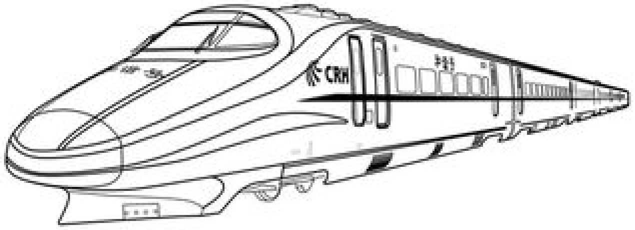
北京交通大学
BEIJING JIAOTONG UNIVERSITY

智能信息技术教育中心
The Education Center of Intelligence Information Technologies

人工智能基础

深度学习

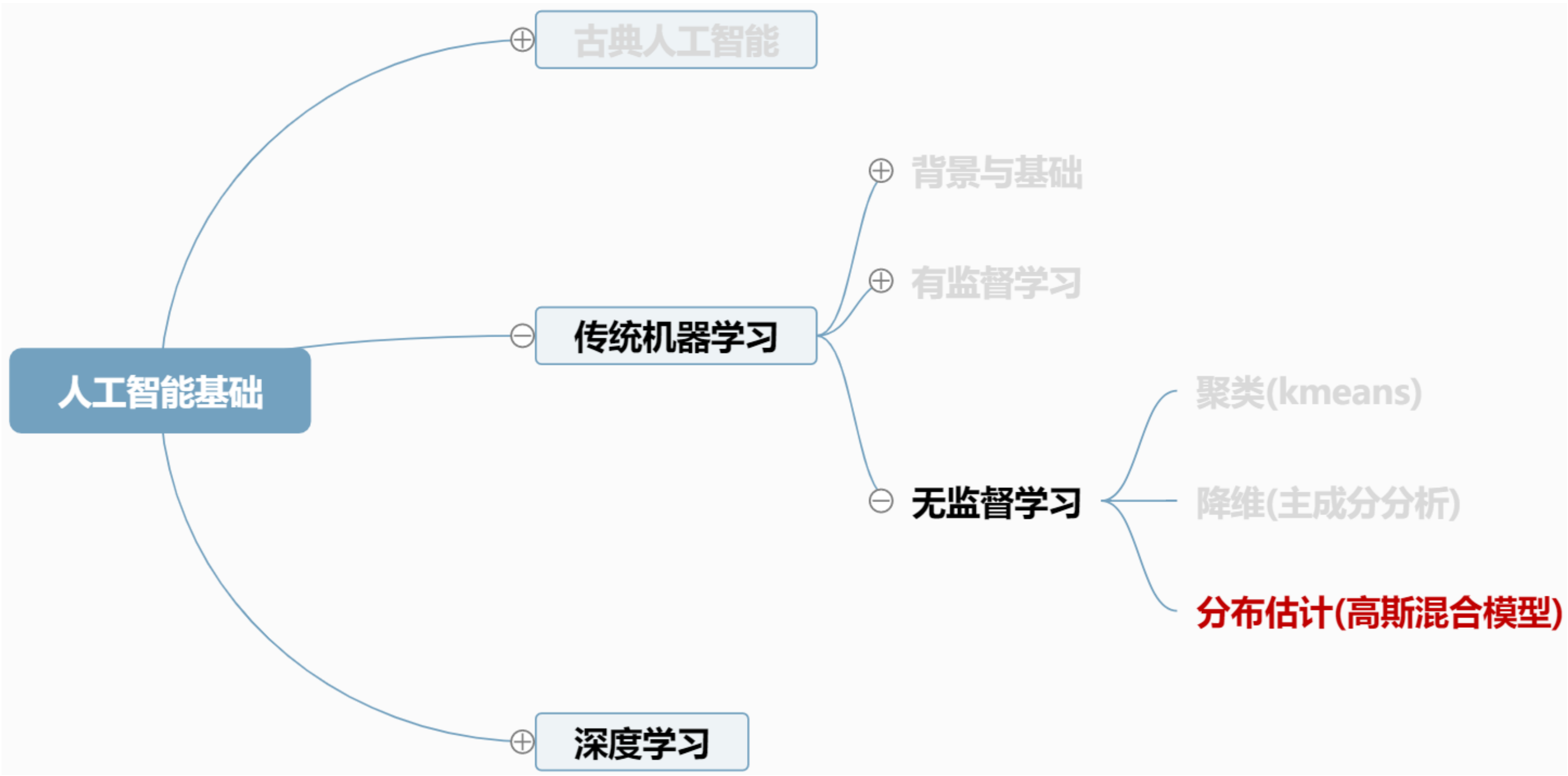
耿阳李敖
2022年3月



内容概要

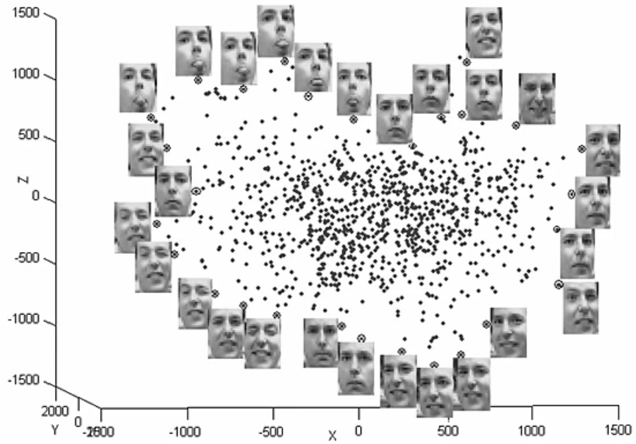
- **内容回顾**
- 神经网络基础
- 全连接神经网络
- 总结

内容回顾


古典人工智能
背景与基础
有监督学习
传统机器学习
人工智能基础
聚类(kmeans)
无监督学习
降维(主成分分析)
分布估计(高斯混合模型)
深度学习

回顾：分布估计

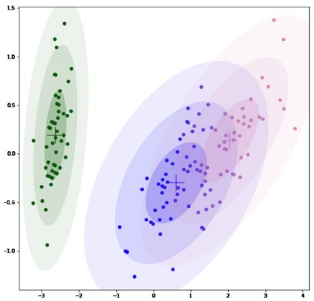
生成式学习

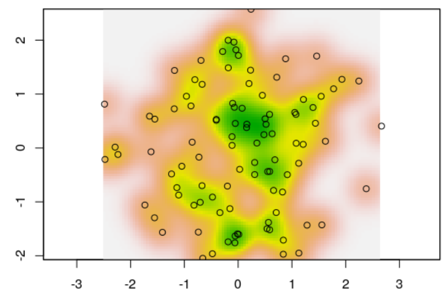


- 基于给定数据集估计数据分布
- 利用估计得到的分布采样生成新样本

分布估计分类

- 带参估计：利用包含参数的假设模型去建模
- 非参估计：不假设模型，直接基于数据本身去估计



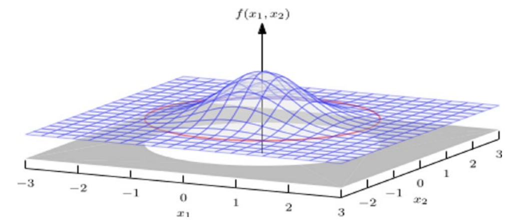


回顾：分布估计

混合高斯模型（GMM）

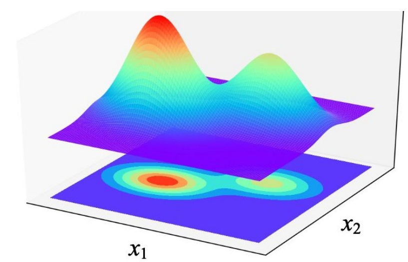
- 单个高斯

$$\mathcal{N}(X=\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Sigma})=\frac{1}{(2\pi)^{D/2}\sqrt{|\boldsymbol{\Sigma}|}}\exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$$



- 混合高斯

$$P(X=\mathbf{x}|\Theta)=\sum_{k=1}^{K}\phi_k\mathcal{N}(X=\mathbf{x}|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)$$



EM算法

- 基于 $\mathcal{X}$ 和 $\widehat{\Theta}$ 估计条件期望（E 步）

$$\mathbb{E}_{P(Z|X,\widehat{\Theta})}\ln P(Z,X|\Theta)$$

- 求解条件期望最大化（M 步）

$$\max_{\Theta}\mathbb{E}_{P(Z|X,\widehat{\Theta})}\ln P(Z,X|\Theta)$$

回顾：分布估计

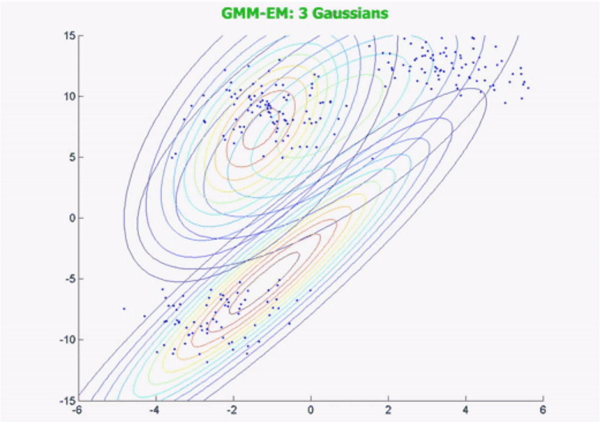
EM算法估计GMM参数

- E步

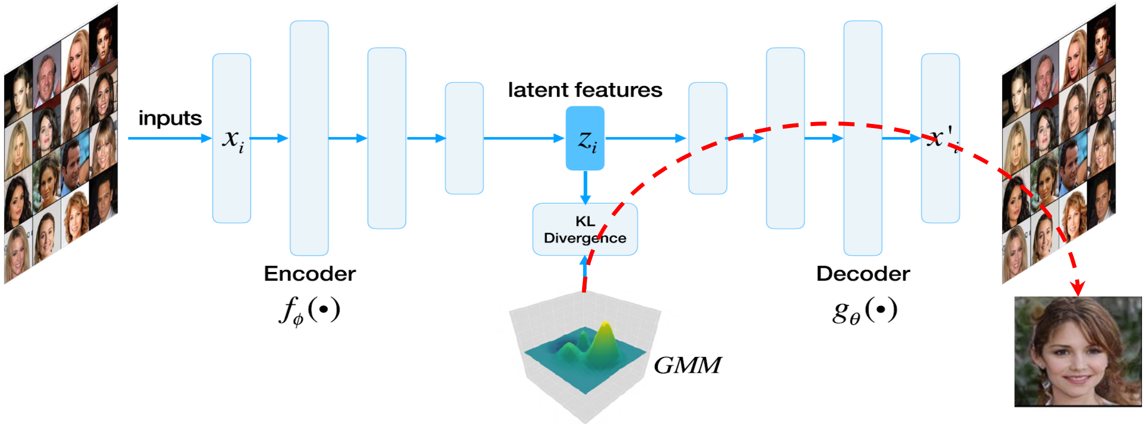
$$\gamma_{ik} \leftarrow \frac{\hat{\phi}_k \mathcal{N}(X = \mathbf{x}_i | \hat{\boldsymbol{\mu}}_k, \hat{\boldsymbol{\Sigma}}_k)}{\sum_{k=1}^{K} \hat{\phi}_k \mathcal{N}(X = \mathbf{x}_i | \hat{\boldsymbol{\mu}}_k, \hat{\boldsymbol{\Sigma}}_k)}$$

- M步

$$\hat{\boldsymbol{\mu}}_k \leftarrow \frac{\sum_{i=1}^{N} \gamma_{ik} \mathbf{x}_i}{\sum_{i=1}^{N} \gamma_{ik}} \quad \hat{\boldsymbol{\Sigma}}_k \leftarrow \frac{\sum_{i=1}^{N} \gamma_{ik} (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_k)(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_k)^T}{\sum_{i=1}^{N} \gamma_{ik}} \quad \hat{\phi}_k \leftarrow \frac{\sum_{i=1}^{N} \gamma_{ik}}{\sum_{i=1}^{N} \sum_{k=1}^{K} \gamma_{ik}}$$



GMM应用

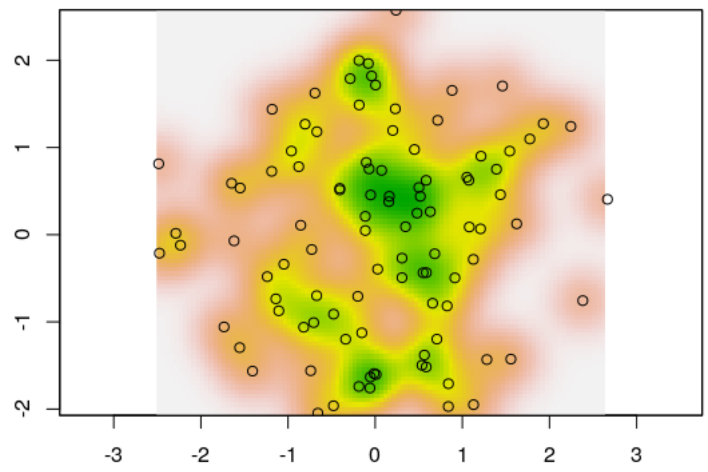


回顾：分布估计

非参估计：核密度估计

- 以每个样本为中心构造分布密度

$$f(\mathbf{x}) = \frac{1}{Nh}\sum_{i=1}^{N} Ker\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right)$$



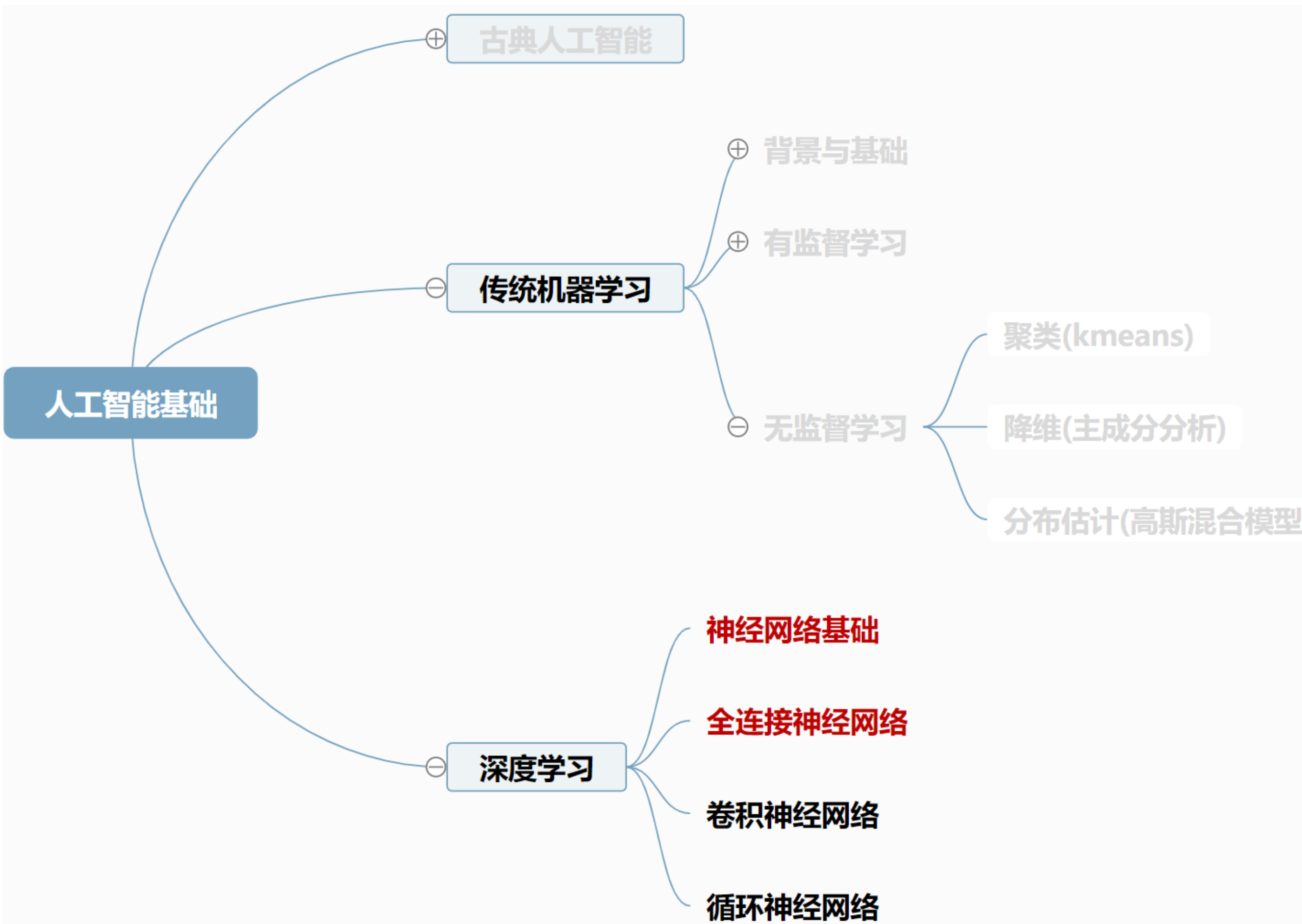
高斯核密度估计

$$Ker(\mathbf{x}) = \frac{1}{(2\pi)^{D/2}} \exp\left\{-\frac{1}{2}\mathbf{x}^T\mathbf{x}\right\}$$

$$f(\mathbf{x}) = \frac{1}{Nh}\sum_{i=1}^{N} Ker\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{(2\pi)^{D/2}h}\exp\left\{-\frac{1}{2}\frac{(\mathbf{x}-\mathbf{x}_i)^T(\mathbf{x}-\mathbf{x}_i)}{h^2}\right\}$$

- 可以看做混合高斯的一种特殊情况

本节概况



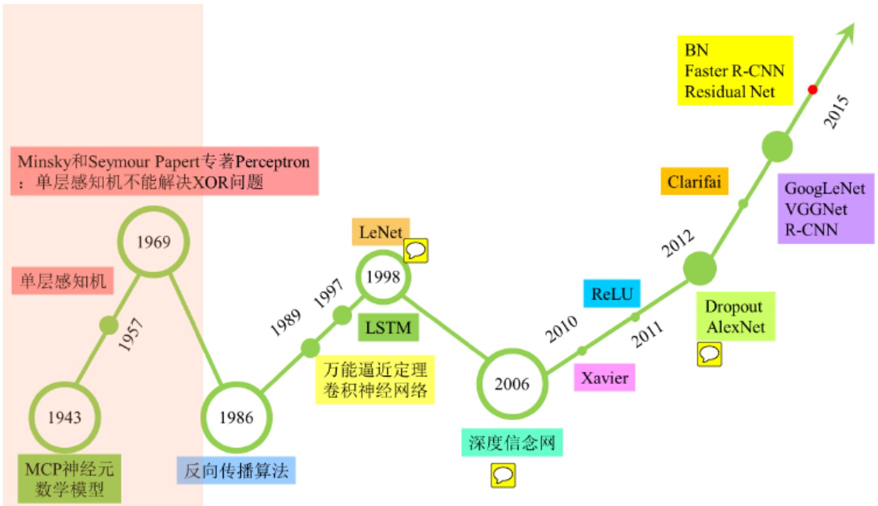
古典人工智能
背景与基础
有监督学习
传统机器学习
聚类(kmeans)
人工智能基础
无监督学习
降维(主成分分析)
分布估计(高斯混合模型)
神经网络基础
全连接神经网络
深度学习
卷积神经网络
循环神经网络

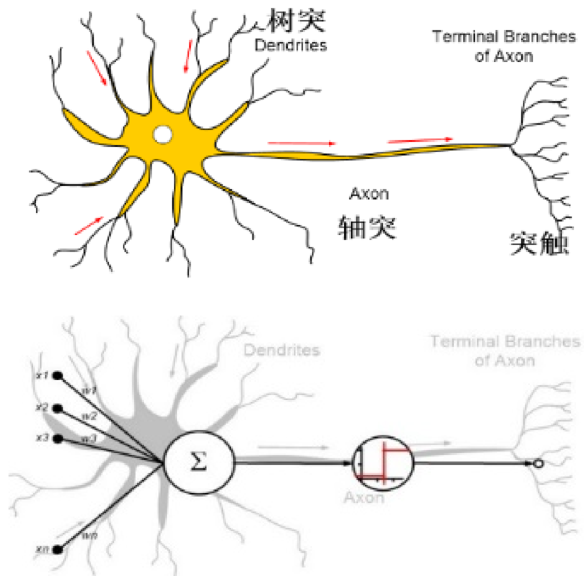
内容概要

- 内容回顾
- **神经网络基础**
- 全连接神经网络
- 总结

神经网络基础

历史回顾

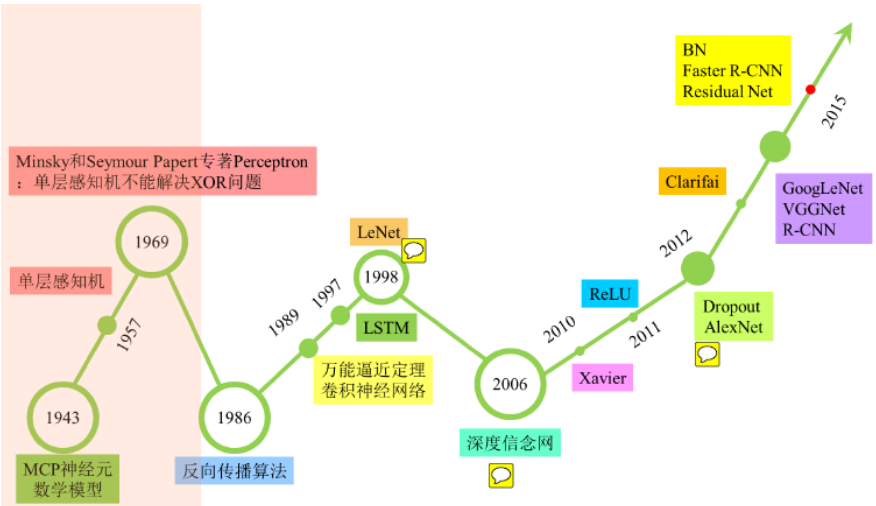


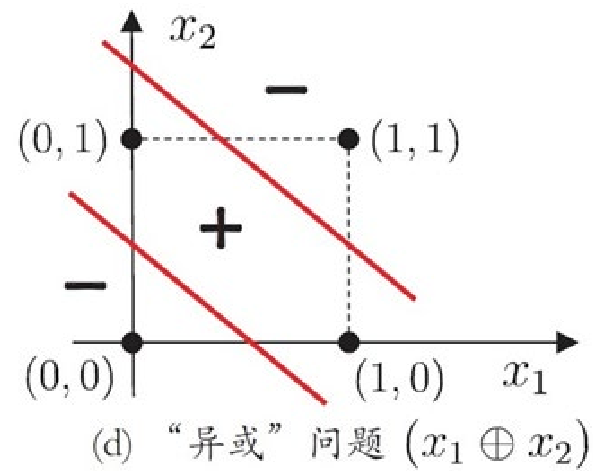


- 第一阶段
  - 1943年, McCulloch和Pitts 提出第一个神经元数学模型，即MCP模型
  - 1949年, Hebb 提出生物神经元学 习的机理，即Hebb学习规则
  - 1957年, Rosenblatt 提出感知机网络（Perceptron）模型和其学习规则（梯度下降）

神经网络基础

历史回顾



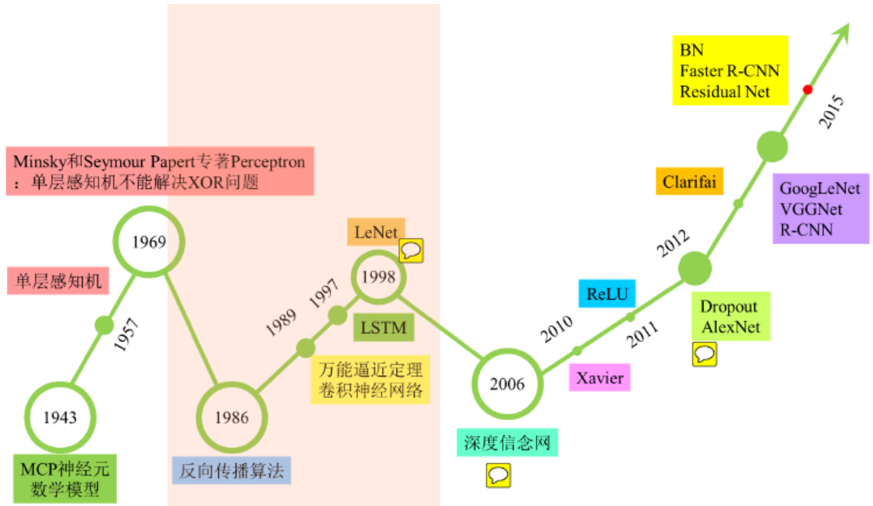


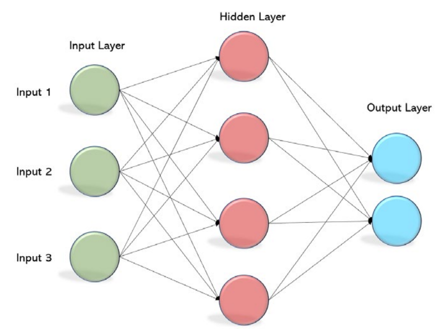
- 第一阶段

  - 1969年，Minsky和Papert 发表《Perceptrons》一书，指出单层神经网路不能解 决非线性问题

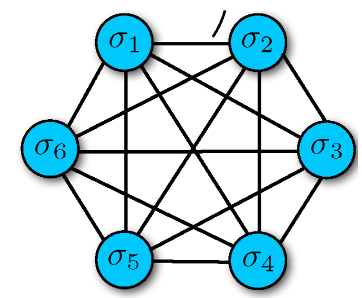
神经网络基础

历史回顾





多层感知机

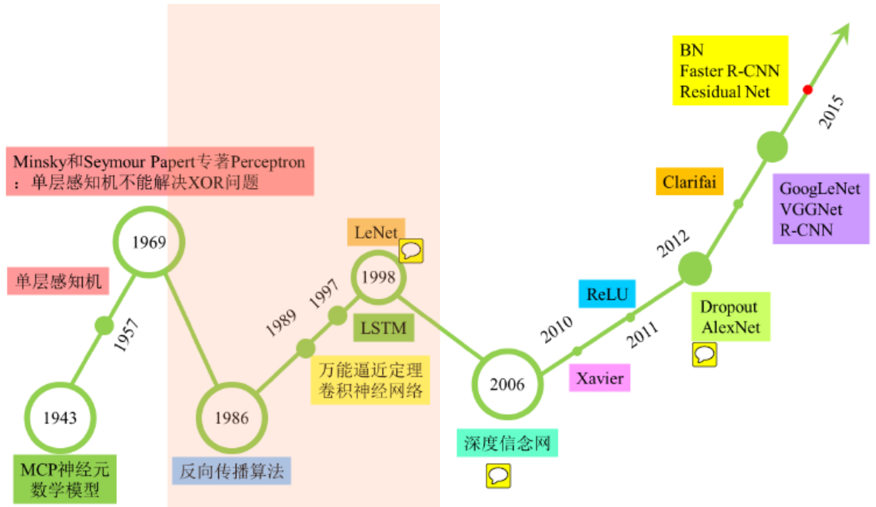


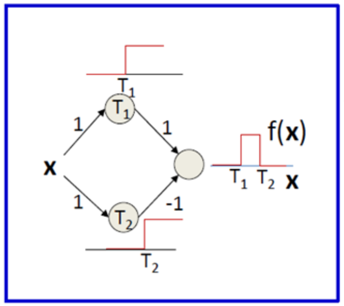
Hopfield Network

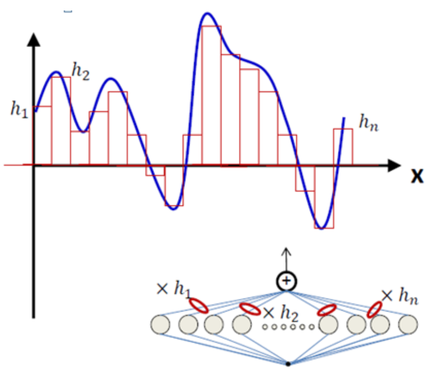
- 第二阶段
  - 1980年代，多层感知机开发出来，已经具有当前深度神经网络的雏形
  - 1982年，Hopfield提出了一种具有联想记忆、优化计算能力的递归网络模型，即 Hopfield 网络
  - 1986年, LeCun反向传播算法（BP），但当隐藏层≥3时效果不太好

神经网络基础

历史回顾





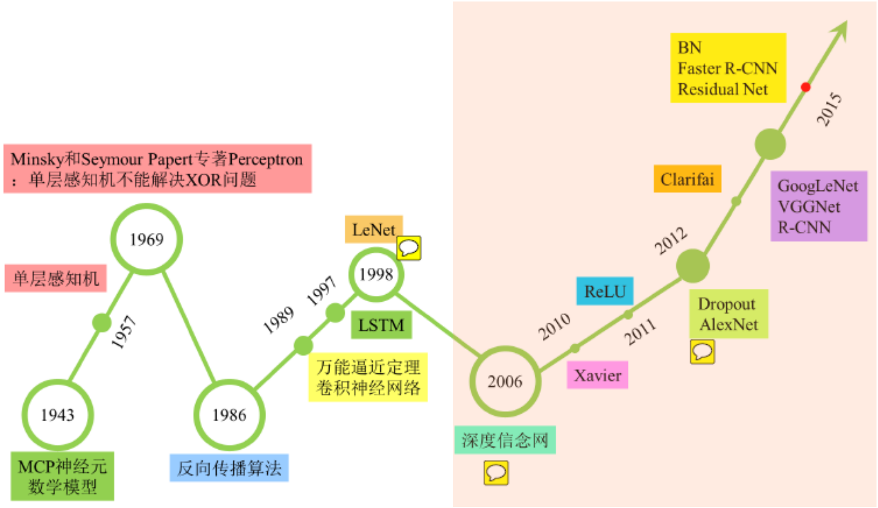


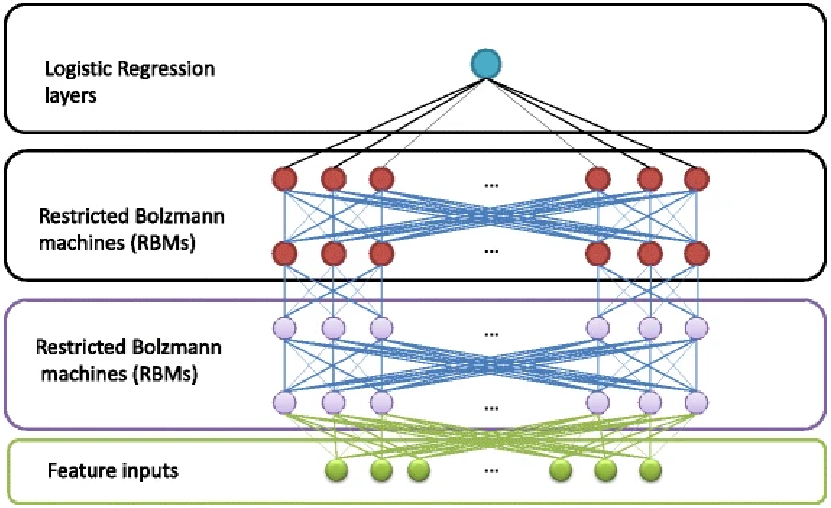
- 第二阶段

  - 1989年，足够宽度的单隐层神经网络被证明能以任意精度逼近任意函数
  - 90年代初, 伴随统计学习理论和SVM的兴起, 神经网络由于理论不够清楚, 试错性强, 难以训练, 再次进入低谷

神经网络基础

历史回顾





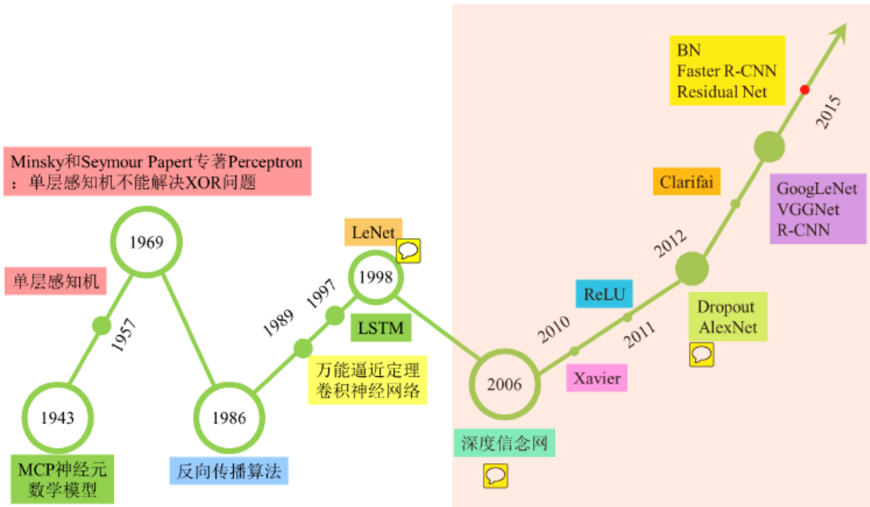
- 第三阶段

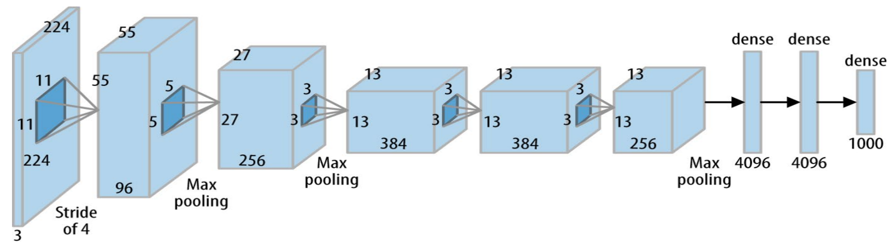
  - 2006年, Hinton提出了深度信念网络(DBN), 通过“预训练+微调”使得深度模 型的最优化变得相对容易

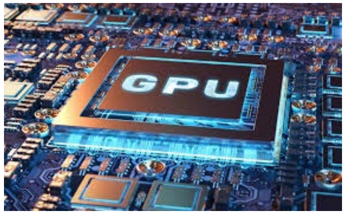
  - 后续，ReLu激活函数的提出解决了梯度消失与爆炸问题，使得可训练神经网络的层数迅速增加

神经网络基础

历史回顾







- 第三阶段

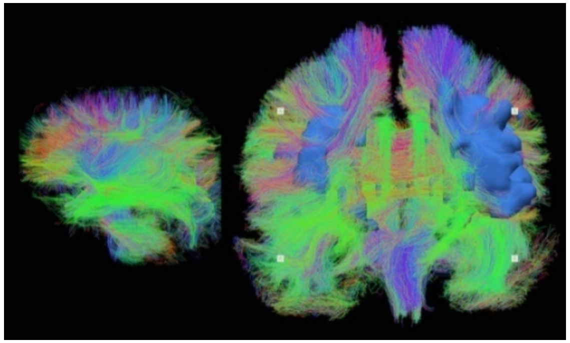
  - 2011年，GPU用于加速神经网络学习的技术逐步成熟

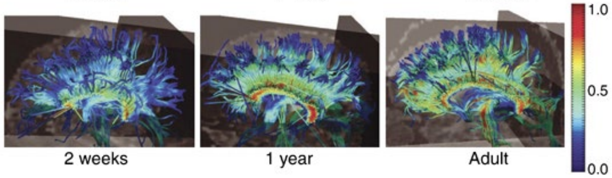
  - 2012年, Hinton 组参加ImageNet 竞赛, 使用AlexNet（一种卷积神经网络）模型以超过第二名10个百 分点的成绩夺得当年竞赛的冠军

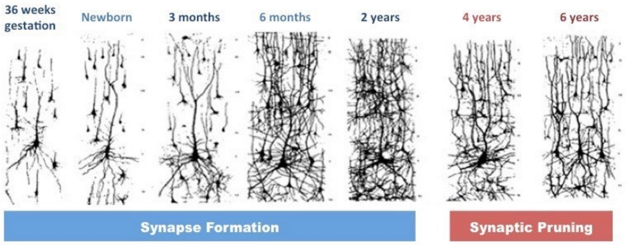
神经网络基础

人类神经网络

- 人类大脑由神经元、神经胶质细胞、神经干细胞和血管组成
- 神经元(neuron)，也叫神经细胞(nerve cell)，是人脑神经系统中最基本的单元
- 人脑神经系统包含近860亿个神经元
- 每个神经元有上千个突触与其他神经元相连人脑神经元连接成巨大的复杂网络，总长度可达数千公里

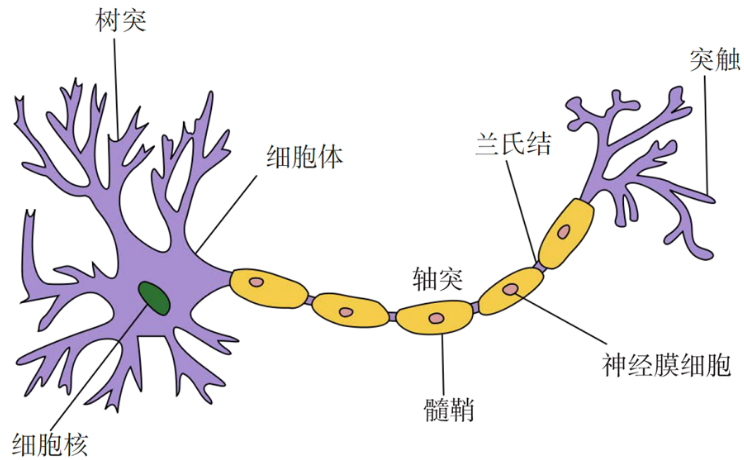






神经网络基础

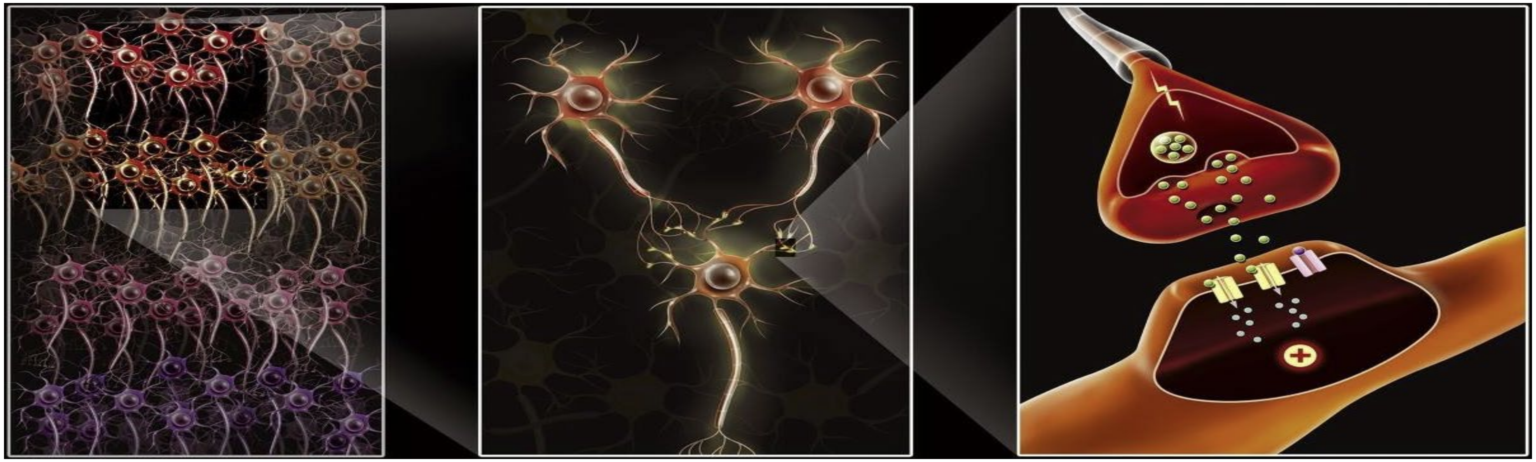
信息传递原理



- **细胞体**：通过生物化学反应，引起细胞膜内外电位差发生改变，形成兴奋或抑制状态
- **细胞突起**由细胞体延伸出来，又可分为树突和轴突：
  - **树突**：可接收刺激并将兴奋传入细胞体，每个神经元可以有一个或多个树突
  - **轴突**：可把自身兴奋状态从细胞体传给另一个神经元，每个神经元只有一个轴突

神经网络基础

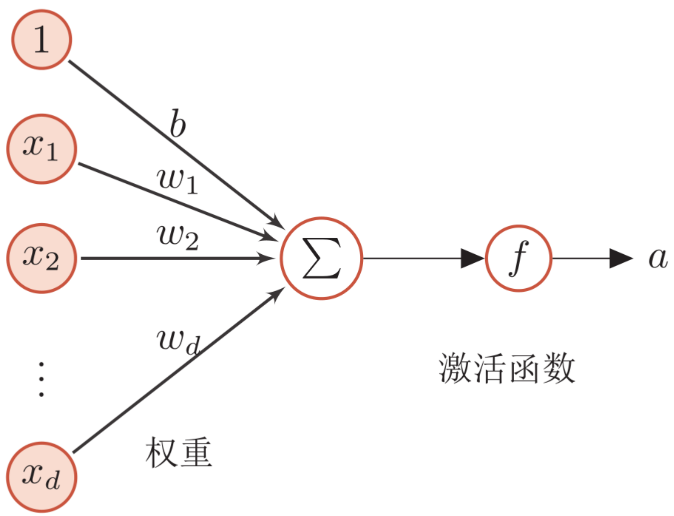
人类神经网络



- 每个神经元与其他神经元相连，当它“兴奋”时，就会向相连的神经元发送化学物质，从而改变这些神经元内的电位；
- 如果神经元的电位超过一定“阈值”，它就会被激活，即“兴奋”起来，然后向其他神经元发送化学物质。

MCP神经元模型（McCulloch and Pitts，1943）

- 神经元接收到来自其他 d 个神经元传递过来的输入信号，这些输入信号通过带权重的连接进行传递，神经元接收到的总输入值将与神经元的阈值（bias）进行比较，然后通过“激活函数”处理产生神经元的输出。



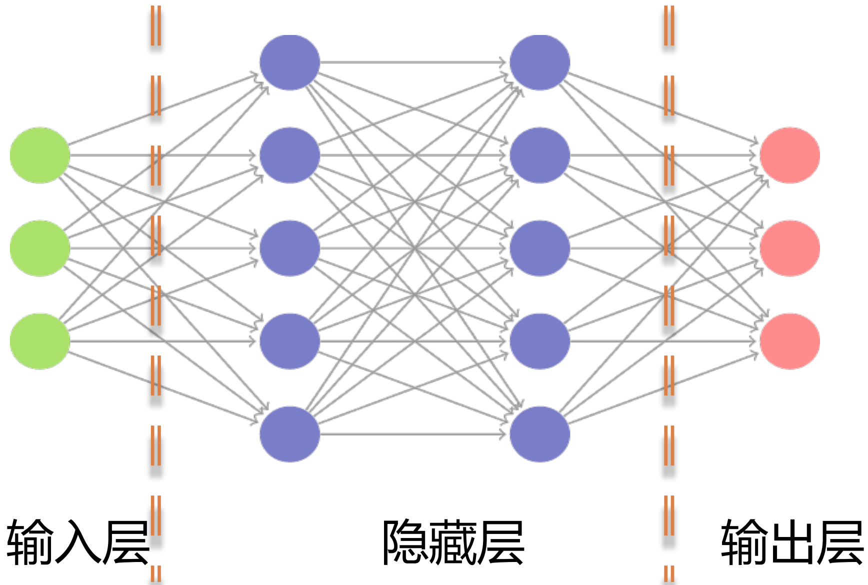
神经网络基础

人工神经网络

- 把许多人工神经元按一定的层次结构连接起来，就形成人工神经网络

- 人工神经网络结构三大要素：

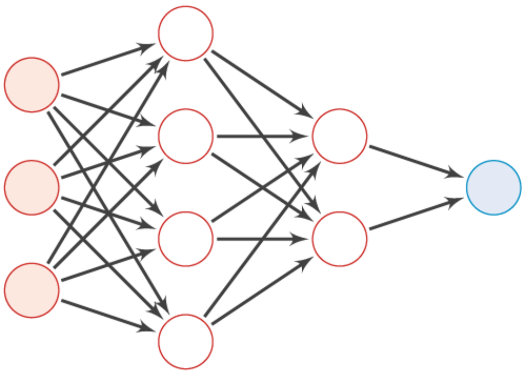
  - **节点设计（激活函数）**
  - **连接结构**
  - **信息传递方向**



神经网络基础

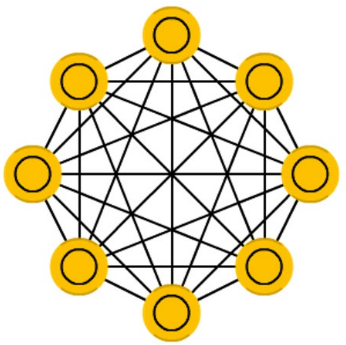
前馈神经网络

- 各个神经元按照接收信息的先后分成不同的组，每一组可看作一个神经层
- 每一层中的神经元接收来自前一层神经元的输出，并输出给下一层神经元
- 整个网络中信息朝一个方向传播，没有反向的信息传播，可以用一个有向无环图表示
- 常见的前馈网络包括：全连接神经网络 与 卷积神经网络

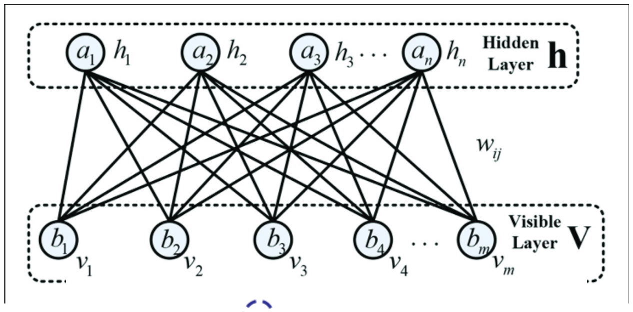


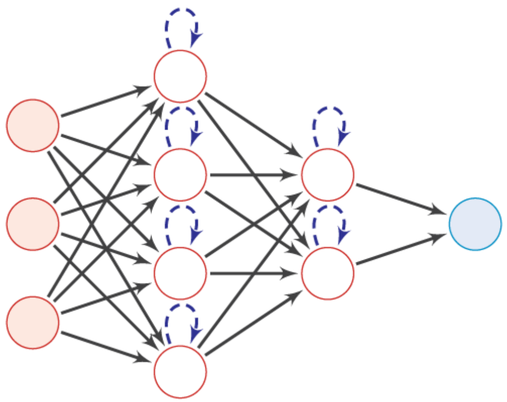
神经网络基础

递归神经网络



- 不一定具有明显的层次结构
- 信息的传播无严格的方向性
- 每个神经元除了接收来自其他神经元的信息，还可能接收自己的历史信息
- 把神经元的信息处理看作是一个时间历程
- 常见的递归神经网络包括：循环神经网络、Hopfield网络、受限玻尔兹曼机等。





内容概要

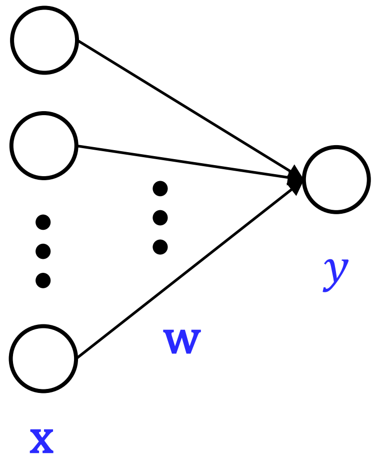
- 内容回顾
- 神经网络基础
- **全连接神经网络**
- 总结

全连接神经网络

单输出感知机

$$y = \sigma\left(\sum_{i=1}^{d} w_i x_{(i)} + b\right) = \sigma(\mathbf{w}^T \mathbf{x}) \qquad \mathbf{w} \in \mathbb{R}^D$$

这里省略了偏置 b



多输出感知机

$$y_1 = \sigma(\mathbf{w}_1{}^T \mathbf{x})$$

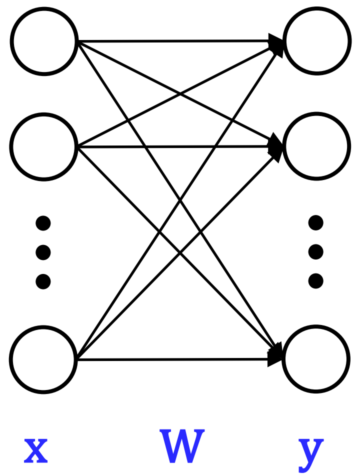
$$y_2 = \sigma(\mathbf{w}_2{}^T \mathbf{x})$$

$$\vdots$$

$$y_M = \sigma(\mathbf{w}_M{}^T \mathbf{x})$$

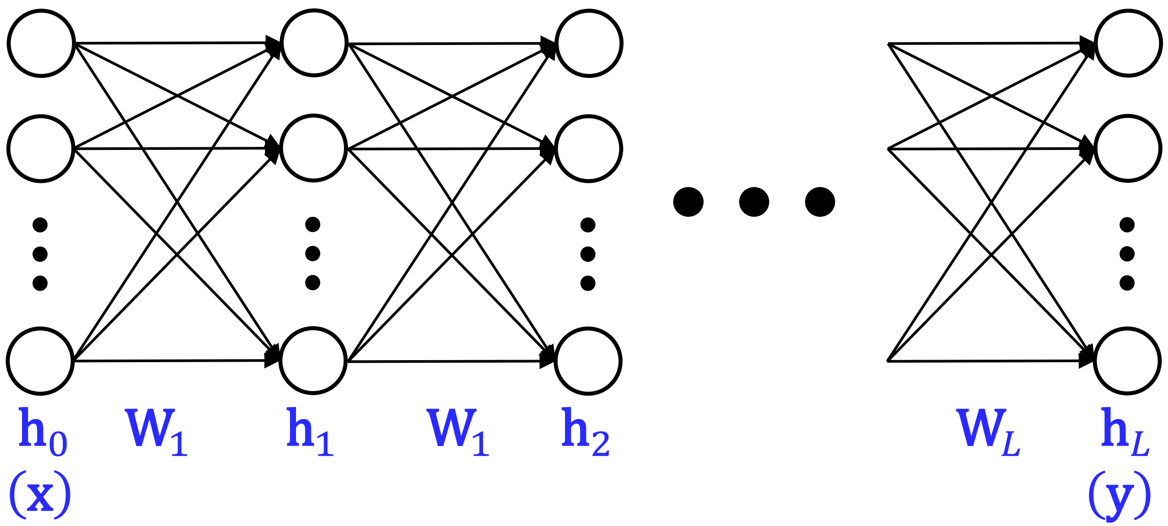
$$\mathbf{W} \in \mathbb{R}^{M \times D}$$

$$\mathbf{y} = \sigma(\mathbf{W}\mathbf{x})$$



全连接神经网络

多层感知机



$$\mathbf{h}_1 = \sigma(\mathbf{W_1}\mathbf{h}_0)$$

$$\mathbf{h}_2 = \sigma(\mathbf{W_2}\mathbf{h}_1)$$

$$\cdots$$

$$\mathbf{h}_L = \sigma(\mathbf{W}_L\mathbf{h}_{L-1})$$

$$\mathbf{h}_i = \begin{cases} \mathbf{x} & i = 0 \\ \sigma(\mathbf{W}_i\mathbf{h}_{i-1}) & 1 \leq i \leq L \end{cases}$$

$$\mathbf{y} = \mathbf{h}_L$$

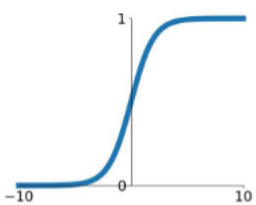
全连接神经网络

常见激活函数

$$\mathbf{h}_i = \begin{cases} \mathbf{x} & i = 0 \\ \sigma(\mathbf{W}_i \mathbf{h}_{i-1}) & 1 \le i \le L \end{cases}$$

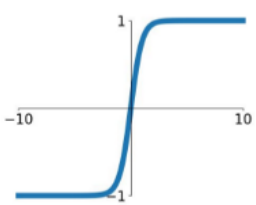
Sigmoid

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



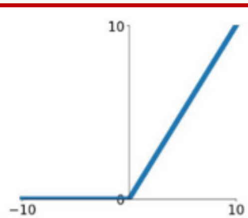
tanh

$\tanh(x)$



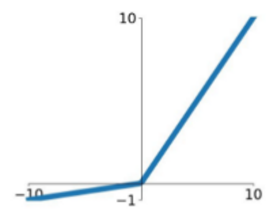
ReLU

$\max(0, x)$



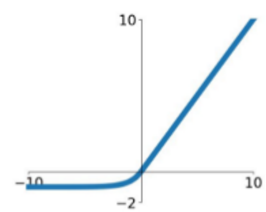
Leaky ReLU

$\max(0.1x, x)$



ELU

$$\begin{cases} x & x \ge 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



全连接神经网络

全连接神经网络学习

- 损失函数

$$l(\mathbf{y},\mathbf{h}_L) \qquad \mathbf{h}_i = \begin{cases} \mathbf{x} & i=0 \\ \sigma(\mathbf{W}_i\mathbf{h}_{i-1}) & 1 \le i \le L \end{cases}$$

例如 $l(\mathbf{y},\mathbf{h}_L) = \|\mathrm{y} - \mathbf{h}_L\|^2$

- 待优化变量

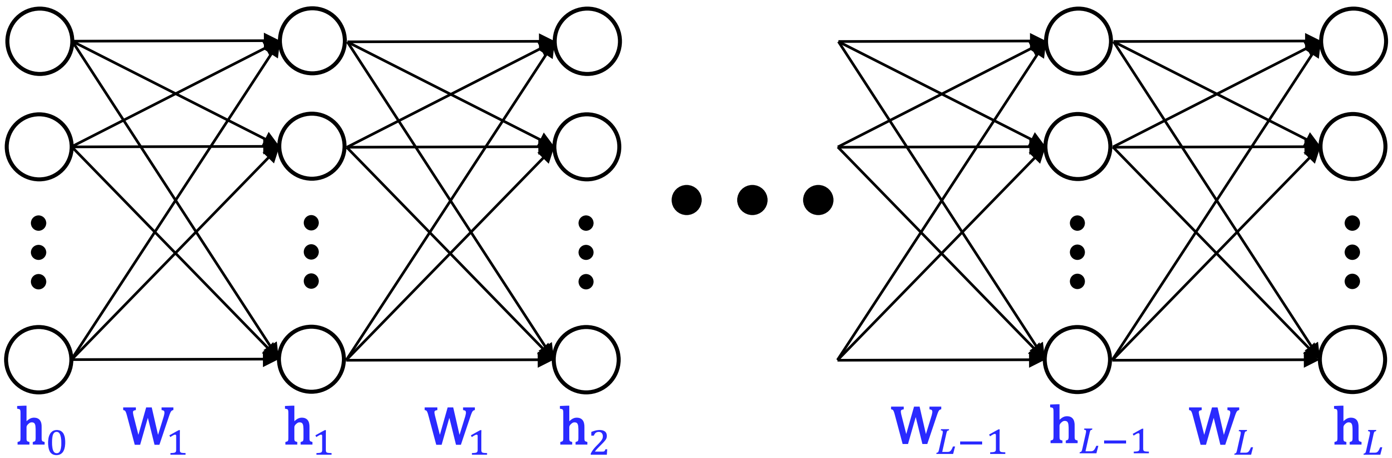
$$\mathcal{W} = \{\mathbf{W_1}, \mathbf{W_2}, \dots, \mathbf{W}_L\}$$

- 梯度下降法更新

$$\frac{\partial l}{\mathcal{W}}$$ 如何计算?

全连接神经网络

反向传播算法



$$l(\mathbf{y}, \mathbf{h}_L) \qquad \mathbf{h}_i = \begin{cases} \mathbf{x} & i = 0 \\ \sigma(\mathbf{W}_i \mathbf{h}_{i-1}) & 1 \le i \le L \end{cases}$$

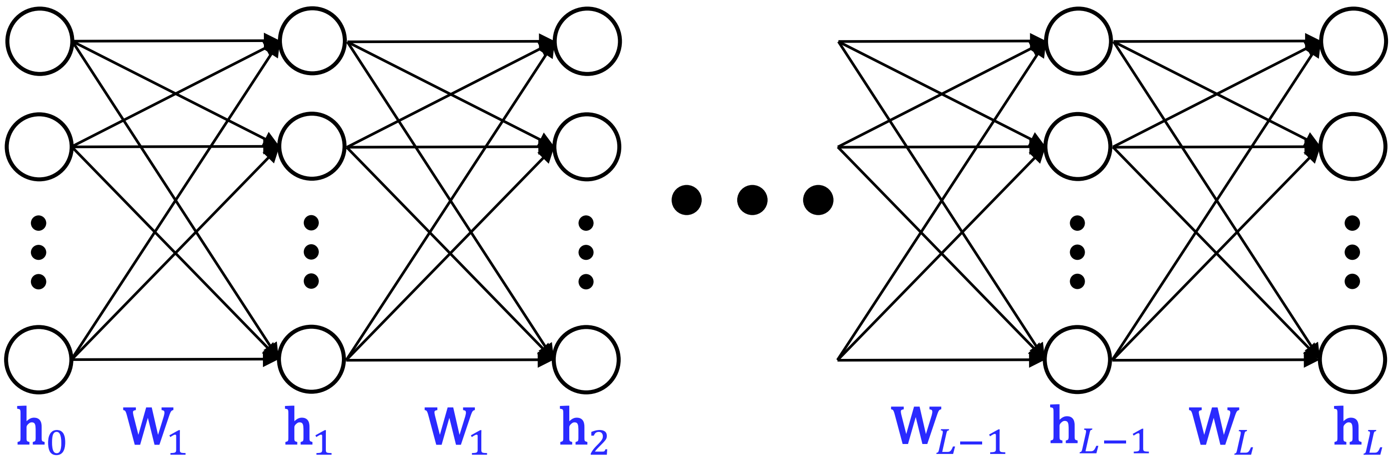
- 微分守恒法则

$\langle A, B \rangle$表示两个同尺寸对象对于相同的维度做“内积”

$$\mathrm{d}g(\mathbf{x}) = \left\langle \frac{\partial g(\mathbf{x})}{\partial \mathbf{x}}, \mathrm{d}\mathbf{x} \right\rangle$$

全连接神经网络

反向传播算法



$$l(\mathbf{y}, \mathbf{h}_L) \qquad \mathbf{h}_i = \begin{cases} \mathbf{x} & i = 0 \\ \sigma(\mathbf{W}_i \mathbf{h}_{i-1}) & 1 \le i \le L \end{cases}$$

- 微分守恒法则

$\langle A, B \rangle$表示两个同尺寸对象对于相同的维度做“内积”

$$\mathrm{d}l = \left\langle \frac{\partial l}{\partial \mathbf{h}_L}, \mathrm{d}\mathbf{h}_L \right\rangle = \left\langle \frac{\partial l}{\partial \mathbf{W}_L}, \mathrm{d}\mathbf{W}_L \right\rangle = \left\langle \frac{\partial l}{\partial \mathbf{h}_{L-1}}, \mathrm{d}\mathbf{h}_{L-1} \right\rangle \quad \bullet\bullet\bullet$$

全连接神经网络

反向传播算法

$$l(\mathbf{y}, \mathbf{h}_L) \qquad \mathbf{h}_i = \begin{cases} \mathbf{x} & i = 0 \\ \sigma(\mathbf{W}_i \mathbf{h}_{i-1}) & 1 \le i \le L \end{cases}$$

- 微分守恒法则

$$\mathrm{d}l = \left\langle \frac{\partial l}{\partial \mathbf{h}_L}, \mathrm{d}\mathbf{h}_L \right\rangle$$

$$\mathrm{d}\mathbf{h}_L = \left\langle \frac{\partial \sigma(\mathbf{W}_L \mathbf{h}_{L-1})}{\partial \mathbf{W}_L \mathbf{h}_{L-1}}, \mathrm{d}\mathbf{W}_L \mathbf{h}_{L-1} \right\rangle = \sigma'(\mathbf{W}_L \mathbf{h}_{L-1}) \odot \mathrm{d}\mathbf{W}_L \mathbf{h}_{L-1}$$

$\odot$ 为哈达玛积（点对点相乘）

$$\mathrm{d}l = \left\langle \frac{\partial l}{\partial \mathbf{h}_L}, \mathrm{d}\mathbf{h}_L \right\rangle = \left\langle \frac{\partial l}{\partial \mathbf{h}_L}, \sigma'(\mathbf{W}_L \mathbf{h}_{L-1}) \odot \mathrm{d}\mathbf{W}_L \mathbf{h}_{L-1} \right\rangle = \left\langle \sigma'(\mathbf{W}_L \mathbf{h}_{L-1}) \odot \frac{\partial l}{\partial \mathbf{h}_L}, \mathrm{d}\mathbf{W}_L \mathbf{h}_{L-1} \right\rangle$$

$$\mathrm{d}\mathbf{W}_L \mathbf{h}_{L-1} = \left\langle \frac{\partial \mathbf{W}_L \mathbf{h}_{L-1}}{\partial \mathbf{W}_L}, \mathrm{d}\mathbf{W}_L \right\rangle = \langle \mathbf{I} \otimes {\mathbf{h}_{L-1}}^T, \mathrm{d}\mathbf{W}_L \rangle$$

$\otimes$为克罗内克积

$$A \otimes B = \begin{bmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{bmatrix}$$

$$\mathrm{d}l = \left\langle \left(\mathbf{I} \otimes {\mathbf{h}_{L-1}}^T\right)^T \left(\sigma'(\mathbf{W}_L \mathbf{h}_{L-1}) \odot \frac{\partial l}{\partial \mathbf{h}_L}\right), \mathrm{d}\mathbf{W}_L \right\rangle$$

全连接神经网络

反向传播算法

$$l(\mathbf{y}, \mathbf{h}_L) \qquad \mathbf{h}_i = \begin{cases} \mathbf{x} & i = 0 \\ \sigma(\mathbf{W}_i \mathbf{h}_{i-1}) & 1 \le i \le L \end{cases}$$

- 微分守恒法则

$$\mathrm{d}l = \left\langle \frac{\partial l}{\partial \mathbf{h}_L}, \mathrm{d}\mathbf{h}_L \right\rangle$$

$$\mathrm{d}\mathbf{h}_L = \left\langle \frac{\partial \sigma(\mathbf{W}_L \mathbf{h}_{L-1})}{\partial \mathbf{W}_L \mathbf{h}_{L-1}}, \mathrm{d}\mathbf{W}_L \mathbf{h}_{L-1} \right\rangle = \sigma'(\mathbf{W}_L \mathbf{h}_{L-1}) \odot \mathrm{d}\mathbf{W}_L \mathbf{h}_{L-1}$$

$\odot$ 为哈达玛积（点对点相乘）

$$\mathrm{d}l = \left\langle \frac{\partial l}{\partial \mathbf{h}_L}, \mathrm{d}\mathbf{h}_L \right\rangle = \left\langle \frac{\partial l}{\partial \mathbf{h}_L}, \sigma'(\mathbf{W}_L \mathbf{h}_{L-1}) \odot \mathrm{d}\mathbf{W}_L \mathbf{h}_{L-1} \right\rangle = \left\langle \sigma'(\mathbf{W}_L \mathbf{h}_{L-1}) \odot \frac{\partial l}{\partial \mathbf{h}_L}, \mathrm{d}\mathbf{W}_L \mathbf{h}_{L-1} \right\rangle$$

$$\mathrm{d}\mathbf{W}_L \mathbf{h}_{L-1} = \left\langle \frac{\partial \mathbf{W}_L \mathbf{h}_{L-1}}{\partial \mathbf{h}_{L-1}}, \mathrm{d}\mathbf{h}_{L-1} \right\rangle = \langle \mathbf{W}_L, \mathrm{d}\mathbf{h}_{L-1} \rangle$$

$$\mathrm{d}l = \left\langle \mathbf{W}_L{}^T \left(\sigma'(\mathbf{W}_L \mathbf{h}_{L-1}) \odot \frac{\partial l}{\partial \mathbf{h}_L} \right), \mathrm{d}\mathbf{h}_{L-1} \right\rangle$$

全连接神经网络

反向传播算法

$$l(\mathbf{y}, \mathbf{h}_L) \qquad \mathbf{h}_i = \begin{cases} \mathbf{x} & i = 0 \\ \sigma(\mathbf{W}_i \mathbf{h}_{i-1}) & 1 \le i \le L \end{cases}$$

- 微分守恒法则

$$\mathrm{d}l = \left\langle \frac{\partial l}{\partial \mathbf{h}_L}, \mathrm{d}\mathbf{h}_L \right\rangle = \left\langle \frac{\partial l}{\partial \mathbf{W}_L}, \mathrm{d}\mathbf{W}_L \right\rangle = \left\langle \frac{\partial l}{\partial \mathbf{h}_{L-1}}, \mathrm{d}\mathbf{h}_{L-1} \right\rangle \quad \bullet\bullet\bullet$$

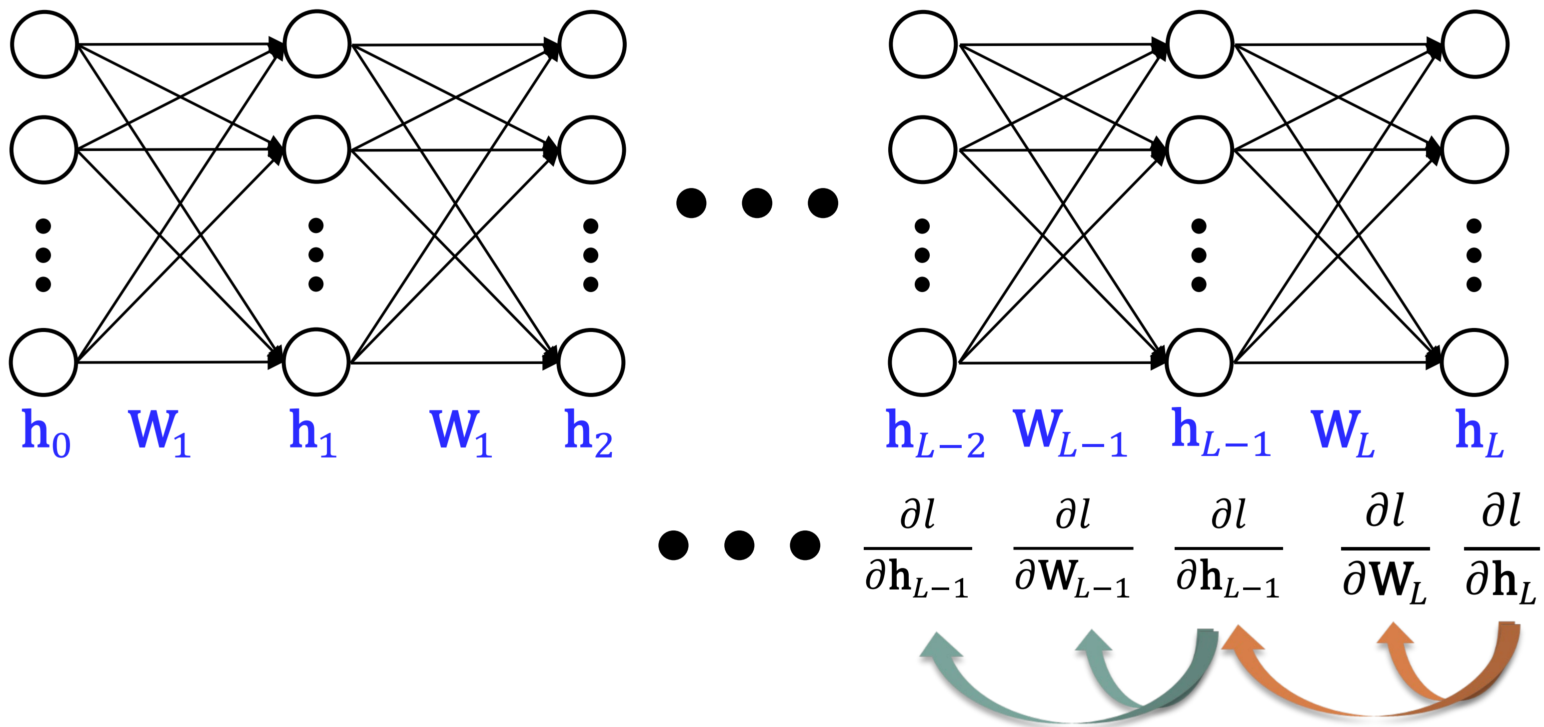
$$\mathrm{d}l = \left\langle \left(\mathbf{I} \otimes \mathbf{h}_{L-1}{}^{T}\right)^{T} \left(\sigma'(\mathbf{W}_L \mathbf{h}_{L-1}) \odot \frac{\partial l}{\partial \mathbf{h}_L}\right), \mathrm{d}\mathbf{W}_L \right\rangle \qquad \mathrm{d}l = \left\langle \mathbf{W}_L{}^{T} \left(\sigma'(\mathbf{W}_L \mathbf{h}_{L-1}) \odot \frac{\partial l}{\partial \mathbf{h}_L}\right), \mathrm{d}\mathbf{h}_{L-1} \right\rangle$$

$$\Downarrow \qquad\qquad \Downarrow$$

$$\frac{\partial l}{\partial \mathbf{W}_L} = \left(\mathbf{I} \otimes \mathbf{h}_{L-1}{}^{T}\right)^{T} \left(\sigma'(\mathbf{W}_L \mathbf{h}_{L-1}) \odot \frac{\partial l}{\partial \mathbf{h}_L}\right) \qquad \frac{\partial l}{\partial \mathbf{h}_{L-1}} = \mathbf{W}_L{}^{T} \left(\sigma'(\mathbf{W}_L \mathbf{h}_{L-1}) \odot \frac{\partial l}{\partial \mathbf{h}_L}\right)$$

全连接神经网络

反向传播算法



全连接神经网络

为何选择ReLu激活函数

$$\frac{\partial l}{\partial \mathbf{h}_{L-1}} = \mathbf{W}_L{}^T \left(\boxed{\sigma'(\mathbf{W}_L \mathbf{h}_{L-1})} \odot \frac{\partial l}{\partial \mathbf{h}_L} \right)$$

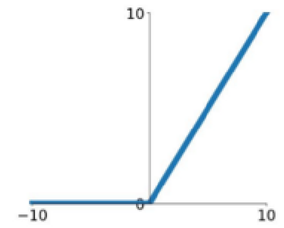
$$\frac{\partial l}{\partial \mathbf{h}_{L-2}} = \mathbf{W}_{L-1}{}^T \left(\boxed{\sigma'(\mathbf{W}_{L-1} \mathbf{h}_{L-2})} \odot \frac{\partial l}{\partial \mathbf{h}_{L-1}} \right)$$

$$\vdots$$

$$\frac{\partial l}{\partial \mathbf{h}_0} = \mathbf{W}_1{}^T \left(\boxed{\sigma'(\mathbf{W}_1 \mathbf{h}_0)} \odot \frac{\partial l}{\partial \mathbf{h}_1} \right)$$

ReLU

$\max(0, x)$



$$\lim_{p \to +\infty} a^p = \begin{cases} \infty & a > 1 \\ 1 & a = 1 \\ 0 & 0 \le a < 1 \end{cases}$$

$$\sigma'(x) = \begin{cases} 1 & x > 0 \\ 0 & x < 0 \end{cases}$$

- ReLu系列激活函数能有效缓解梯度爆炸与梯度消失问题

全连接神经网络

反向传播算法

- 给定训练数据 $\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N)\}$
- 初始化神经网络参数 $\mathcal{W}$
- 基于递推式逐层计算神经网络梯度 $\frac{\partial l}{\mathcal{W}}$

$$\frac{\partial l}{\partial \mathbf{W}_i} = \left(\mathbf{I} \otimes \mathbf{h}_{i-1}{}^{T}\right)^{T} \left(\sigma'(\mathbf{W}_i \mathbf{h}_{i-1}) \odot \frac{\partial l}{\partial \mathbf{h}_i}\right) \qquad \frac{\partial l}{\partial \mathbf{h}_{i-1}} = \mathbf{W}_i{}^{T} \left(\sigma'(\mathbf{W}_i \mathbf{h}_{i-1}) \odot \frac{\partial l}{\partial \mathbf{h}_i}\right)$$

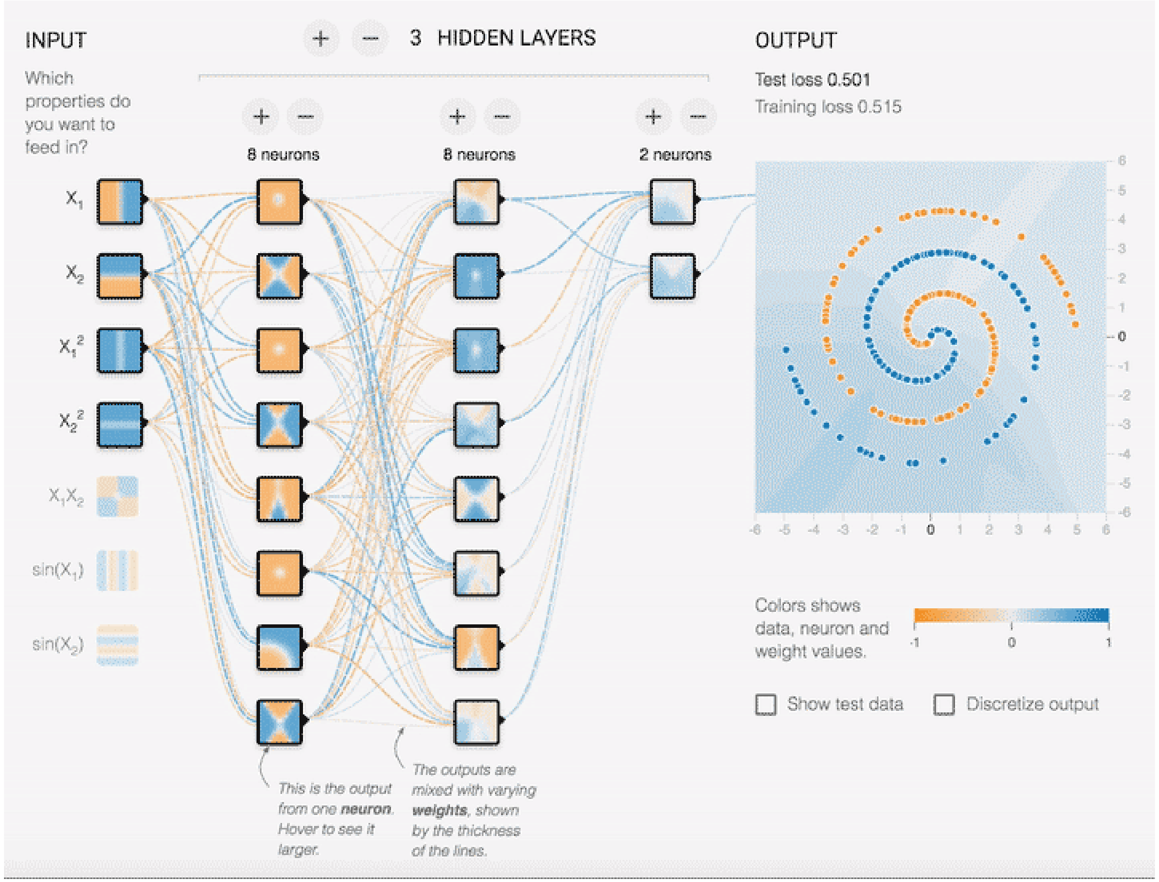
- 对于给定的学习率 η 进行参数更新

$$\mathcal{W} \leftarrow \mathcal{W} - \eta \frac{\partial l}{\mathcal{W}}$$

- 重复上述两步直至收敛

全连接神经网络

神经网络训练过程可视化



http://playground.tensorflow.org/

全连接神经网络

关于反向传播算法的进一步讨论

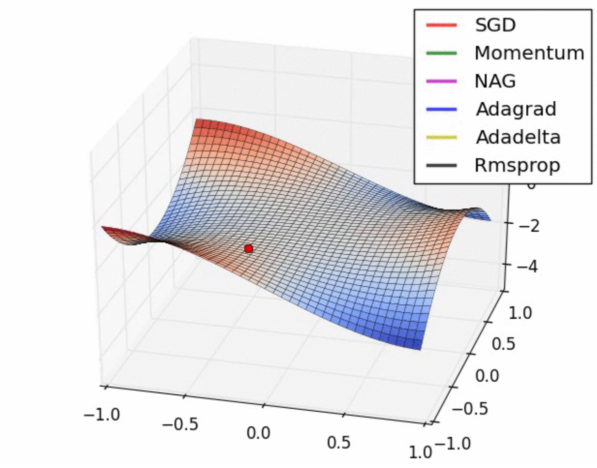
- 全批量数据更新

  对完整数据集计算梯度，能够快速收敛到局部最优

- 随机梯度下降（主流）

  每次随机抽取一部分数据计算梯度并进行更新，有助于跳出局部最优

Optimiser	Year	Learning Rate	Gradient
Momentum	1964		✓
AdaGrad	2011	✓	
RMSprop	2012	✓	
Adadelta	2012	✓	
Nesterov	2013		✓
Adam	2014	✓	✓
AdaMax	2015	✓	✓
Nadam	2015	✓	✓
AMSGrad	2018	✓	✓



https://towardsdatascience.com/10-gradient-descent-optimisation-algorithms-86989510b5e9

全连接神经网络

全连接神经网络表达能力讨论

- 只需要一个包含足够多神经元的隐层，多层前馈神经网络就能以任意精度逼近任意复杂度 的连续函数

  [Hornik et al. , 1989]

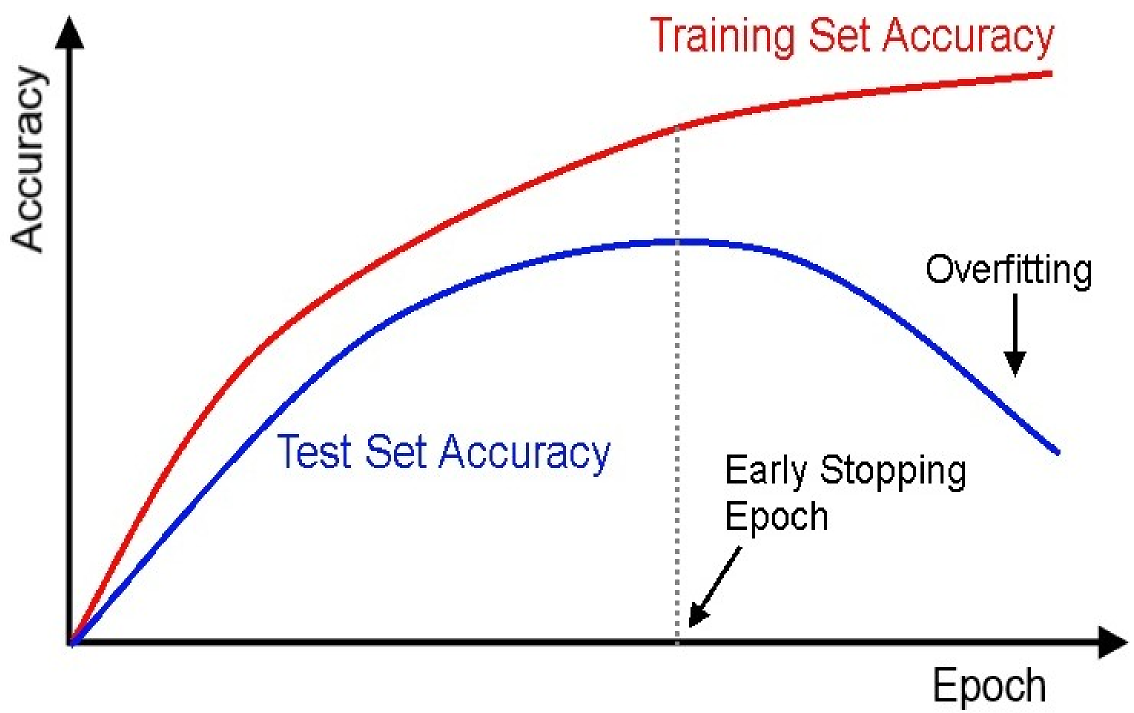
- 神经网络由于强大的表示能力，经常遭遇**过拟合**。表现为：训练误差持续降低，但测试误差却可能上升

- 实际使用中往往需要一些策略限制神经网络的表达能力

全连接神经网络

缓解过拟合的策略

- 早停（early stopping）：在训练过程中，若训练误差降低，但验证误差升高，则停止 训练

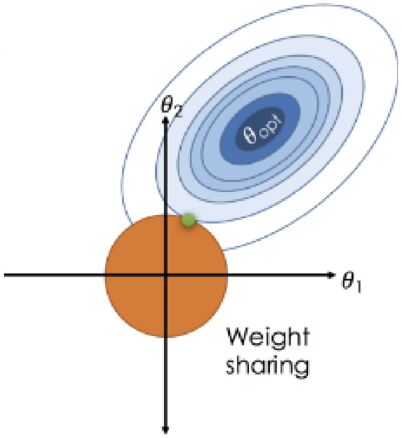


全连接神经网络

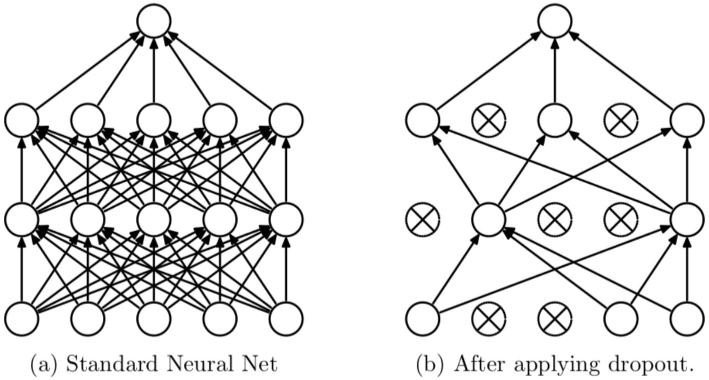
缓解过拟合的策略

- 增加正则项

$$L = loss(\mathcal{W}) + \lambda \|\mathcal{W}\|_2^2$$



- Drop-out 策略
  - 训练时，以概率1-p为概率丢弃掉一些神经元
  - 测试时，使用所有神经元，但要每个网络权重参数 w 乘以 p 以保证数量级的期望一致



全连接神经网络

深度学习主要开源框架

- PyTorch
  - Facebook人工智能研究小组领衔开发
  - 易上手、用户广泛、API丰富
- TensorFlow
  - Google Brain 领衔开发
  - 效率高、API丰富、工业界应用广发
- 飞桨（PaddlePaddle）
  - 百度深度学习研究院领衔开发
  - 国内最早框架，积累丰富，开发生态好
- 昇思（ MindSpore ）
  - 华为公司领衔开发
  - 后起之秀，API设计更现代，推广力度大

内容概要

- 内容回顾
- 神经网络基础
- 全连接神经网络
- **总结**

总结

- 神经网络基础
  - 发展历程
  - 人类神经网络工作原理
  - MCP神经元模型
  - 人工神经网络分类
- 全连接神经网络
  - 基本架构及形式化
  - 微分守恒法则推导反向传播算法
  - 缓解神经网络过拟合的策略

北京交通大學
BEIJING JIAOTONG UNIVERSITY

谢 谢!

问 题?