x - y\| \gamma(x, y) = \inf_{\gamma \in \Pi(P_r, P_\theta)} \mathbb{E}_{(x,y)\sim\gamma} \|x - y\|$$

Kantorovich-Rubinstein Duality

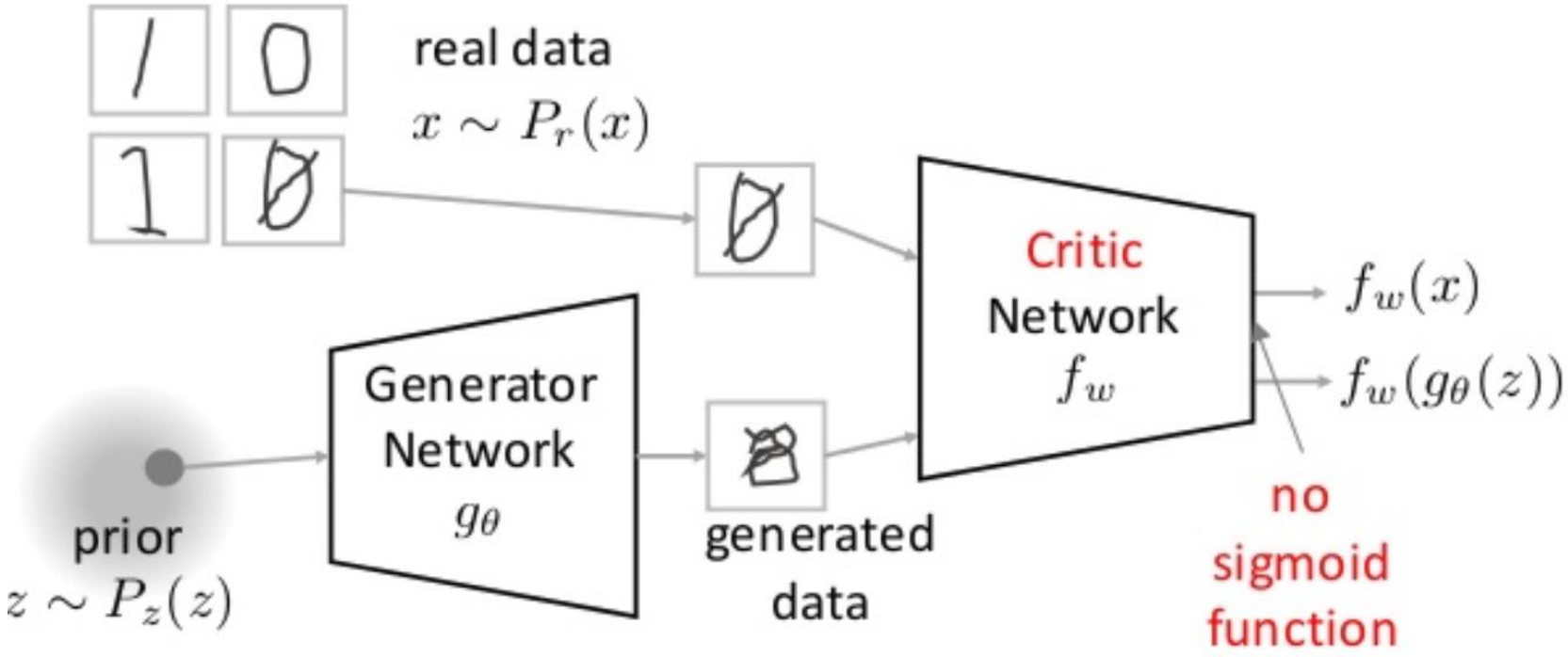
$$\mathrm{EMD}(P_r, P_\theta) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x\sim P_r} f(x) - \mathbb{E}_{x\sim P_\theta} f(x).$$

1-Lipschitz Constraint

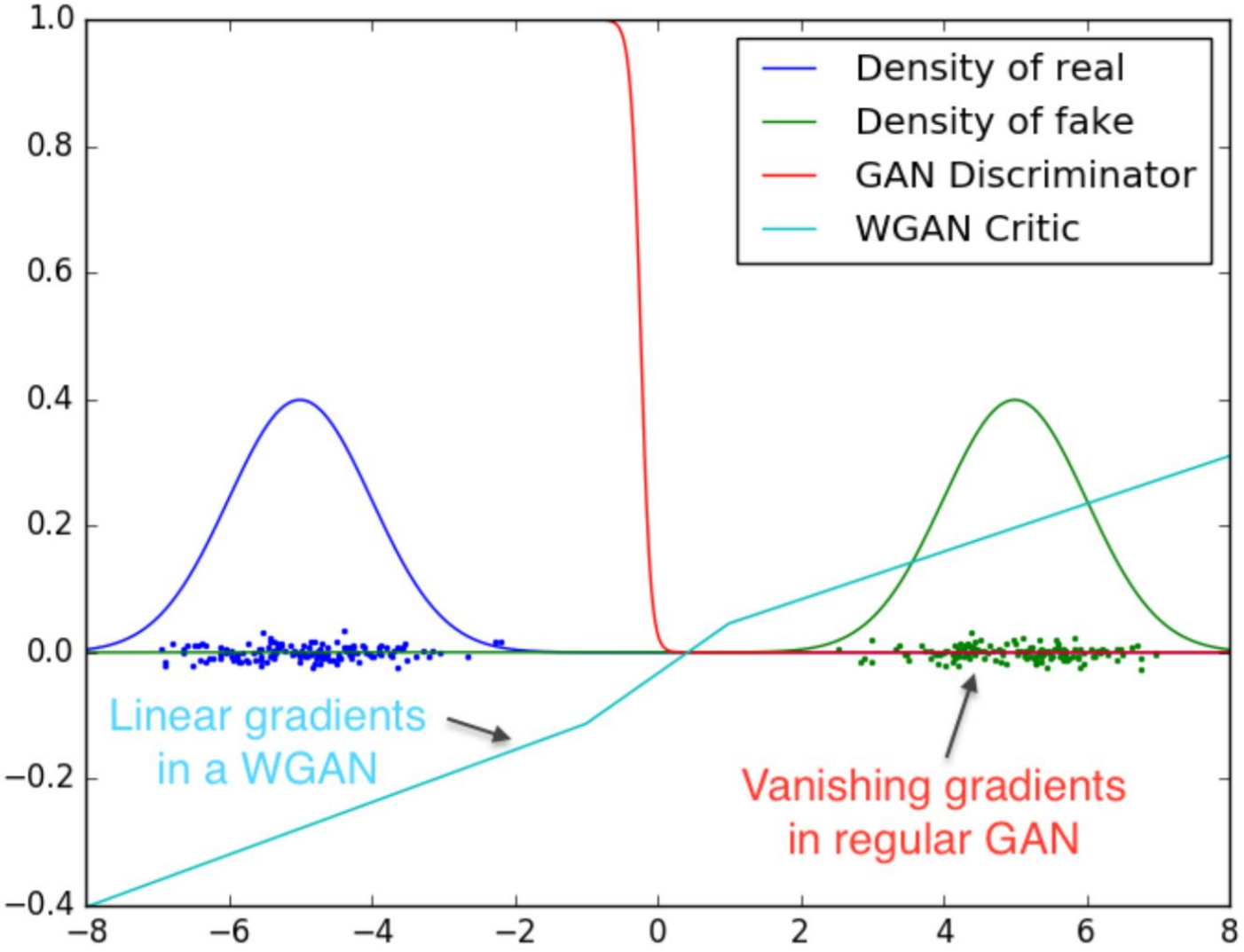
Wasserstein GAN

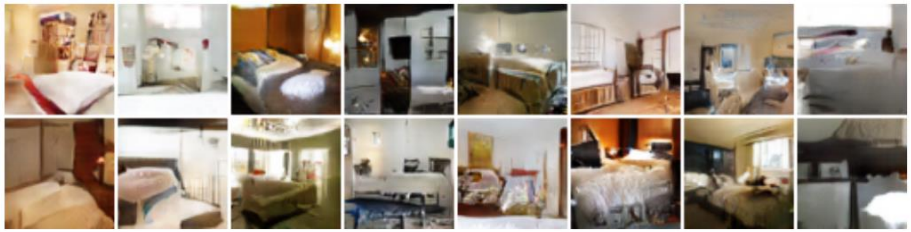
$$\min_{\theta} \max_{w \in [-k,k]^l} \mathbb{E}_{x \sim P_r}[f_w(x)] - \mathbb{E}_{z \sim P_z}[f_w(g_\theta(z))]$$

k-Lipschitz Constraint



梯度问题

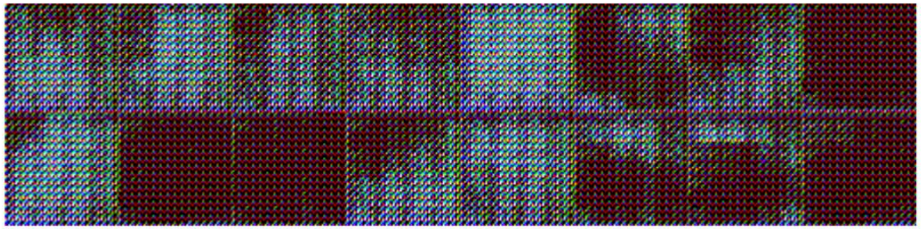

Density of real
Density of fake
GAN Discriminator
WGAN Critic
Linear gradients
in a WGAN
Vanishing gradients
in regular GAN
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−8
−6
−4
−2
0
2
4
6
8



WGAN

DCGAN

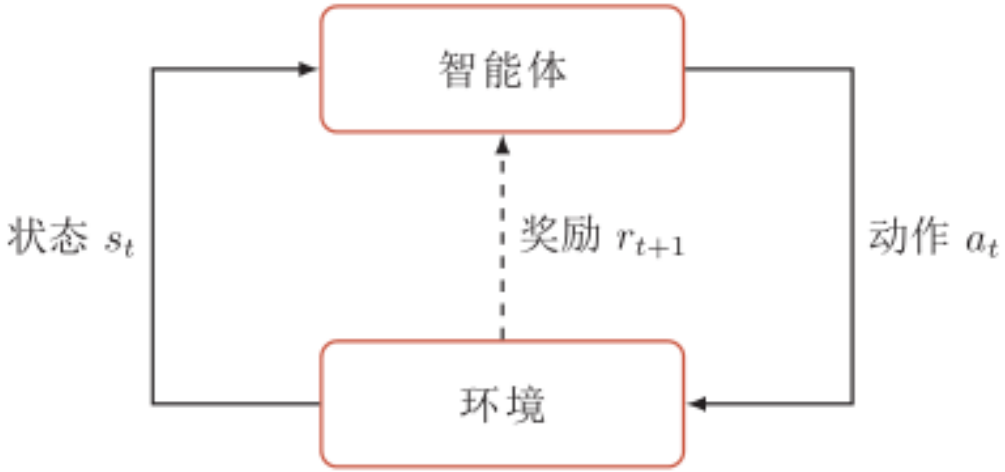
batch normalization
constant number of filters at every layer



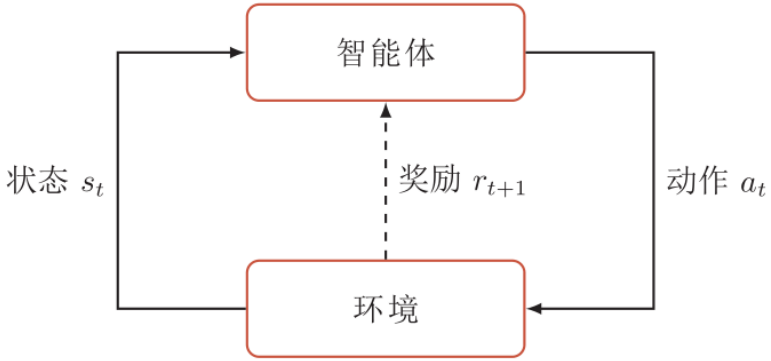
深度强化学习

强化学习

- 强化学习问题可以描述为一个智能体从与环境的交互中不断学习以完成特定目标（比如取得最大奖励值）。
- 强化学习就是智能体不断与环境进行交互，并根据经验调整其策略来最大化其长远的所有奖励的累积值。



马尔可夫决策过程



$$s_0, a_0, s_1, r_1, a_1, \cdots, s_{t-1}, r_{t-1}, a_{t-1}, s_t, r_t, \cdots,$$

- 马尔可夫过程

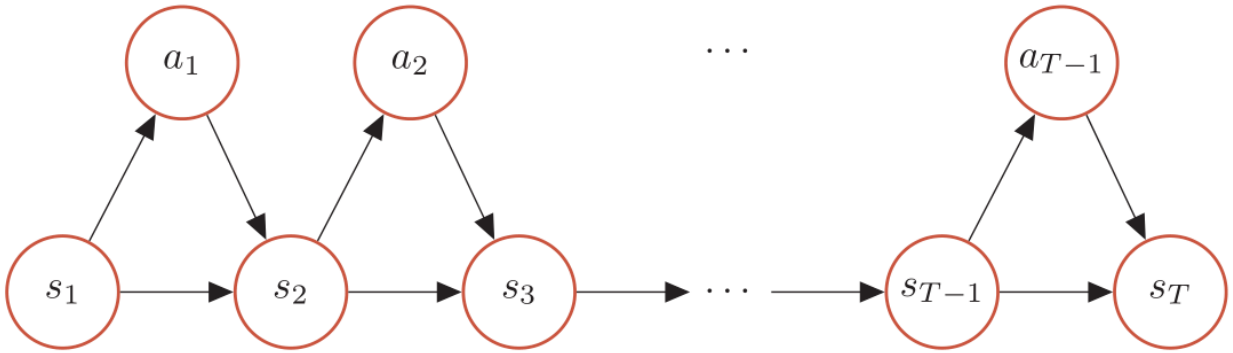
$$p(s_{t+1}|s_t, a_t, \cdots, s_0, a_0) = p(s_{t+1}|s_t, a_t),$$

策略π(a|s)

- 马尔可夫决策过程的一个轨迹（trajectory）

$$\tau = s_0, a_0, s_1, r_1, a_1, \cdots, s_{T-1}, a_{T-1}, s_T, r_T$$

- τ的概率



$$\begin{aligned} p(\tau) &= p(s_0, a_0, s_1, a_1, \cdots), \\ &= p(s_0) \prod_{t=0}^{T-1} \pi(a_t|s_t) p(s_{t+1}|s_t, a_t). \end{aligned}$$

总回报

- 给定策略π(a|s)，智能体和环境一次交互过程的轨迹τ 所收到的累积奖励为总回报（return）

$$G(\tau) = \sum_{t=0}^{T-1} \gamma^t r_{t+1}$$

  - γ ∈ [0,1]是折扣率。当γ接近于0时，智能体更在意短期回报；而当γ接近于1时，长期回报变得更重要。
  - 环境中有一个或多个特殊的终止状态（terminal state）

强化学习目标函数

- 强化学习的目标是学习到一个策略$\pi_\theta(a|s)$来最大化期望回报（expected return）

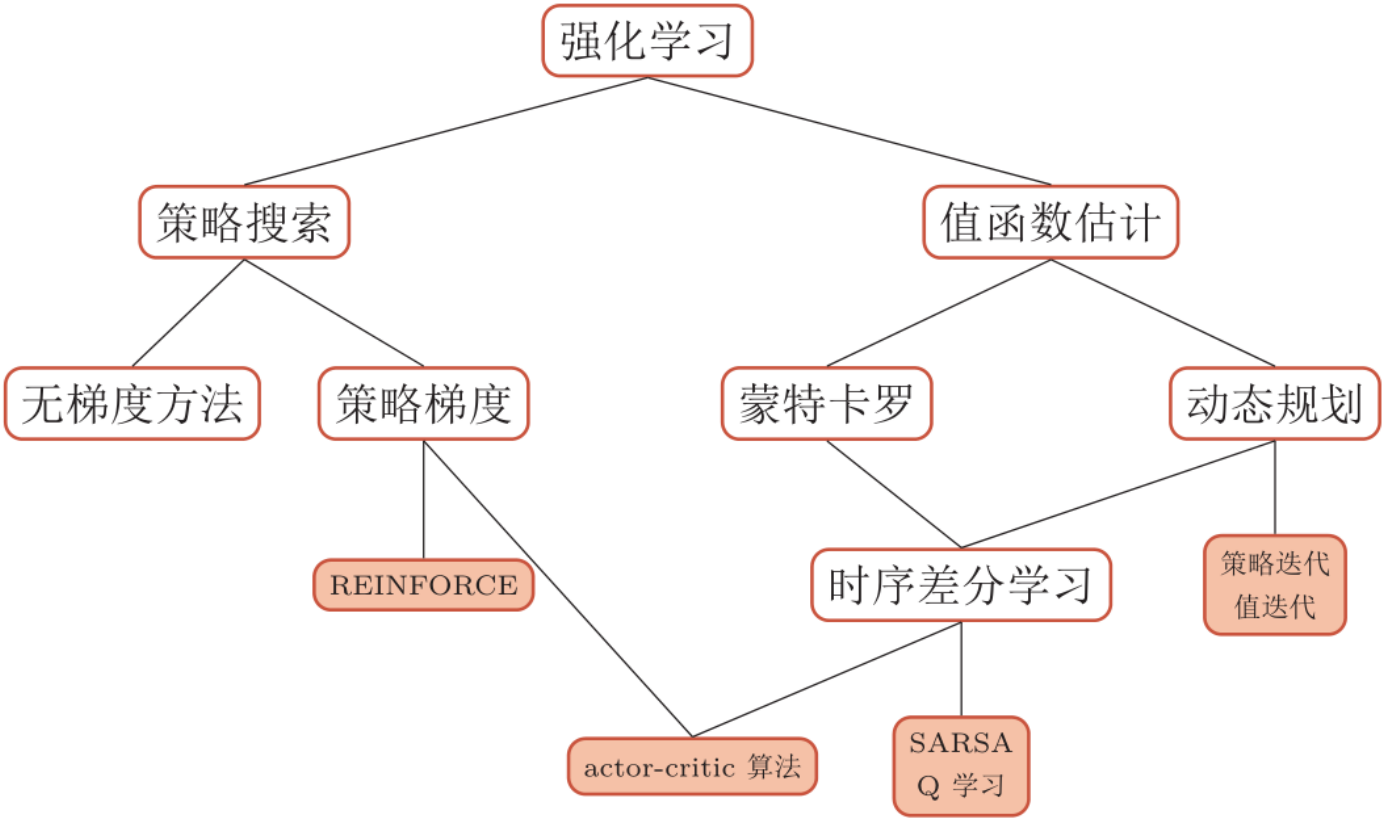
$$J(\theta) = \mathbb{E}_{\tau \sim p_\theta(\tau)}[G(\tau)] = \mathbb{E}_{\tau \sim p_\theta(\tau)}[\sum_{t=0}^{T-1} \gamma^t r_{t+1}]$$

  - θ为策略函数的参数

深度强化学习

- 深度强化学习是将强化学习和深度学习结合在一起，用强化学习来定义问题和优化目标，用深度学习来解决状态表示、策略表示等问题。
  - 两种不同的结合强化学习和深度学习的方式，分别用深度神经网络来建模强化学习中的值函数、策略，然后用误差反向传播算法来优化目标函数。

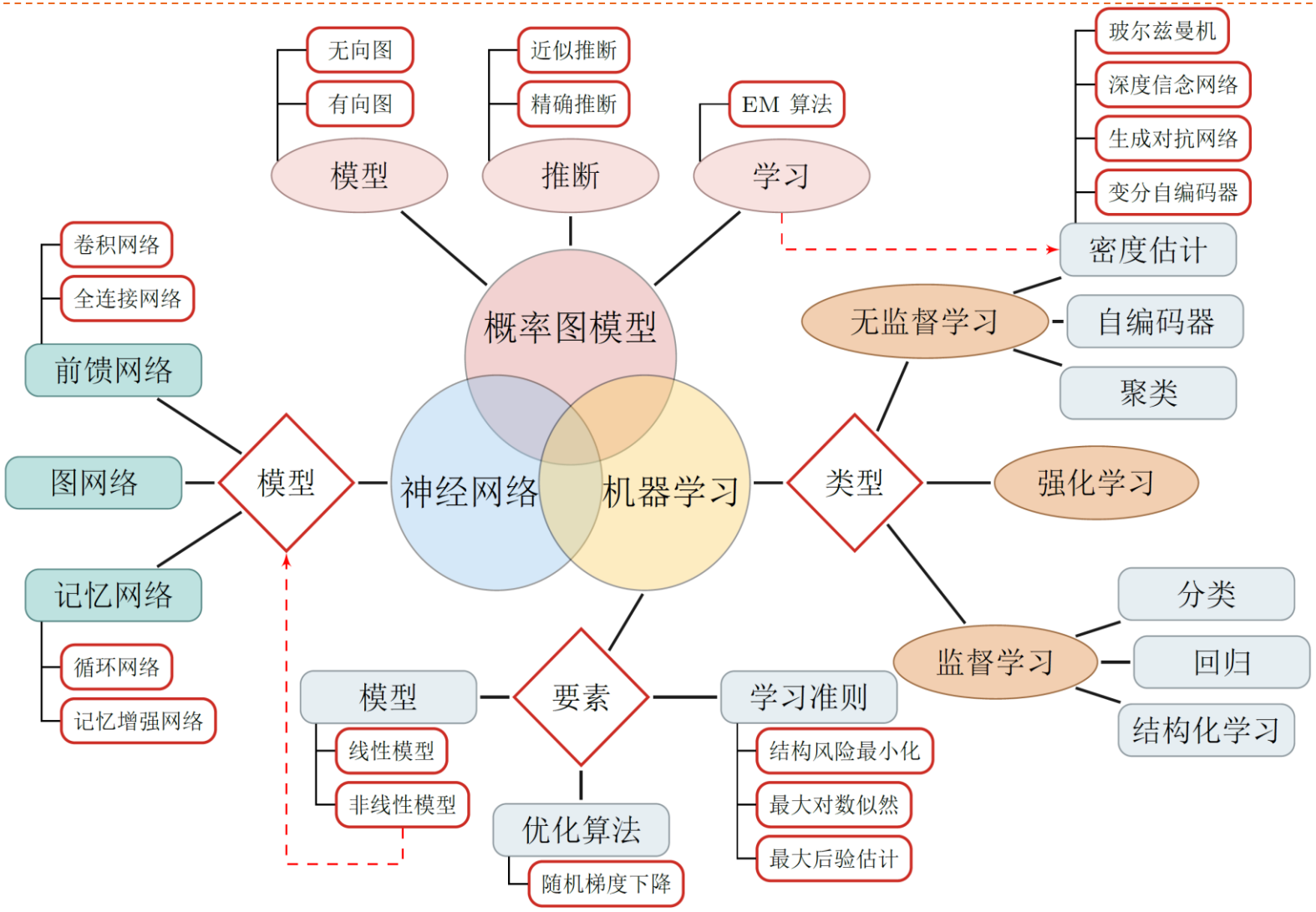
不同强化学习算法之间的关系



汇总

算法	步骤
SARSA	（1）执行策略，生成样本：s, a, r, s', a' （2）估计回报：$Q(s,a) \leftarrow Q(s,a) + \alpha\Big(r + \gamma Q(s',a') - Q(s,a)\Big)$ （3）更新策略：$\pi(s) = \arg\max_{a \in \lvert\mathcal{A}\rvert} Q(s,a)$
Q学习	（1）执行策略，生成样本：s, a, r, s' （2）估计回报：$Q(s,a) \leftarrow Q(s,a) + \alpha\Big(r + \gamma \max_{a'} Q(s',a') - Q(s,a)\Big)$ （3）更新策略：$\pi(s) = \arg\max_{a \in \lvert\mathcal{A}\rvert} Q(s,a)$
REINFORCE	（1）执行策略，生成样本：$\tau = s_0, a_0, s_1, a_1, \cdots$ （2）估计回报：$G(\tau) = \sum_{t=0}^{T-1} r_{t+1}$ （3）更新策略：$\theta \leftarrow \theta + \sum_{t=0}^{T-1} \Big(\frac{\partial}{\partial\theta} \log \pi_\theta(a_t \lvert s_t) \gamma^t G(\tau_{t:T})\Big)$
Actor-Critic	（1）执行策略，生成样本：s, a, s', r （2）估计回报：$G(s) = r + \gamma V_\phi(s')$ $\phi \leftarrow \phi + \beta\Big(G(s) - V_\phi(s)\Big) \frac{\partial}{\partial\phi} V_\phi(s)$ （3）更新策略：$\lambda \leftarrow \gamma\lambda$ $\theta \leftarrow \theta + \alpha\lambda\Big(G(s) - V_\phi(s)\Big) \frac{\partial}{\partial\theta} \log \pi_\theta(a \lvert s)$

汇总


概率图模型
神经网络
机器学习
模型
推断
学习
无向图
有向图
近似推断
精确推断
EM 算法
卷积网络
全连接网络
前馈网络
图网络
记忆网络
循环网络
记忆增强网络
模型
类型
要素
无监督学习
密度估计
自编码器
聚类
玻尔兹曼机
深度信念网络
生成对抗网络
变分自编码器
强化学习
监督学习
分类
回归
结构化学习
模型
线性模型
非线性模型
优化算法
随机梯度下降
学习准则
结构风险最小化
最大对数似然
最大后验估计

推荐课程

- 斯坦福大学CS224d: Deep Learning for Natural Language Processing
  - http://cs224d.stanford.edu/
  - Richard Socher 主要讲解自然语言处理领域的各种深度学习模型
- 斯坦福大学CS231n: Convolutional Neural Networks for Visual Recognition
  - http://cs231n.stanford.edu/
  - Fei-Fei Li Andrej Karpathy 主要讲解CNN、RNN在图像领域的应用
- 加州大学伯克利分校 CS 294: Deep Reinforcement Learning
  - http://rail.eecs.berkeley.edu/deeprlcourse/

推荐材料

- 林轩田《机器学习基石》《机器学习技法》
  - https://www.csie.ntu.edu.tw/~htlin/mooc/
- 李宏毅《1天搞懂深度学习》
  - http://speech.ee.ntu.edu.tw/~tlkagk/slide/Tutorial_HYLee_Deep.pptx
- 李宏毅《Generative Adversarial Network (GAN)》
  - http://speech.ee.ntu.edu.tw/~tlkagk/slide/Tutorial_HYLee_GAN.pptx

谢 谢

https://nndl.github.io/