模式识别
Pattern Recognition

李泽榉，复旦大学 生物医学工程与技术创新学院

目录

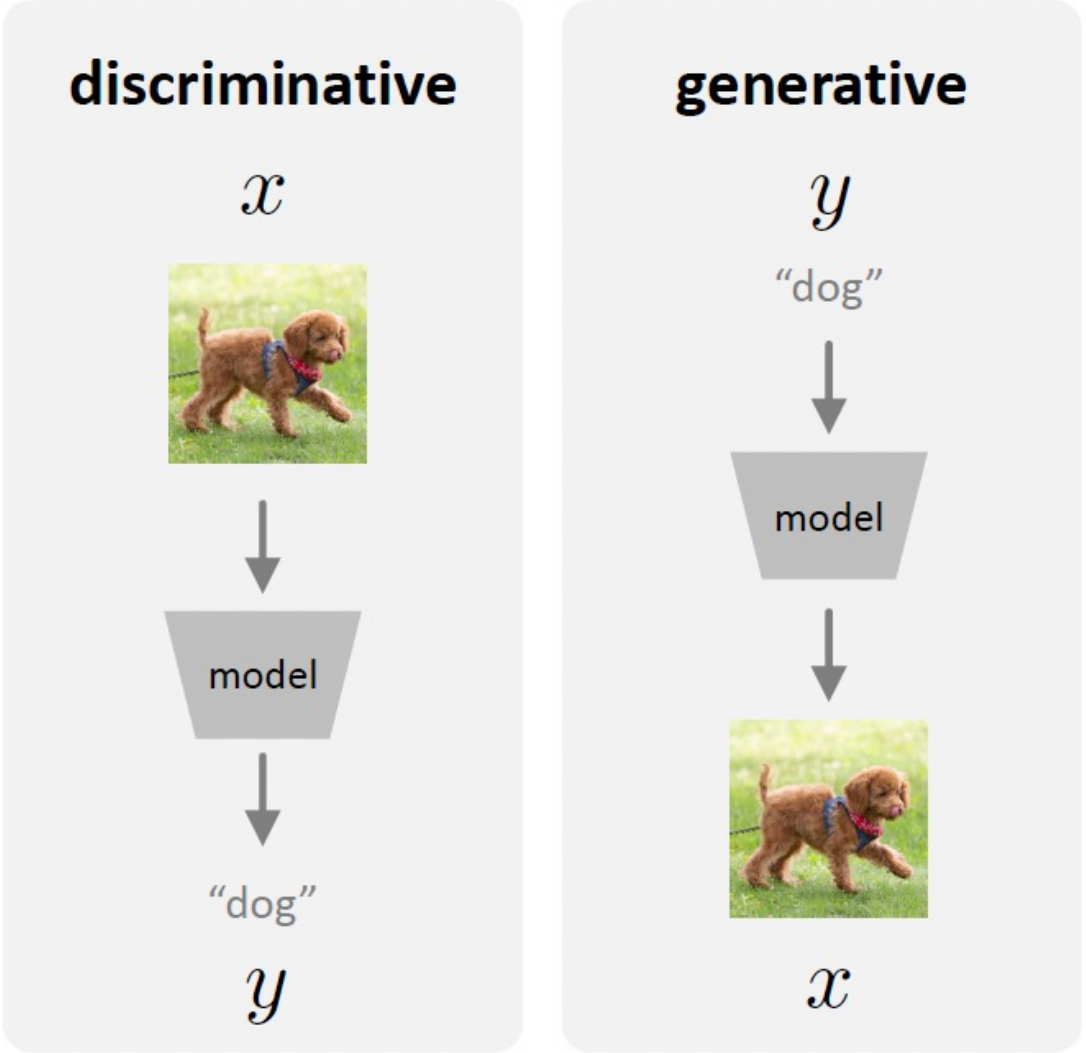
判别模型与生成模型

判别模型

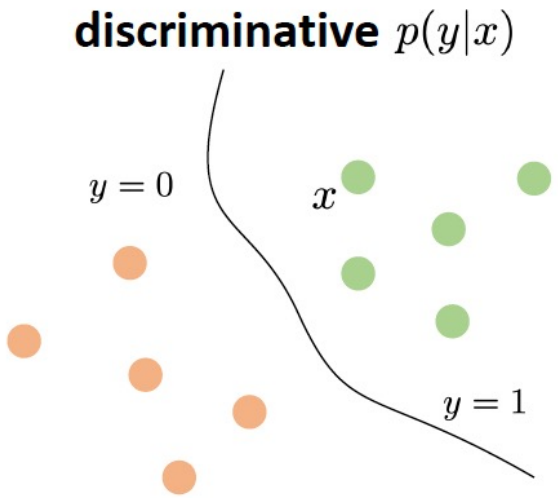
- ”样本”x ⇨ “类别”y
- 仅一个输出

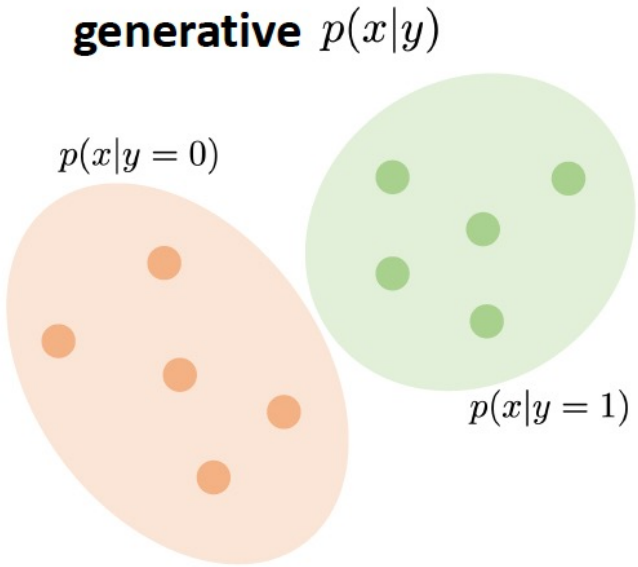
生成模型

- “类别”y ⇨ ”样本”x
- 多个可能输出



判别模型与生成模型





- 生成模型也可以具有判别性：贝叶斯法则
- 判别模型不具有生成性

判别模型与生成模型

- 生成模型也可以具有判别性：贝叶斯法则

$$p(y|x) = p(x|y)\frac{p(y)}{p(x)}$$

discriminative　　generative

类别先验

对于特定x，概率固定

判别模型与生成模型

- 生成模型也可以具有判别性：贝叶斯法则。

$$p(y|x) = p(x|y)\frac{p(y)}{p(x)}$$

discriminative　　generative

类别先验

对于特定x，概率固定

- 判别模型不具有生成性

$$p(x|y) = p(y|x)\frac{p(x)}{p(y)}$$

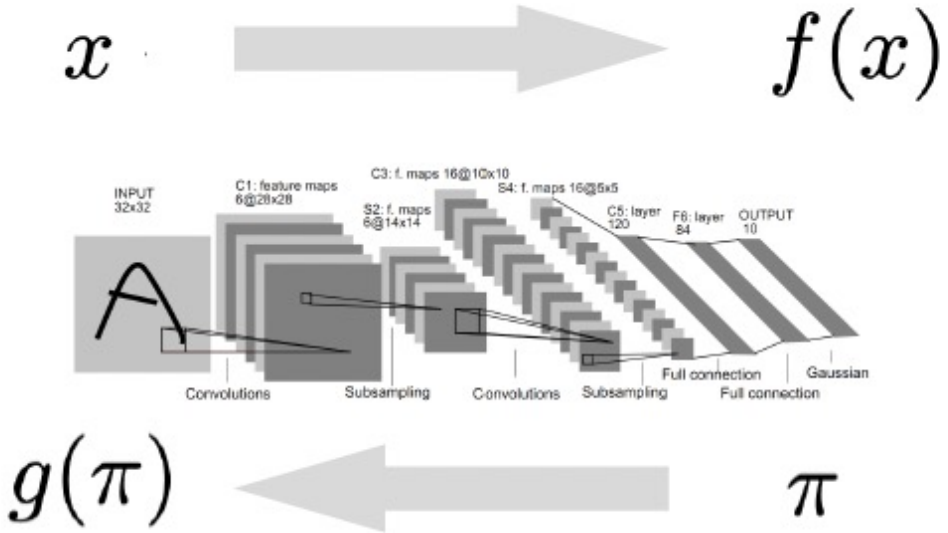
generative　　discriminative

x的分布无法获得

对于特定y，概率固定

判别模型与生成模型：一体两面

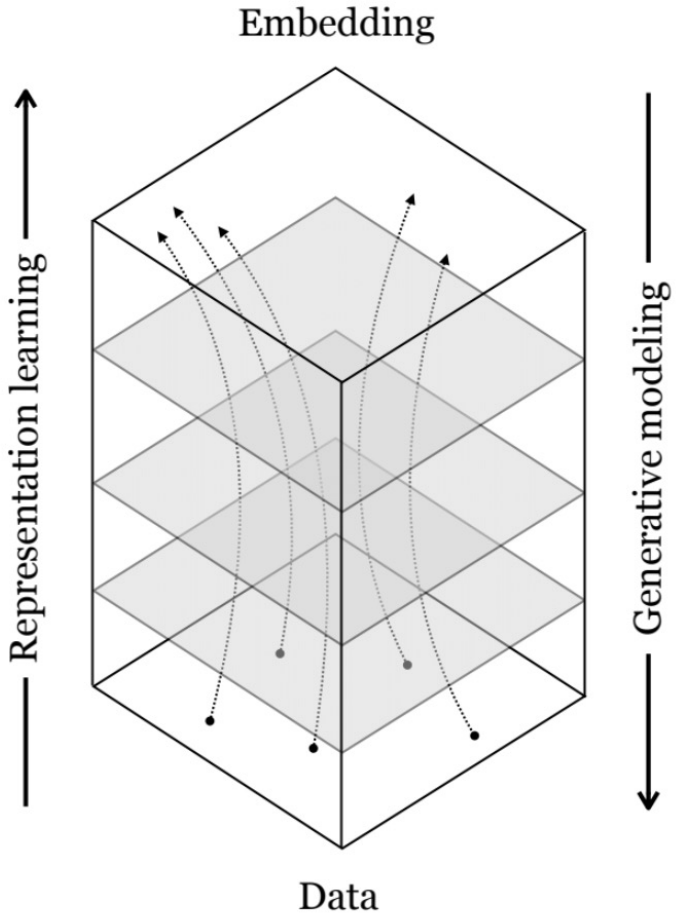
- 判别模型
  - 从数据映射到特征：$x \to f(x)$
  - 优化损失函数：$\mathcal{L}(y, f(x))$

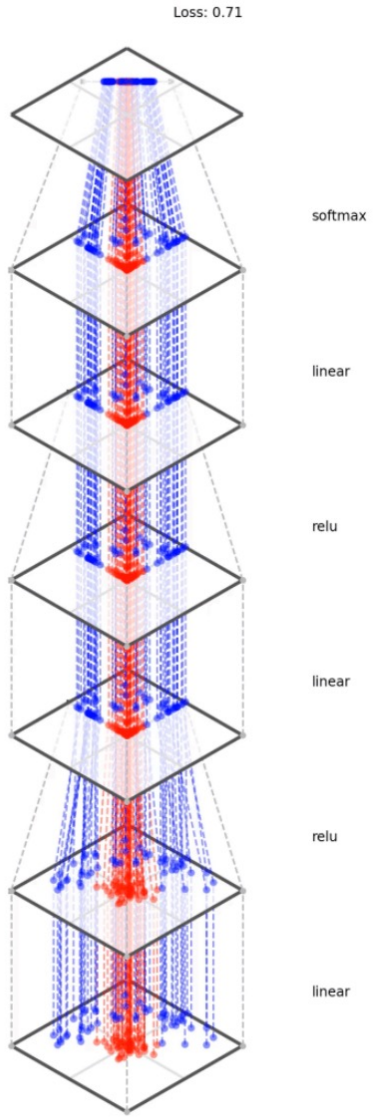


- 生成模型
  - 从简单分布映射到复杂分布：$\pi \to g(\pi)$
  - 优化损失函数：$\mathcal{L}(p_x, g(\pi))$

判别模型与生成模型：一体两面

- 深度网络学习每一层的特征映射
- 前向传播（观测数据->潜在表征）是判别模型
- 反向传播（潜在表征->观测数据）是生成模型





高斯过程

- 一个常用来做判别任务的**生成模型**
- 非参数化模型（Non-parametric functions）
- 高斯过程：观测值出现在一个连续域（例如时间或空间）的随机过程

$$(y(x_1), \ldots, y(x_n)) \sim N_n(\mu, \Sigma)$$

- $y(\cdot)$ 是x的function

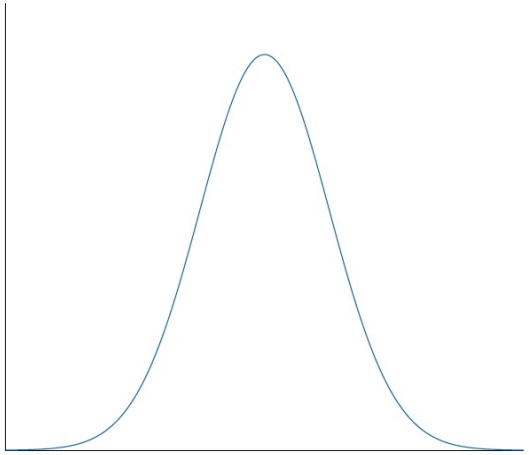
高斯分布

- 高斯分布：需要计算均值（μ）和方差（Σ）

$$X \sim N(\mu, \Sigma)$$

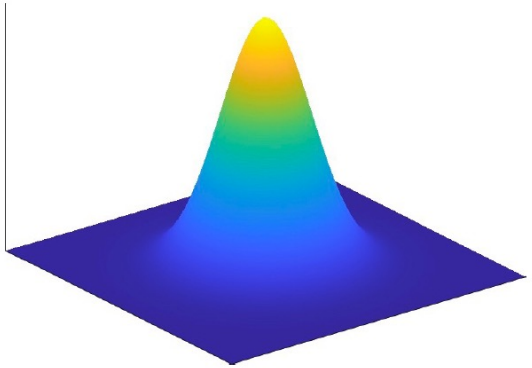
- (Univariate) Gaussians:

$$x \sim \mathcal{N}(x; \mu = 0, \sigma^2 = 1)$$



$$\mu = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} \qquad \Sigma = \begin{pmatrix} \sigma_1^2 & \rho_{12}\sigma_1\sigma_2 \\ \rho_{21}\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix}$$

$$\text{Cor}(Y_i, Y_j) = \rho_{12} \text{ for } i \neq j$$



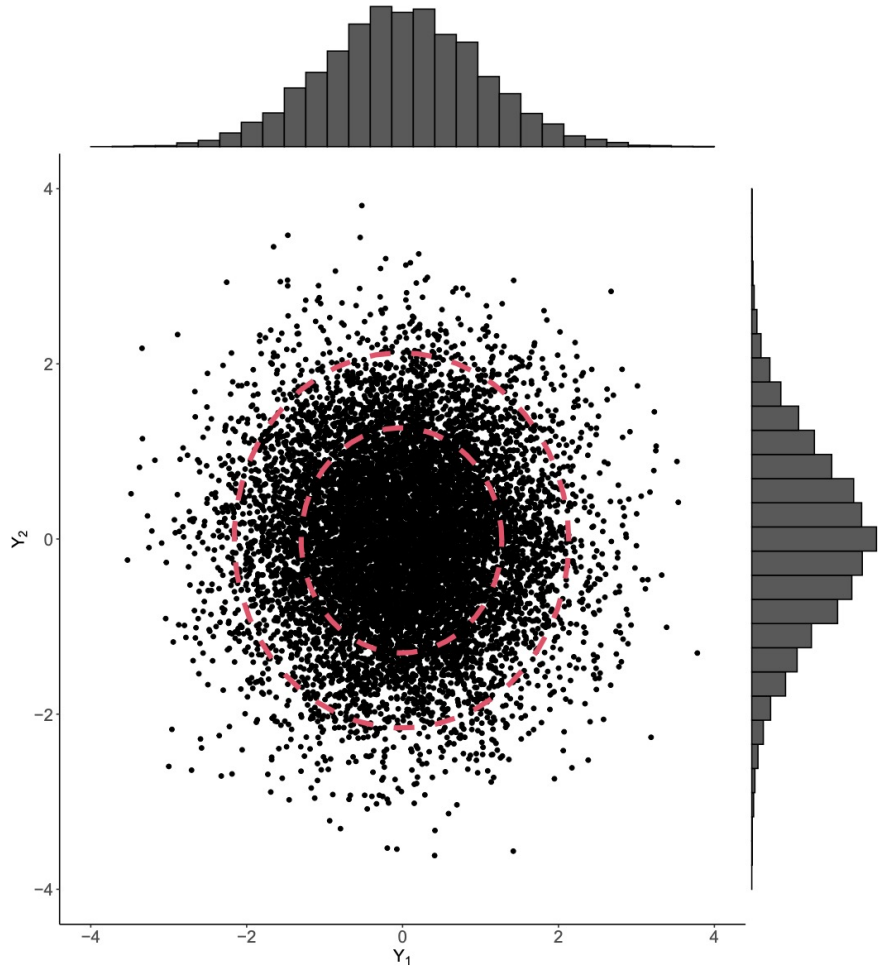
- Multivariate Gaussians:

$$\boldsymbol{x} = [x_1, \ldots, x_D]^T$$

$$\sim \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu} = \mathbf{0}_D, \Sigma = I_D)$$

二元高斯分布

- Y_1和 Y_2无关

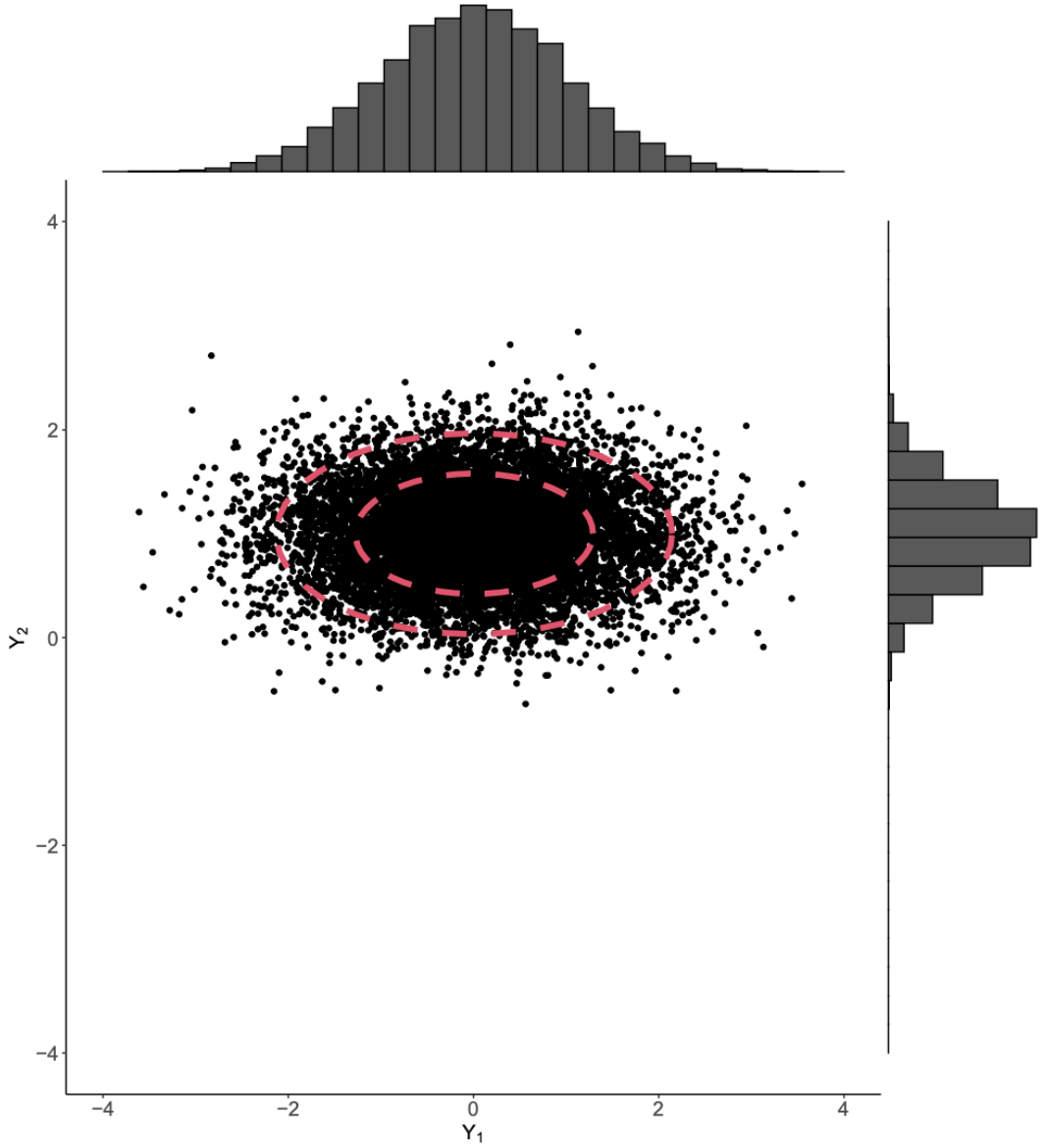


$$\mu = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$\Sigma = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

二元高斯分布

- Y_1和 Y_2无关

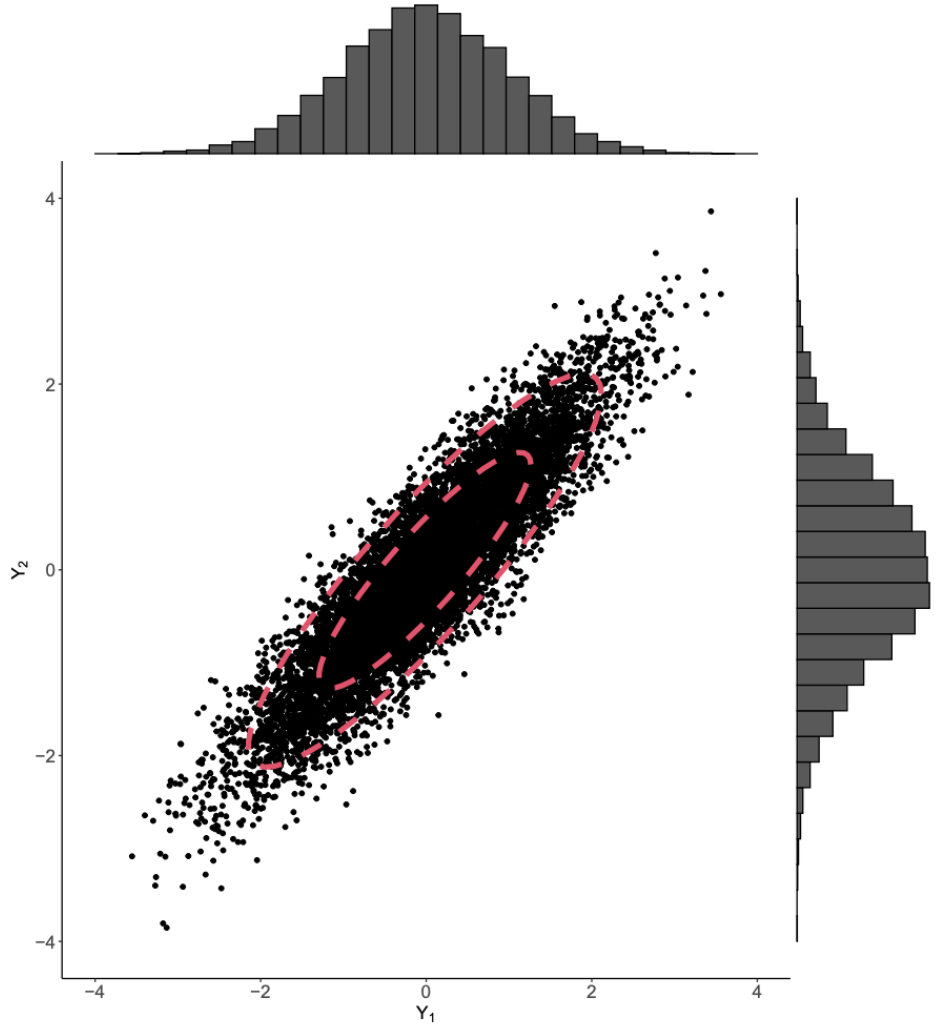


$$\mu = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$$\Sigma = \begin{pmatrix} 1 & 0 \\ 0 & 0.2 \end{pmatrix}$$

二元高斯分布

- Y_1和 Y_2相关

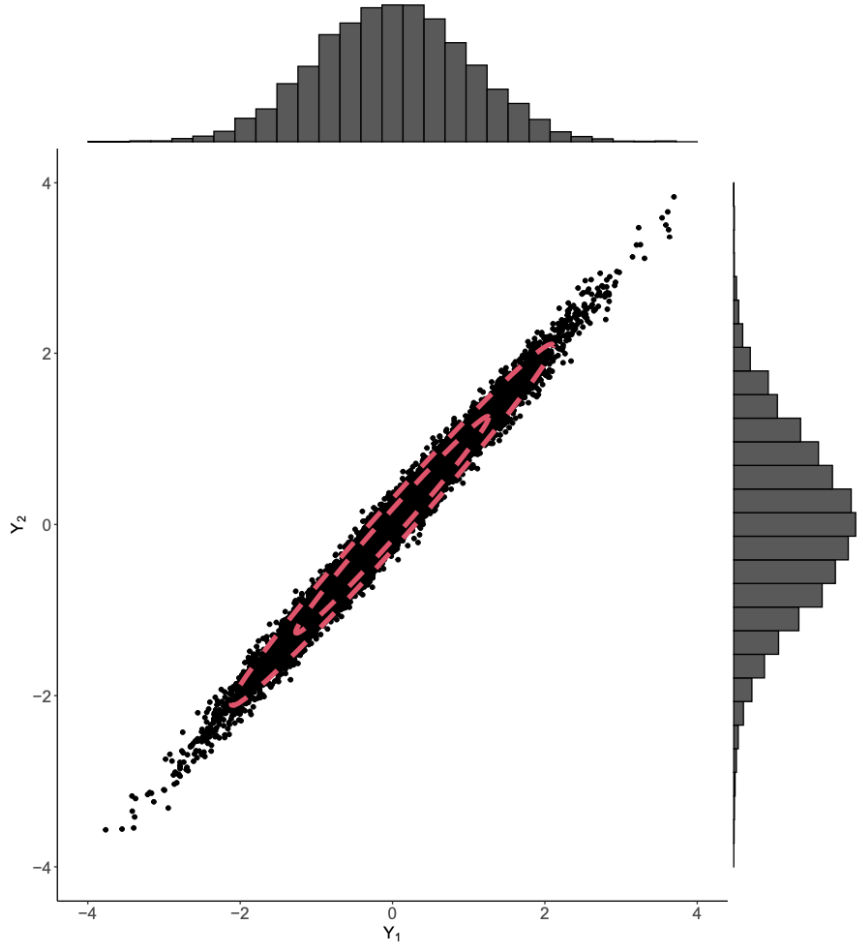


$$\mu = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$\Sigma = \begin{pmatrix} 1 & 0.9 \\ 0.9 & 1 \end{pmatrix}$$

二元高斯分布

- Y_1和 Y_2特别相关

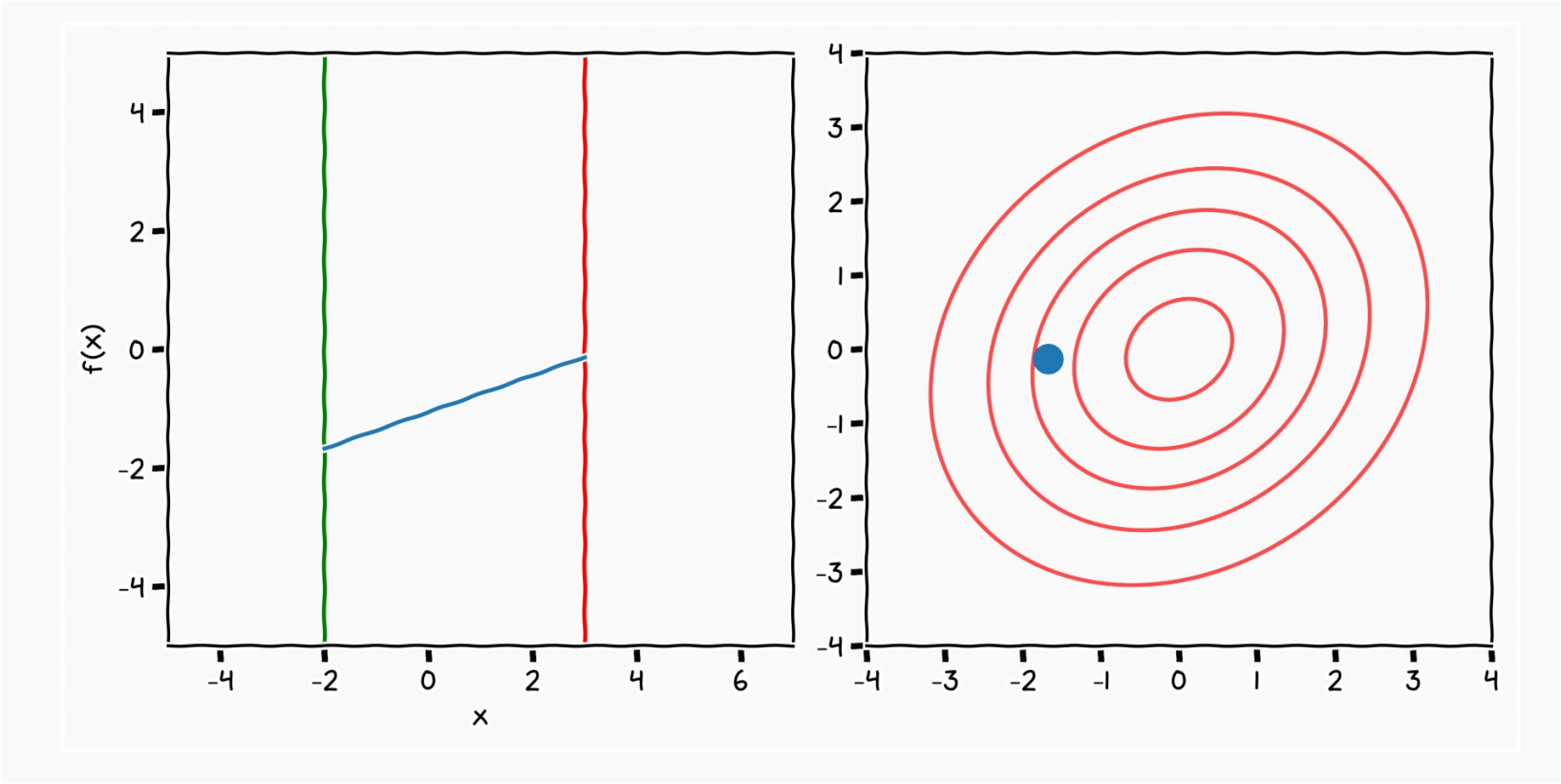


$$\mu = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$\Sigma = \begin{pmatrix} 1 & 0.99 \\ 0.99 & 1 \end{pmatrix}$$

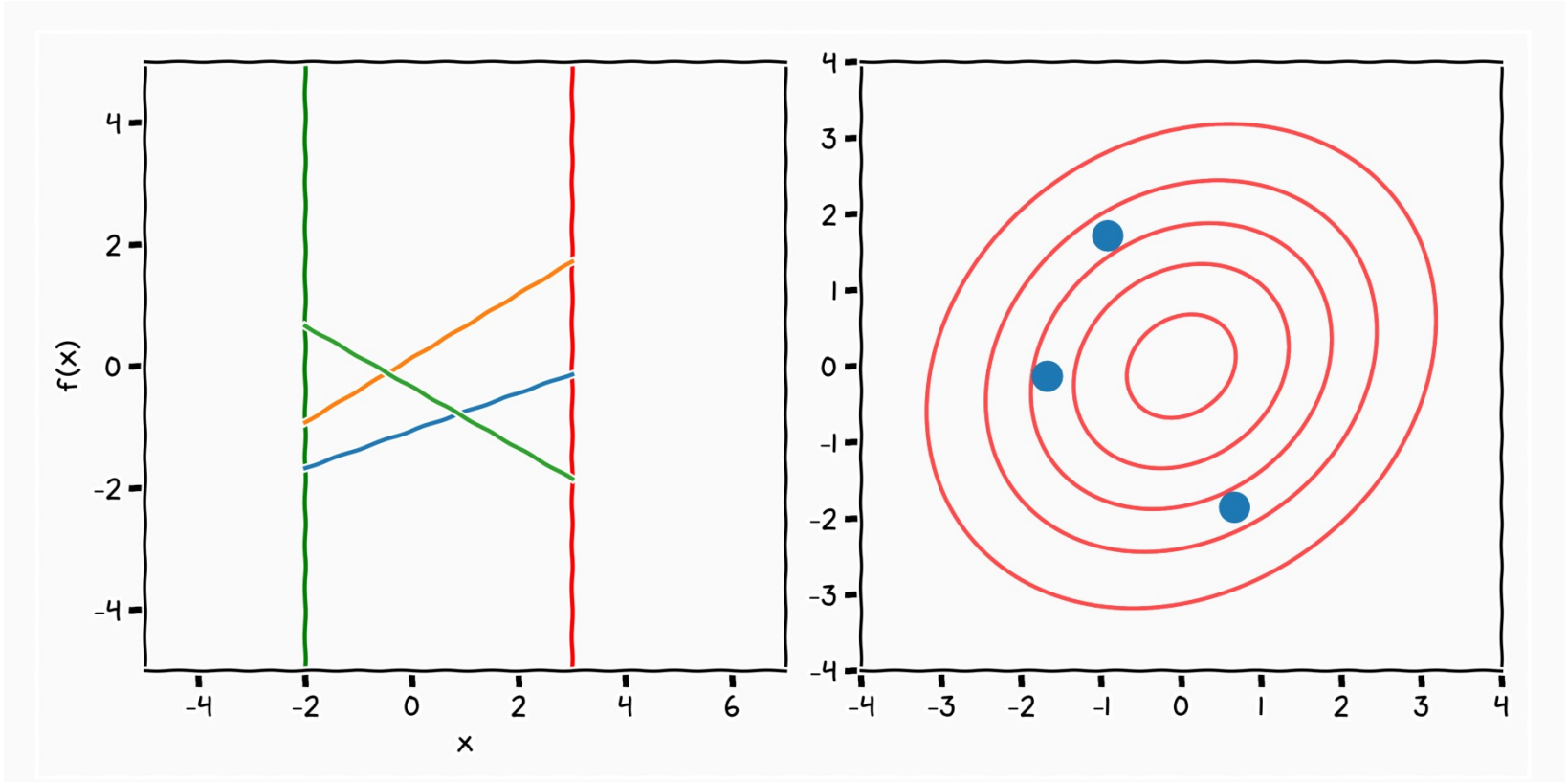
二元高斯分布

- 可以看作是从 Y_1到 Y_2的推断
- 右边的点=左边的线=单次采样



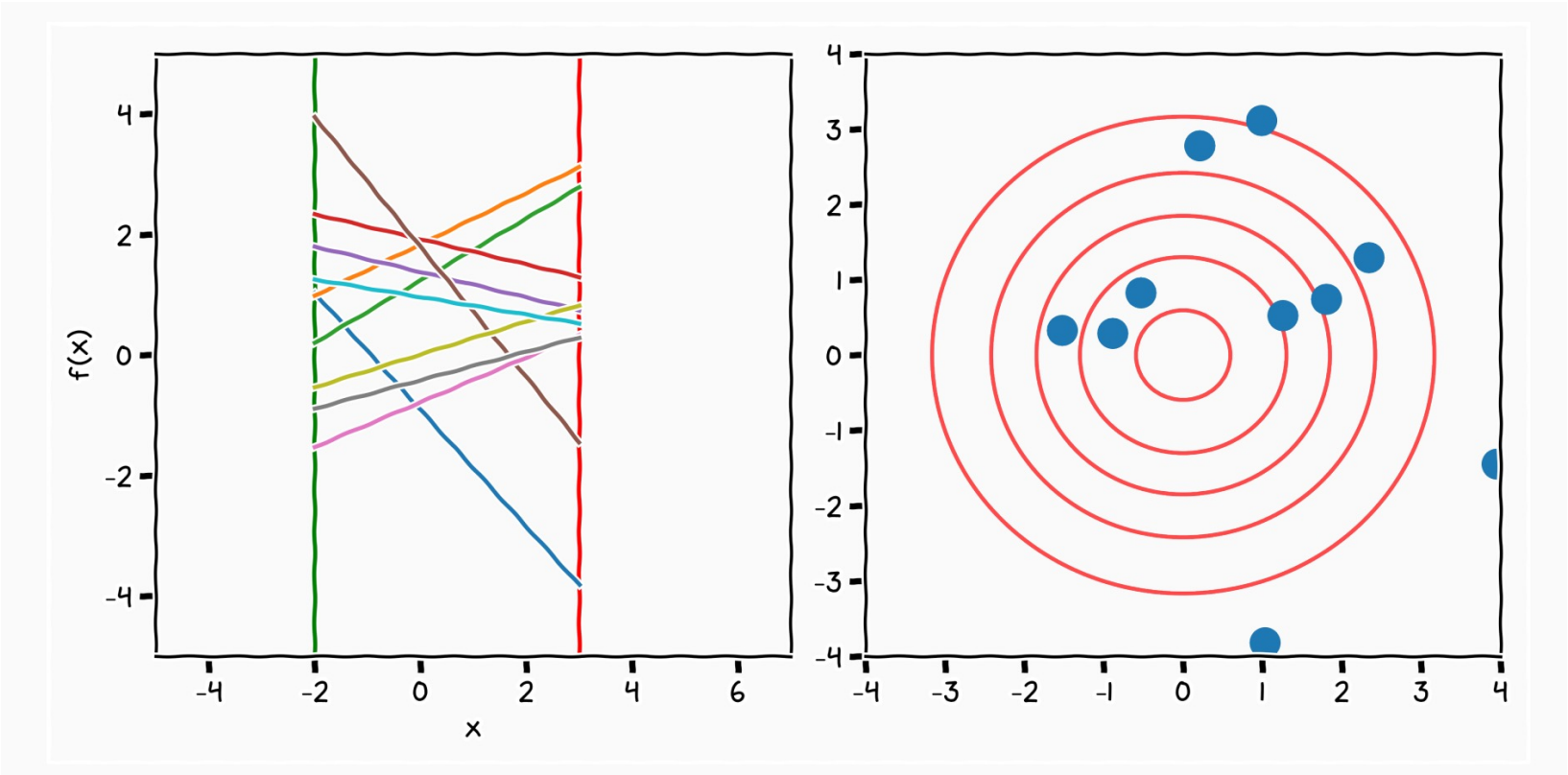
二元高斯分布

- 可以看作是从 Y_1到 Y_2的推断
- 右边的点=左边的线=多次采样



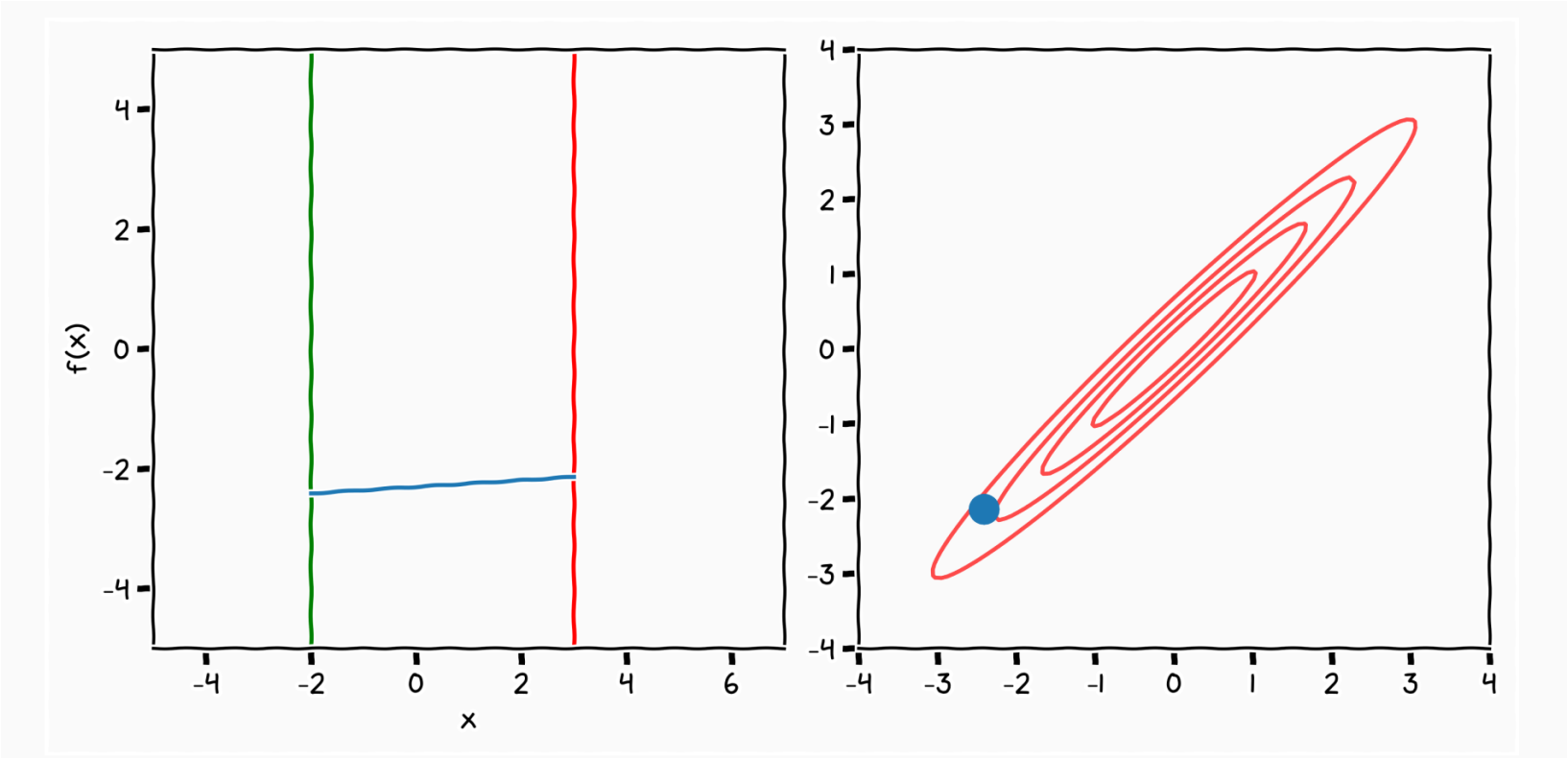
二元高斯分布

- 可以看作是从 Y_1到 Y_2的推断
- 右边的点=左边的线=多次采样，不相关



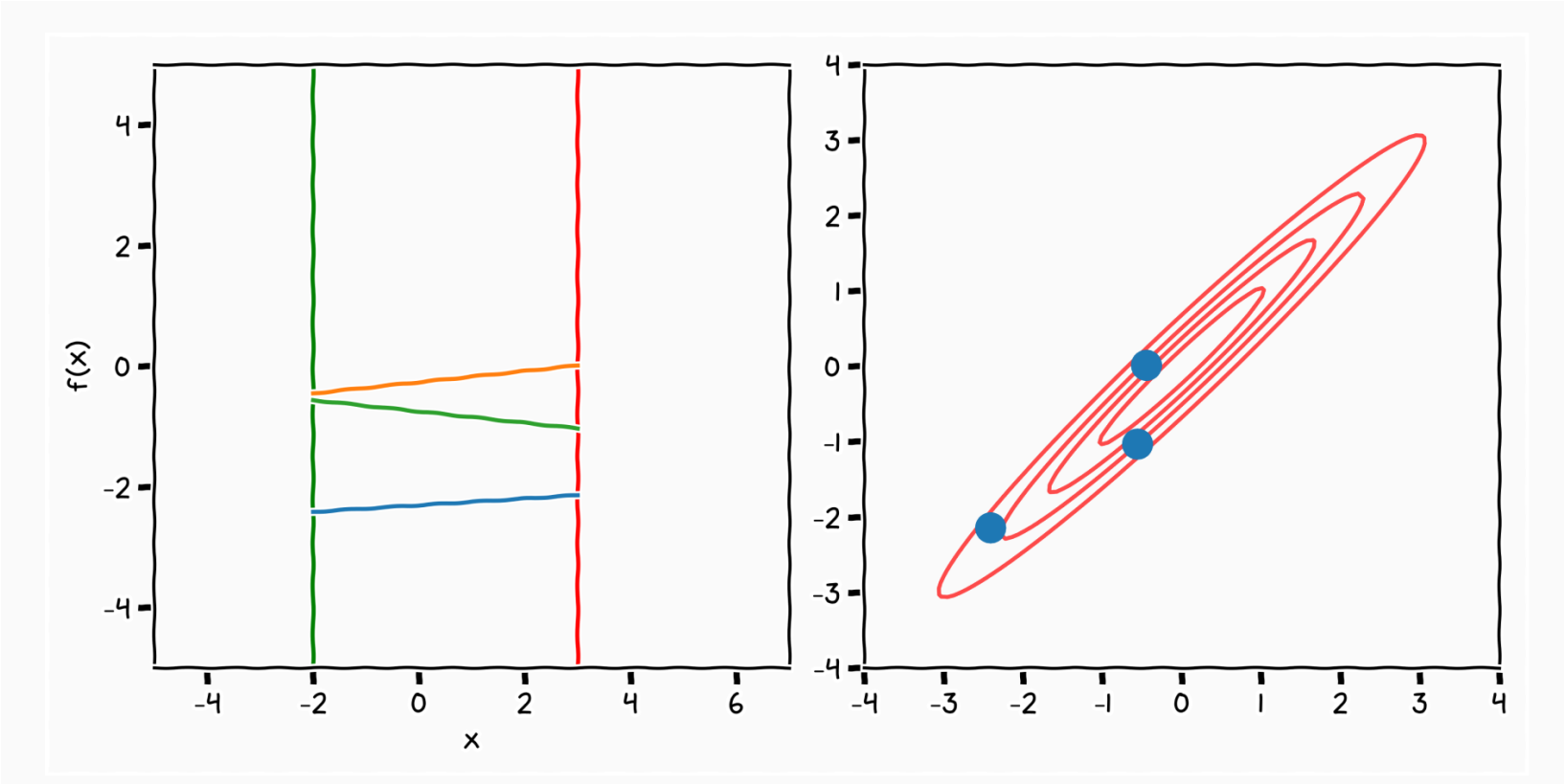
二元高斯分布

- 可以看作是从 Y_1到 Y_2的推断
- 右边的点=左边的线=单次采样



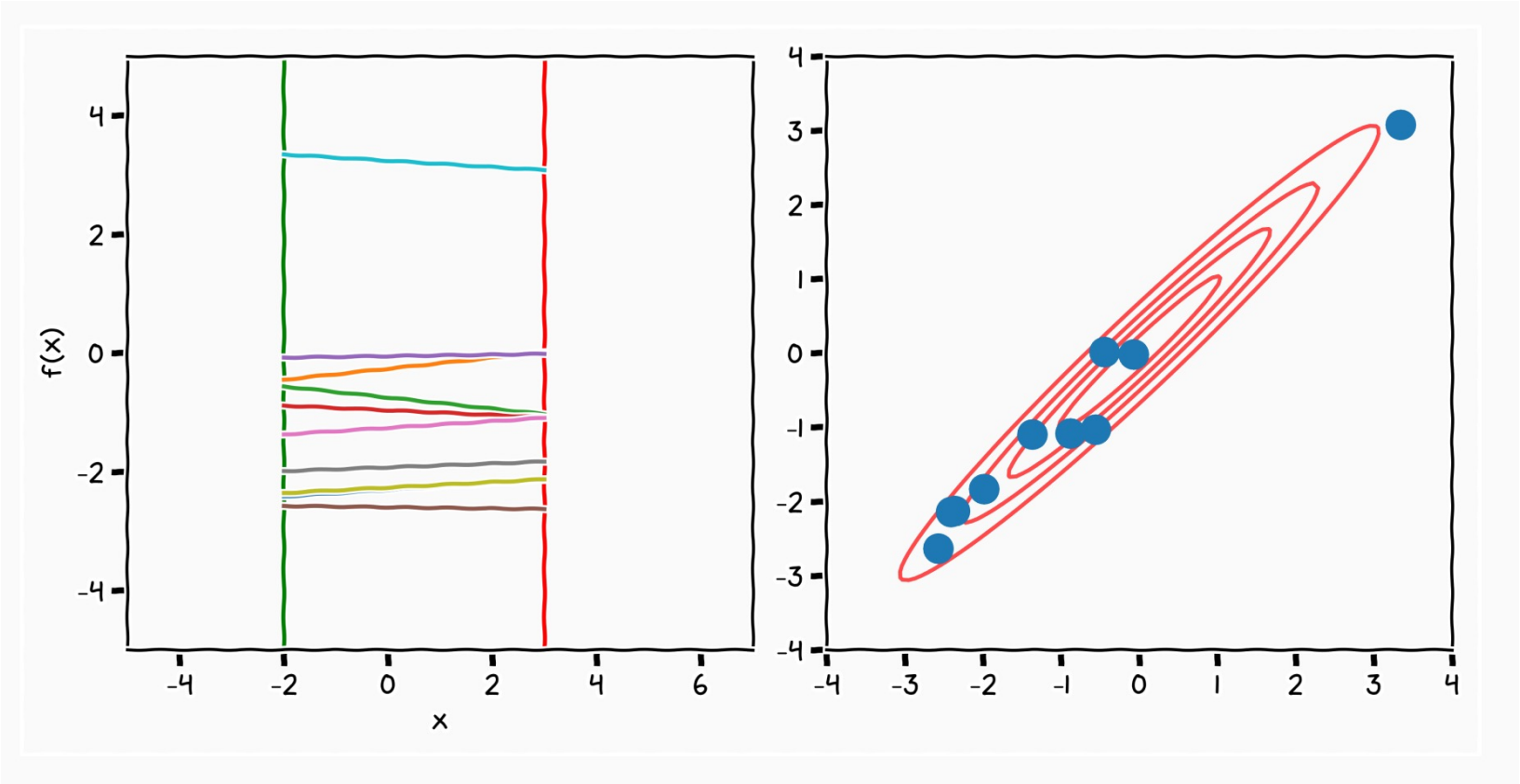
二元高斯分布

- 可以看作是从 Y_1到 Y_2的推断
- 右边的点=左边的线=多次采样



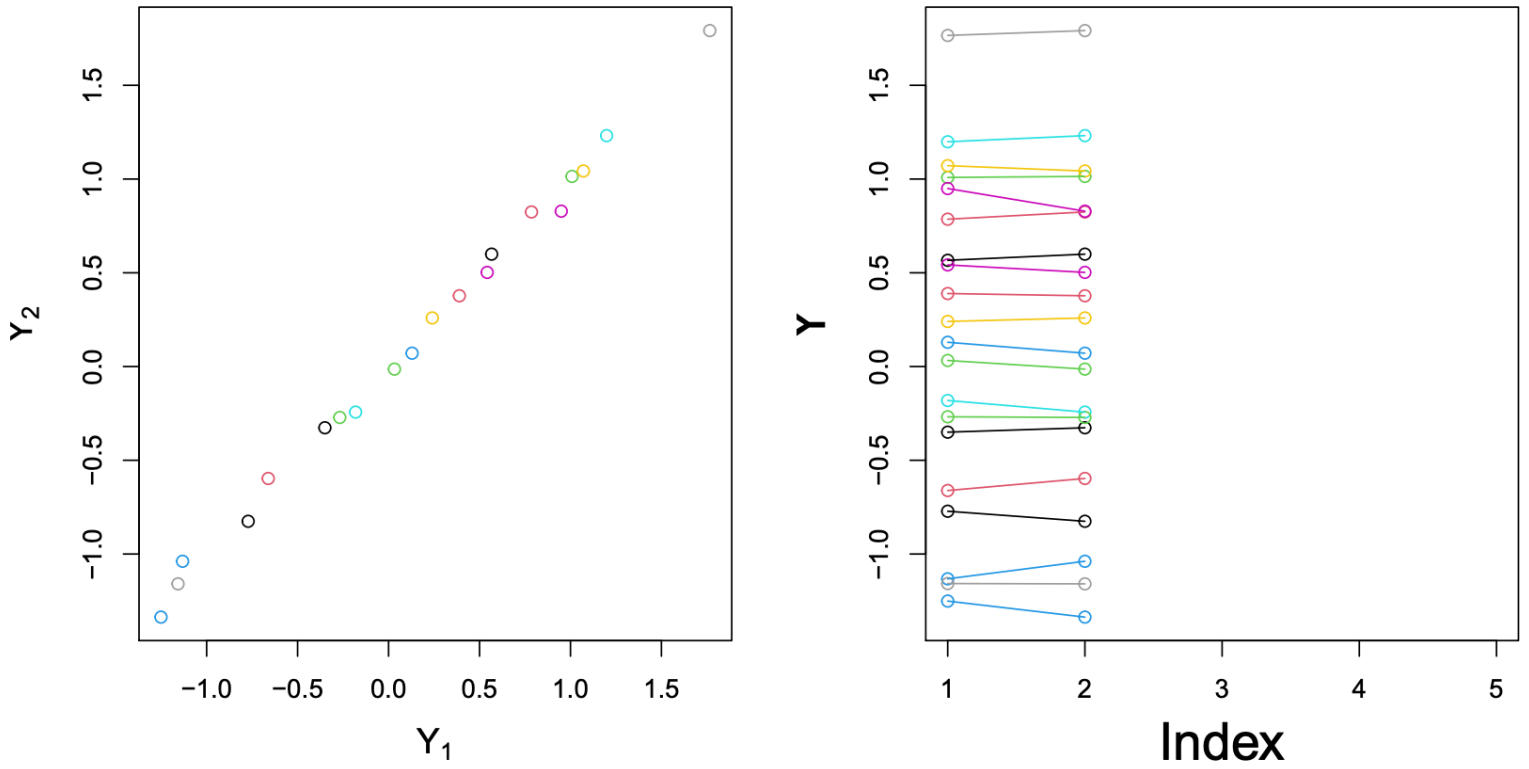
二元高斯分布

- 可以看作是从 Y_1到 Y_2的推断
- 右边的点=左边的线=多次采样，相关



二元高斯分布

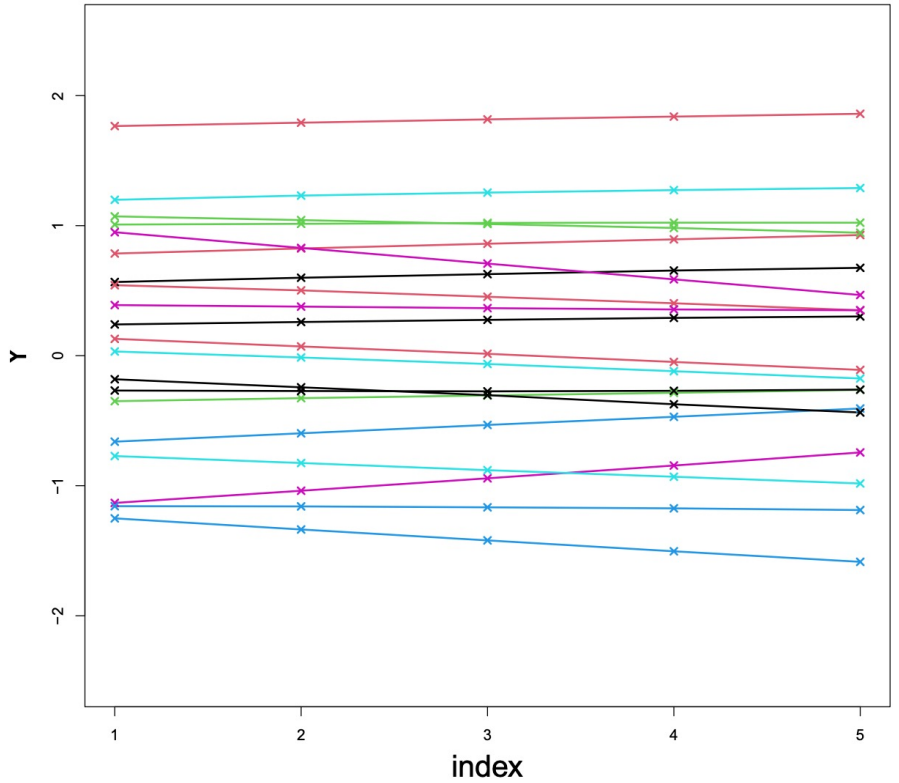
- 可以看作是从 Y_1到 Y_2的推断



多元高斯分布

- d=5

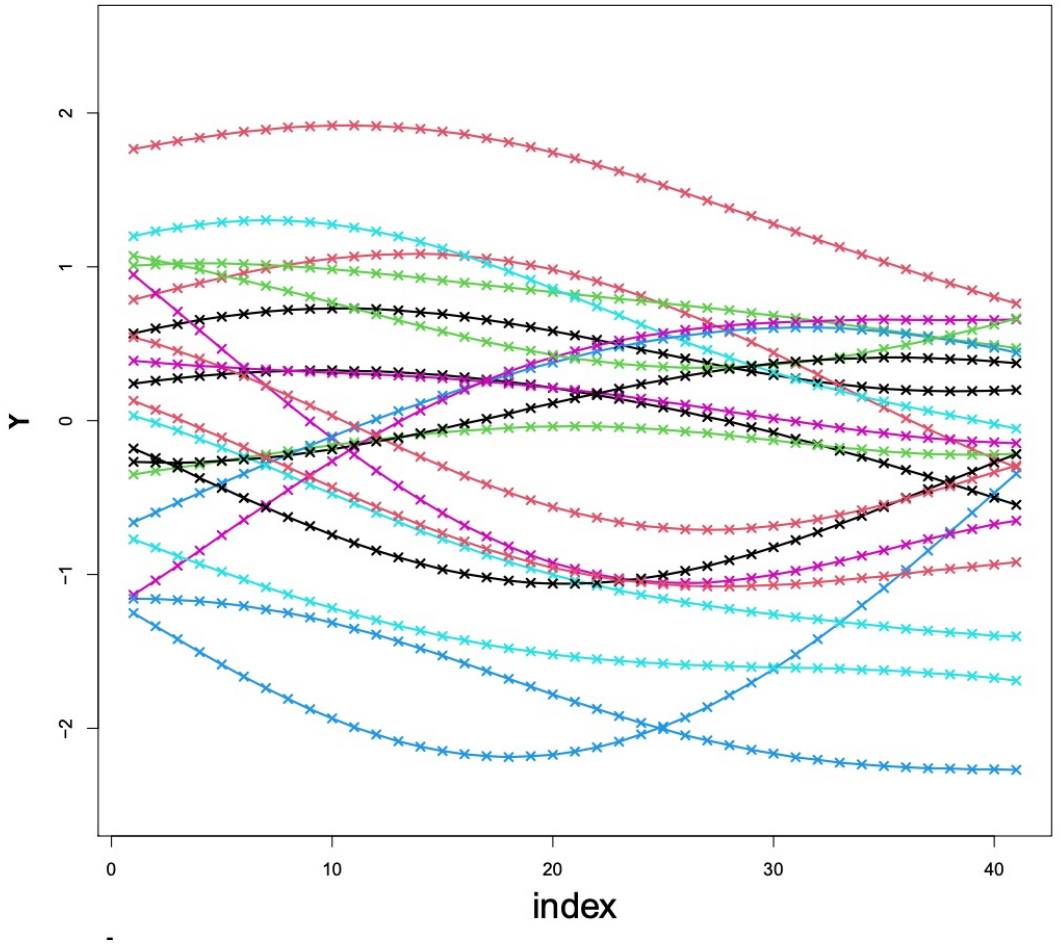
$$\mu = \begin{pmatrix} 0 \\ 0 \\ . \\ . \\ . \\ 0 \end{pmatrix} \qquad \Sigma = \begin{pmatrix} 1 & 0.99 & 0.98 & 0.97 & 0.96 \\ 0.99 & 1 & 0.99 & 0.98 & 0.97 \\ 0.98 & 0.99 & 1 & 0.99 & 0.98 \\ 0.97 & 0.98 & 0.99 & 1 & 0.99 \\ 0.96 & 0.97 & 0.98 & 0.99 & 1 \end{pmatrix}$$



多元高斯分布

- d=50

$$\mu = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \qquad \Sigma = \begin{pmatrix} 1 & 0.99 & 0.98 & 0.97 & 0.96 & \dots \\ 0.99 & 1 & 0.99 & 0.98 & 0.97 & \dots \\ 0.98 & 0.99 & 1 & 0.99 & 0.98 & \dots \\ 0.97 & 0.98 & 0.99 & 1 & 0.99 & \dots \\ 0.96 & 0.97 & 0.98 & 0.99 & 1 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix}$$



高斯过程：函数的高斯分布

- 高斯过程可视为函数的无限维分布

$$p(\mathbf{f}) = \mathcal{N}\left(\begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_N \\ \vdots \end{bmatrix} \middle\| \begin{bmatrix} \mu(x_1) \\ \mu(x_2) \\ \vdots \\ \mu(x_N) \\ \vdots \end{bmatrix}, \begin{bmatrix} k(x_1,x_1) & k(x_1,x_2) & \dots & k(x_1,x_N) & \dots \\ k(x_2,x_1) & k(x_2,x_2) & \dots & k(x_2,x_N) & \dots \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ k(x_N,x_1) & k(x_N,x_2) & \dots & k(x_N,x_N) & \dots \\ \vdots & \vdots & \dots & \vdots & \ddots \end{bmatrix}\right)$$

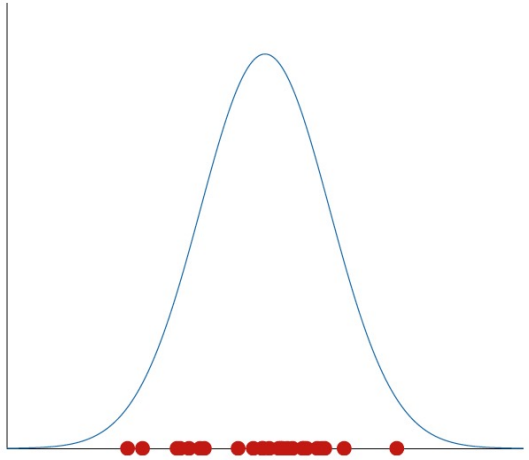
高斯分布

- 高斯分布：需要计算均值（μ）和方差（Σ）

$$X \sim N(\mu, \Sigma)$$

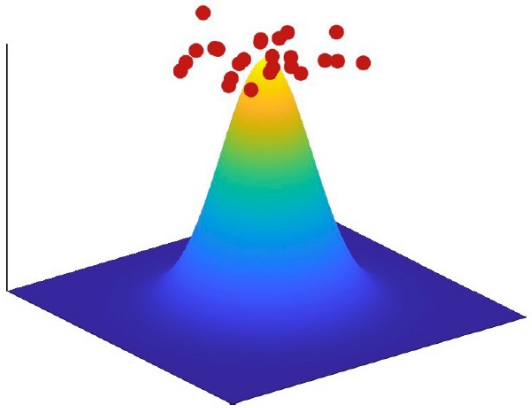
- (Univariate) Gaussians:

$$x \sim \mathcal{N}(x; \mu = 0, \sigma^2 = 1)$$



$$\mu = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} \qquad \Sigma = \begin{pmatrix} \sigma_1^2 & \rho_{12}\sigma_1\sigma_2 \\ \rho_{21}\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix}$$

$$\text{Cor}(Y_i, Y_j) = \rho_{12} \text{ for } i \neq j$$



- Multivariate Gaussians:

$$\boldsymbol{x} = [x_1, \ldots, x_D]^T$$

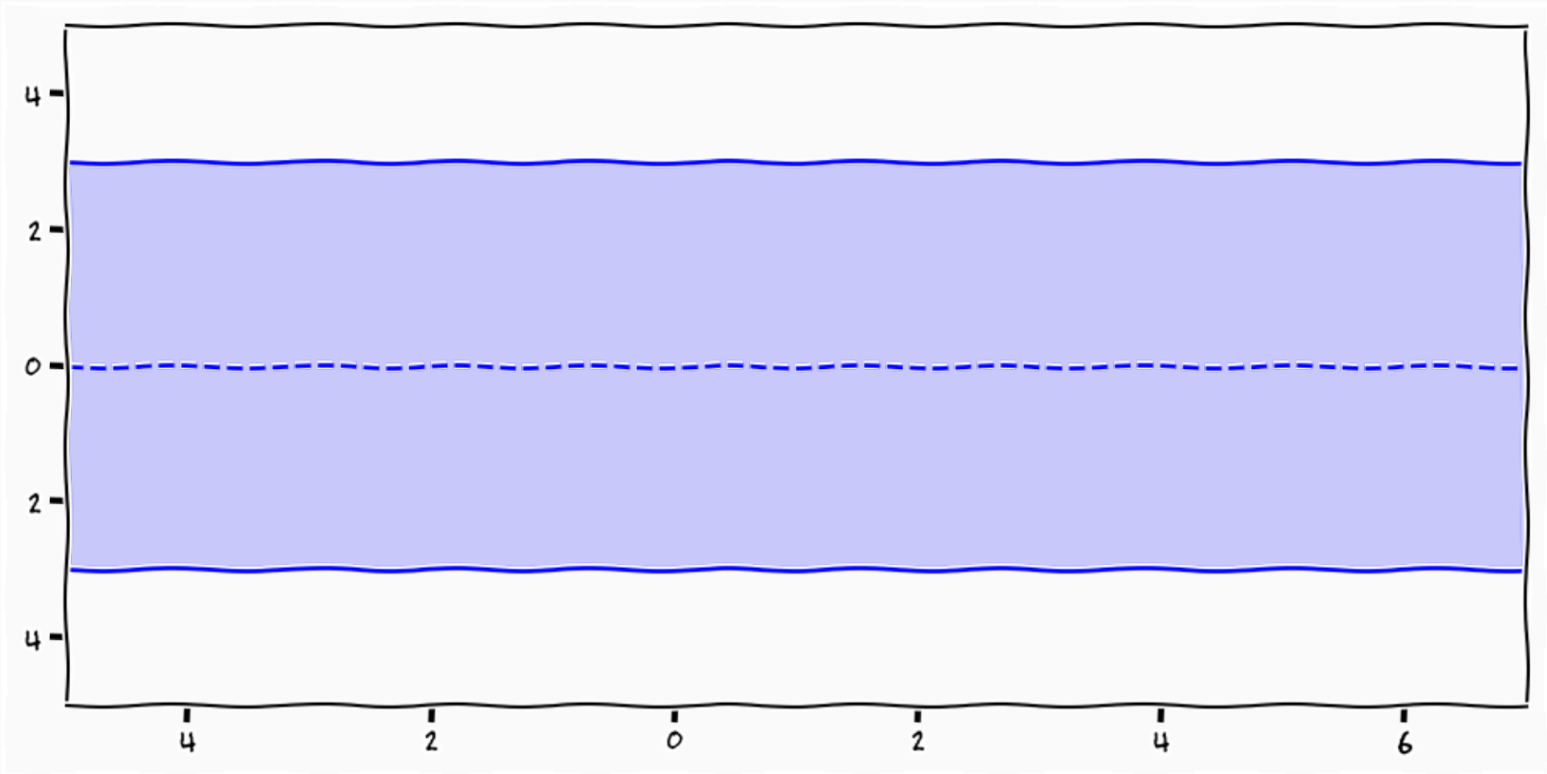
$$\sim \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu} = \mathbf{0}_D, \Sigma = I_D)$$

高斯过程：函数的高斯分布

- 高斯过程：需要计算函数均值（$m(\cdot)$）和函数方差（$k(\cdot,\cdot)$）

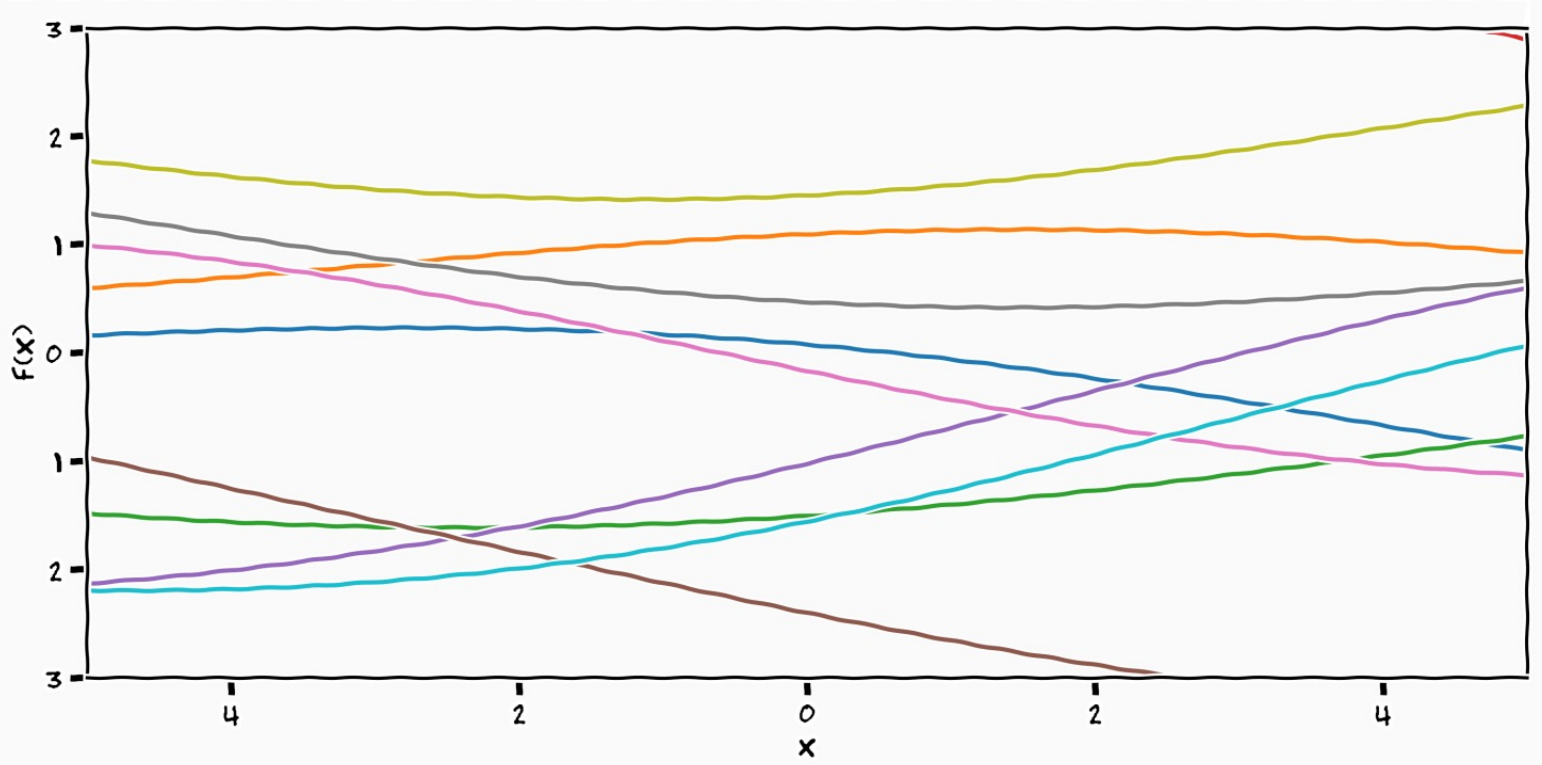
$$y(\cdot) \sim GP(m(\cdot), k(\cdot,\cdot))$$

$$\mathbb{E}(y(x)) = m(x)$$
$$\mathbb{C}\text{ov}(y(x), y(x')) = k(x, x')$$



高斯过程：函数的高斯分布

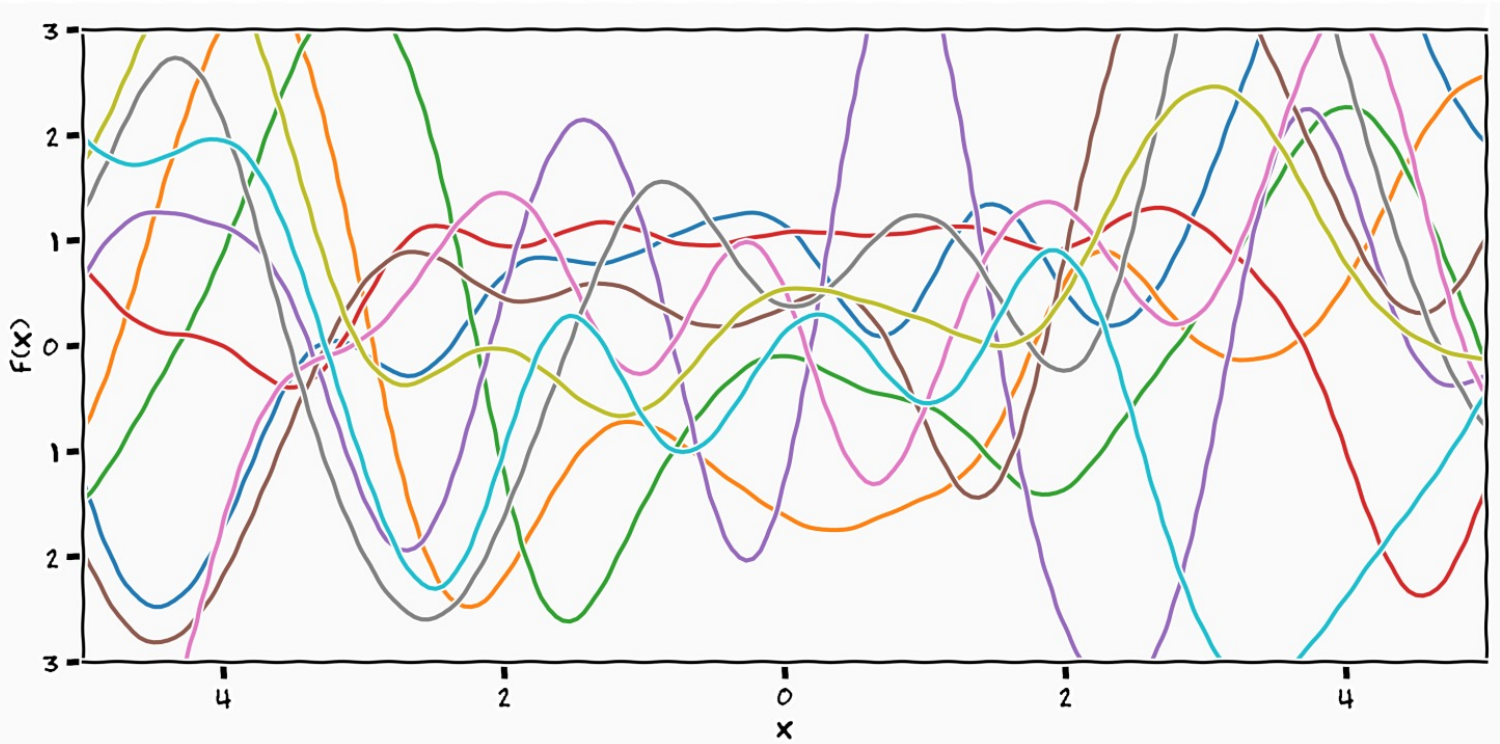
- 每个时间点比较相关，$k(\mathrm{x}, x')$小



$$k(x_i, x_j) = 3 \cdot e^{-\frac{(x_i - x_j)^2}{150}}$$

高斯过程：函数的高斯分布

- 每个时间点不太相关，$k(\mathrm{x}, x')$大

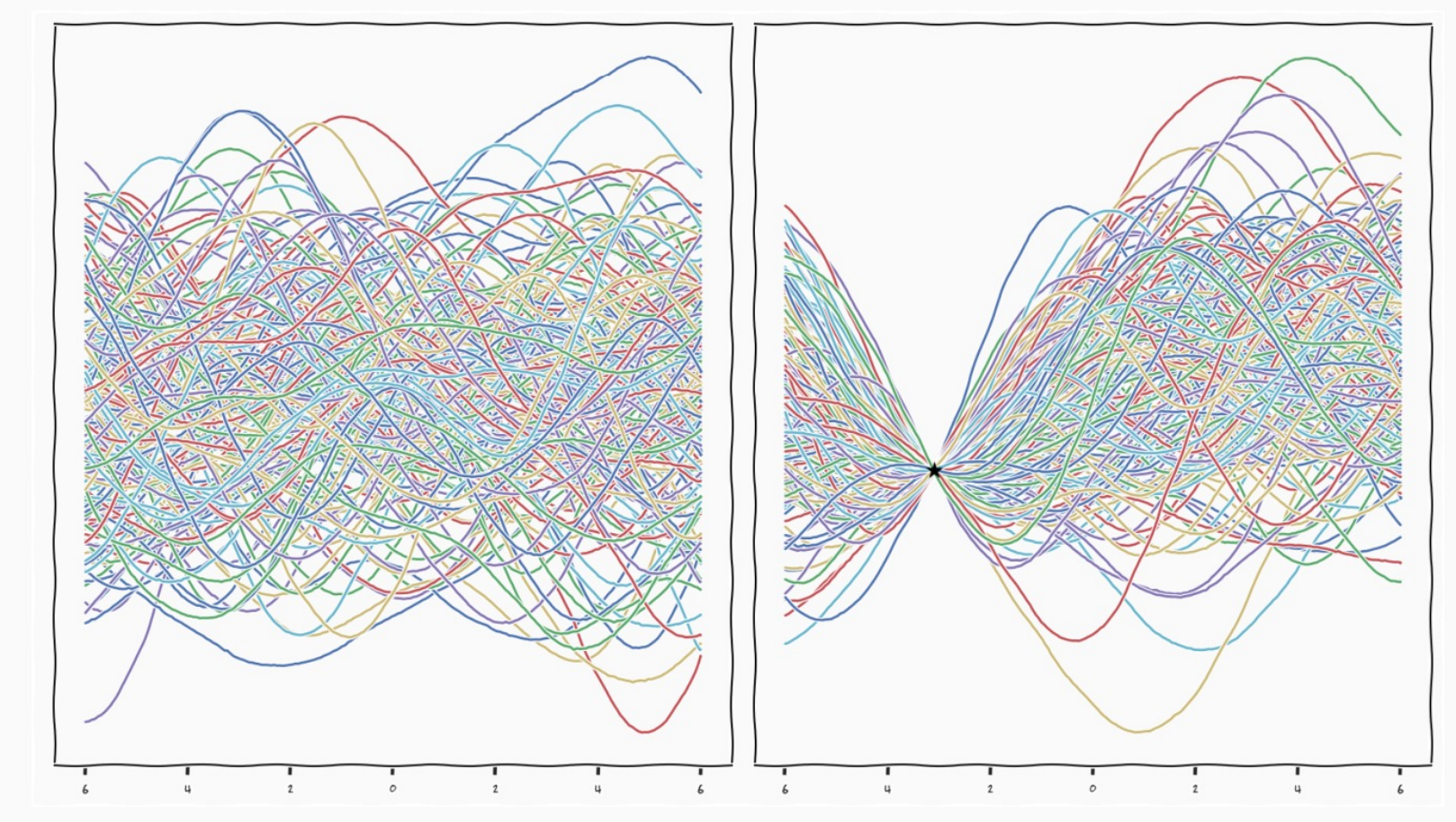


$$k(x_i, x_j) = 3 \cdot e^{-\frac{(x_i - x_j)^2}{1}}$$

高斯过程的后验更新

先验假设

后验假设



二元高斯分布的条件更新

Suppose

$$Y = \begin{pmatrix} Y_1 \\ Y_2 \end{pmatrix} \sim N(\mu, \Sigma)$$

where

$$\mu = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} \qquad \Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix}$$

Then

$$Y_2 \mid Y_1 = y_1 \sim N\left(\mu_2 + \Sigma_{21}\Sigma_{11}^{-1}(y_1 - \mu_1), \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12}\right)$$

高斯过程的条件更新

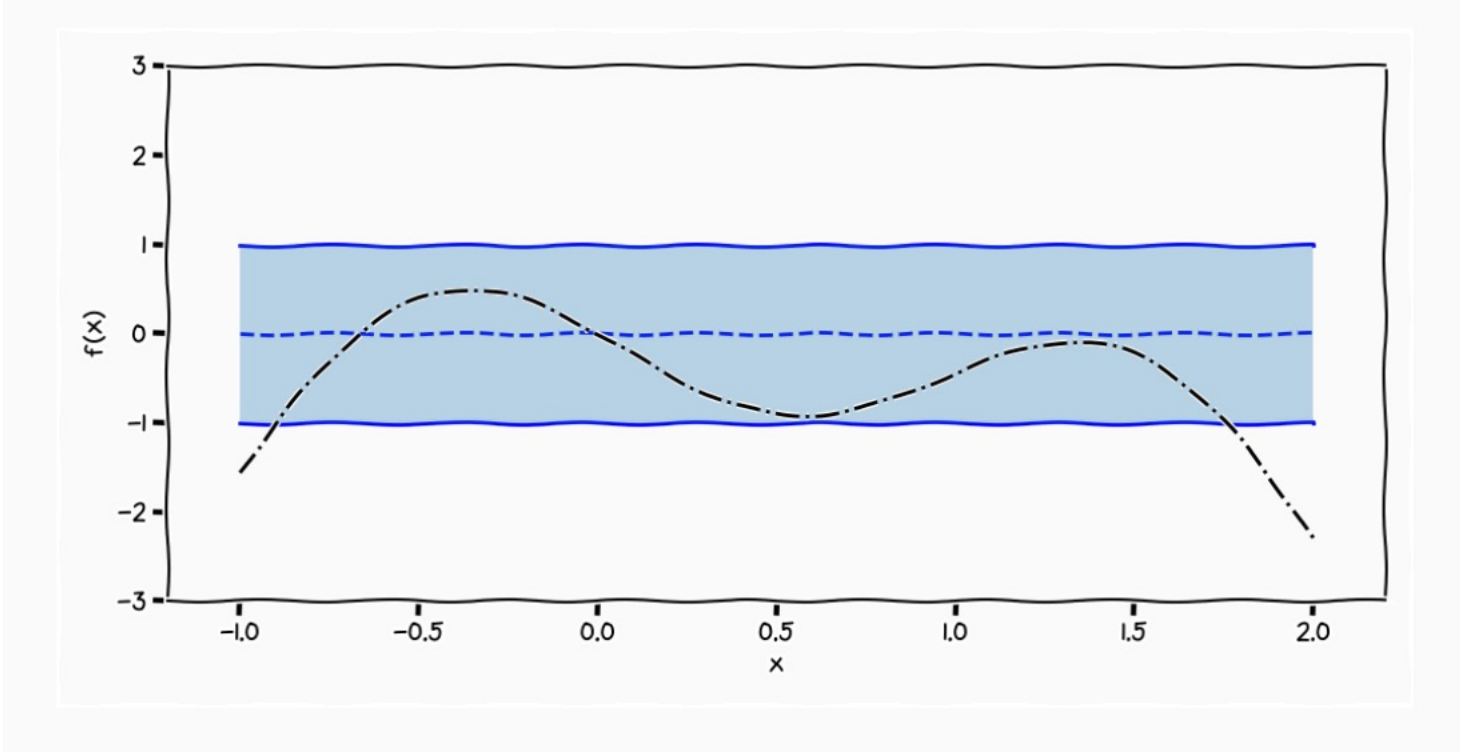
- 利用贝叶斯法则，利用所有在x观测到的f，f_*为未观测的点：

$$p(\mathbf{f}_* \mid \mathbf{f}) = \frac{p(\mathbf{f}, \mathbf{f}_*)}{p(\mathbf{f})} = \frac{p(\mathbf{f}, \mathbf{f}_*)}{\int p(\mathbf{f}, \mathbf{f}_*) \mathrm{d}\mathbf{f}_*}$$

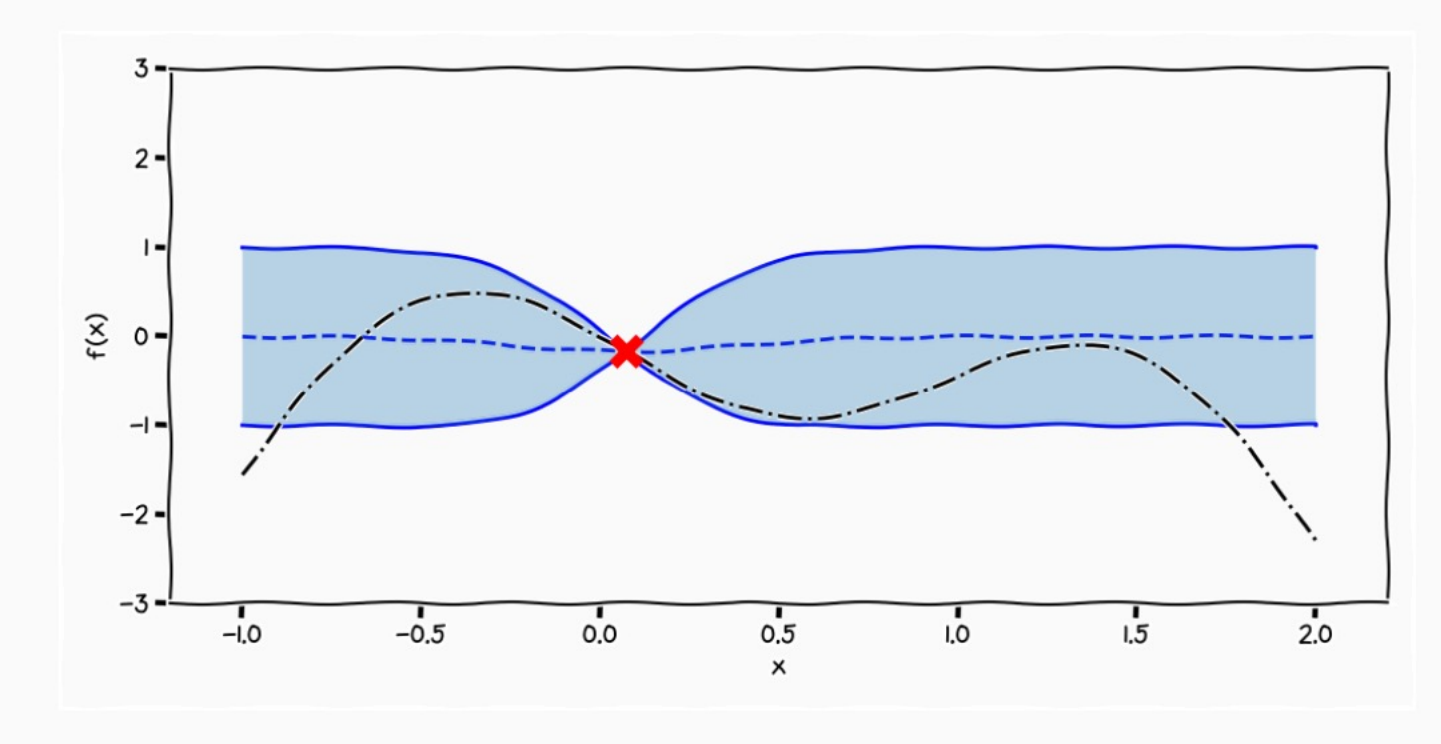
- 在观测f之后，仍然符合高斯假设：

$$p(f_*|\mathbf{x}_*, \mathbf{x}, \mathbf{f}) = \mathcal{N}(k(\mathbf{x}_*, \mathbf{x})^{\mathrm{T}} k(\mathbf{x}, \mathbf{x})^{-1} \mathbf{f}, \\ k(\mathbf{x}_*, \mathbf{x}_*) - k(\mathbf{x}_*, \mathbf{x})^{\mathrm{T}} k(\mathbf{x}, \mathbf{x})^{-1} k(\mathbf{x}, \mathbf{x}_*))$$

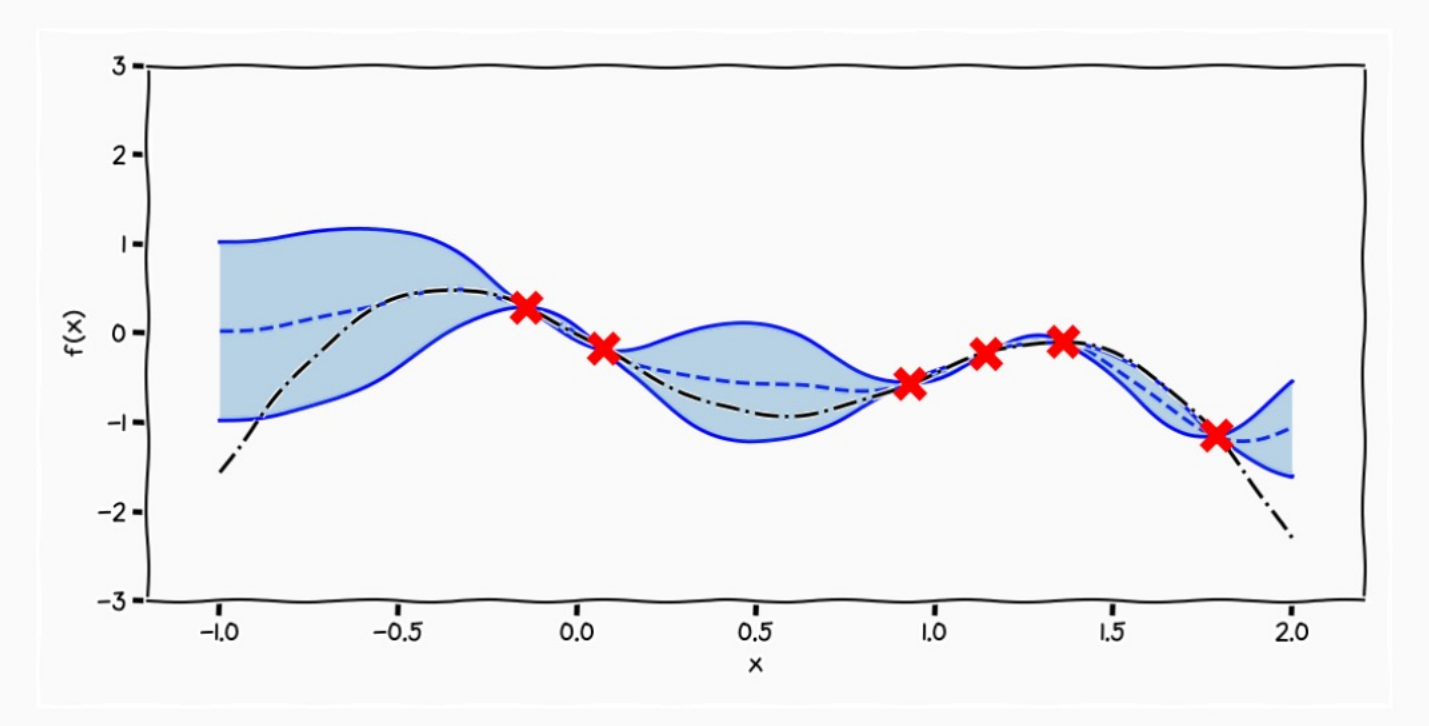
高斯过程的条件更新



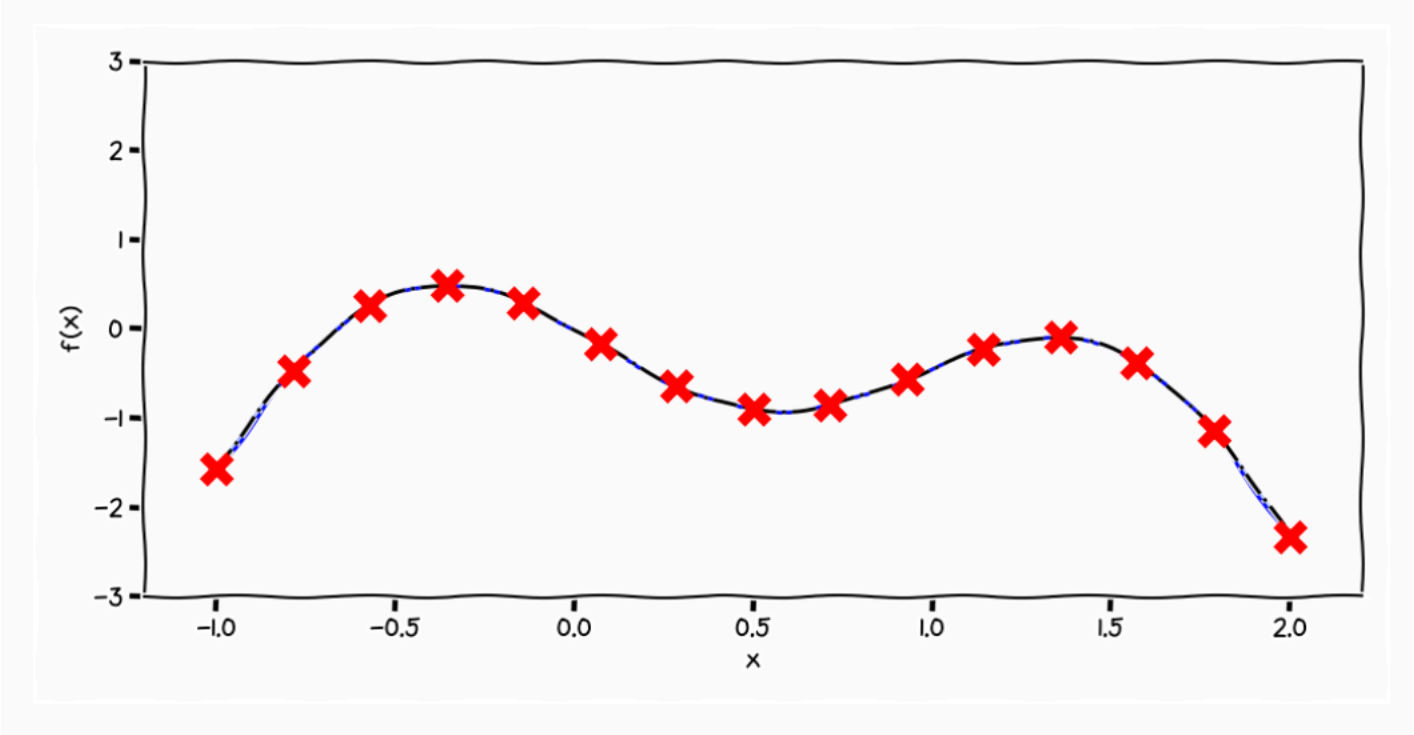
高斯过程的条件更新



高斯过程的条件更新



高斯过程的条件更新



高斯过程的条件更新

- 假设观测也是有噪声的

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} k(\mathbf{x},\mathbf{x}) & k(\mathbf{x},\mathbf{x}_*) \\ k(\mathbf{x}_*,\mathbf{x}) & k(\mathbf{x}_*,\mathbf{x}_*) \end{bmatrix}\right)$$

$$p(f_*|\mathbf{x}_*,\mathbf{x},\mathbf{f}) = \mathcal{N}(k(\mathbf{x}_*,\mathbf{x})^{\mathrm{T}}k(\mathbf{x},\mathbf{x})^{-1}\mathbf{f},\\ k(\mathbf{x}_*,\mathbf{x}_*) - k(\mathbf{x}_*,\mathbf{x})^{\mathrm{T}}k(\mathbf{x},\mathbf{x})^{-1}k(\mathbf{x},\mathbf{x}_*))$$

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} k(\mathbf{x},\mathbf{x}) + \sigma^2\mathbf{I} & k(\mathbf{x},\mathbf{x}_*) \\ k(\mathbf{x}_*,\mathbf{x}) & k(\mathbf{x}_*,\mathbf{x}_*) \end{bmatrix}\right)$$

$$p(f_*|\mathbf{x}_*,\mathbf{x},\mathbf{y},\boldsymbol{\theta}) = \mathcal{N}(k(\mathbf{x}_*,\mathbf{x})^{\mathrm{T}}(K(\mathbf{x},\mathbf{x}) + \sigma^2\mathbf{I})^{-1}\mathbf{y},\\ k(\mathbf{x}_*,\mathbf{x}_*) - k(\mathbf{x}_*,\mathbf{x})^{\mathrm{T}}(K(\mathbf{x},\mathbf{x}) + \sigma^2\mathbf{I})^{-1}K(\mathbf{x},\mathbf{x}_*))$$

高斯过程的条件更新

- 假设观测也是有噪声的（一个超参数）

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} k(\mathbf{x},\mathbf{x}) & k(\mathbf{x},\mathbf{x}_*) \\ k(\mathbf{x}_*,\mathbf{x}) & k(\mathbf{x}_*,\mathbf{x}_*) \end{bmatrix}\right)$$

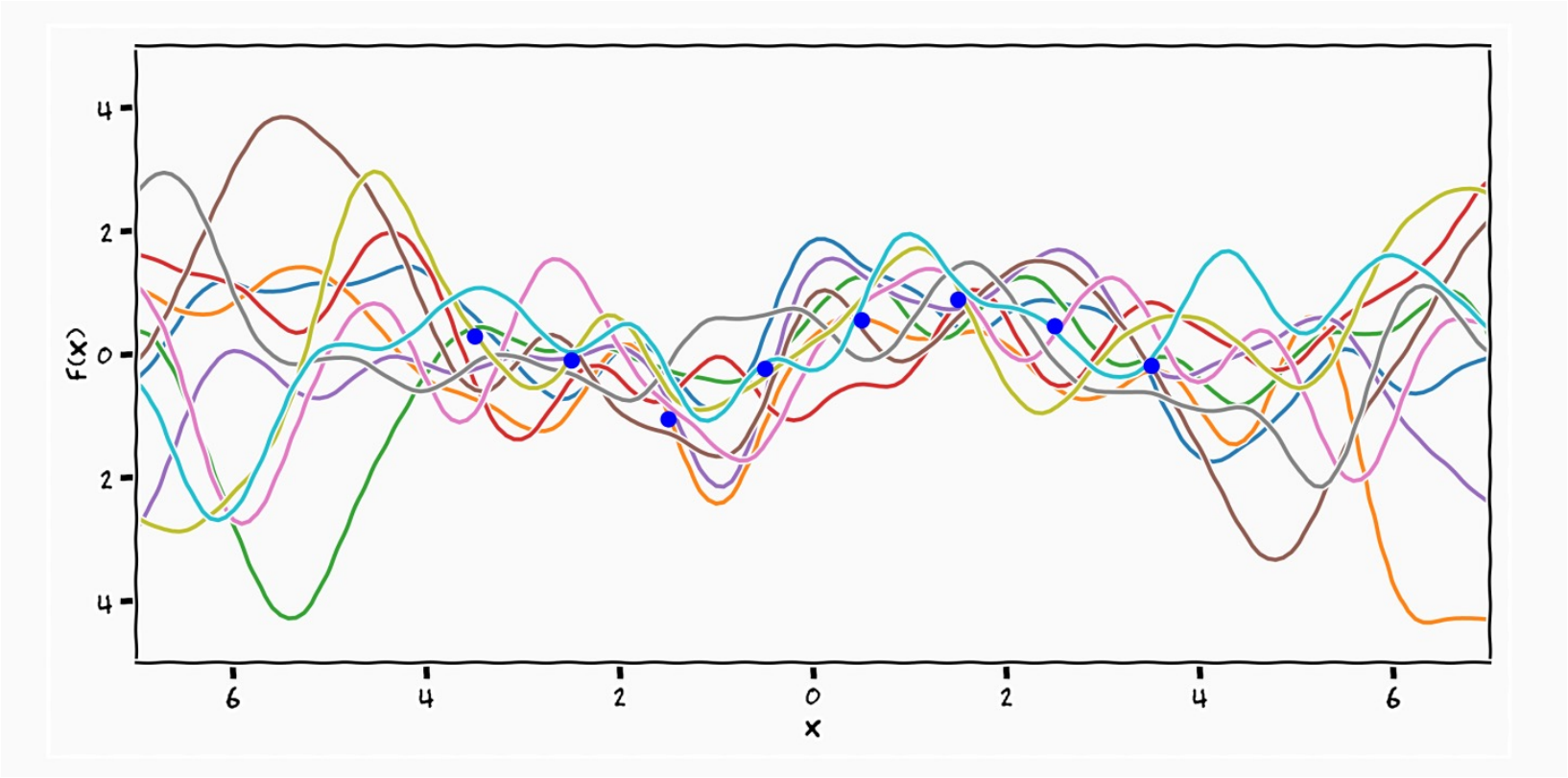
$$p(f_*|\mathbf{x}_*,\mathbf{x},\mathbf{f}) = \mathcal{N}(k(\mathbf{x}_*,\mathbf{x})^{\mathrm{T}}k(\mathbf{x},\mathbf{x})^{-1}\mathbf{f}, \\ k(\mathbf{x}_*,\mathbf{x}_*) - k(\mathbf{x}_*,\mathbf{x})^{\mathrm{T}}k(\mathbf{x},\mathbf{x})^{-1}k(\mathbf{x},\mathbf{x}_*)$$

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} k(\mathbf{x},\mathbf{x}) + \sigma^2\mathbf{I} & k(\mathbf{x},\mathbf{x}_*) \\ k(\mathbf{x}_*,\mathbf{x}) & k(\mathbf{x}_*,\mathbf{x}_*) \end{bmatrix}\right)$$

$$p(f_*|\mathbf{x}_*,\mathbf{x},\mathbf{y},\boldsymbol{\theta}) = \mathcal{N}(k(\mathbf{x}_*,\mathbf{x})^{\mathrm{T}}(K(\mathbf{x},\mathbf{x}) + \sigma^2\mathbf{I})^{-1}\mathbf{y}, \\ k(\mathbf{x}_*,\mathbf{x}_*) - k(\mathbf{x}_*,\mathbf{x})^{\mathrm{T}}(K(\mathbf{x},\mathbf{x}) + \sigma^2\mathbf{I})^{-1}K(\mathbf{x},\mathbf{x}_*))$$

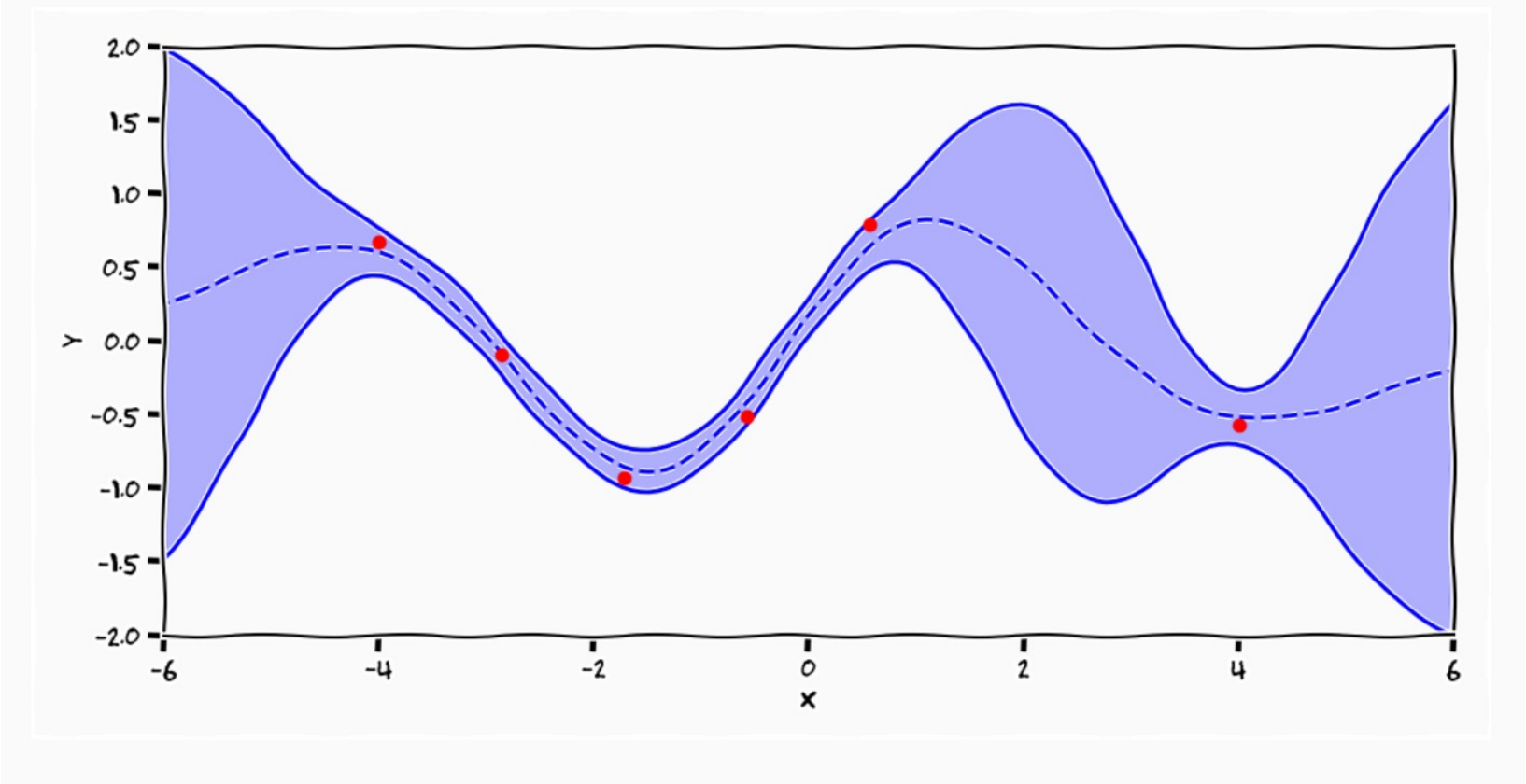
高斯过程的条件更新

- 假设观测也是有噪声的



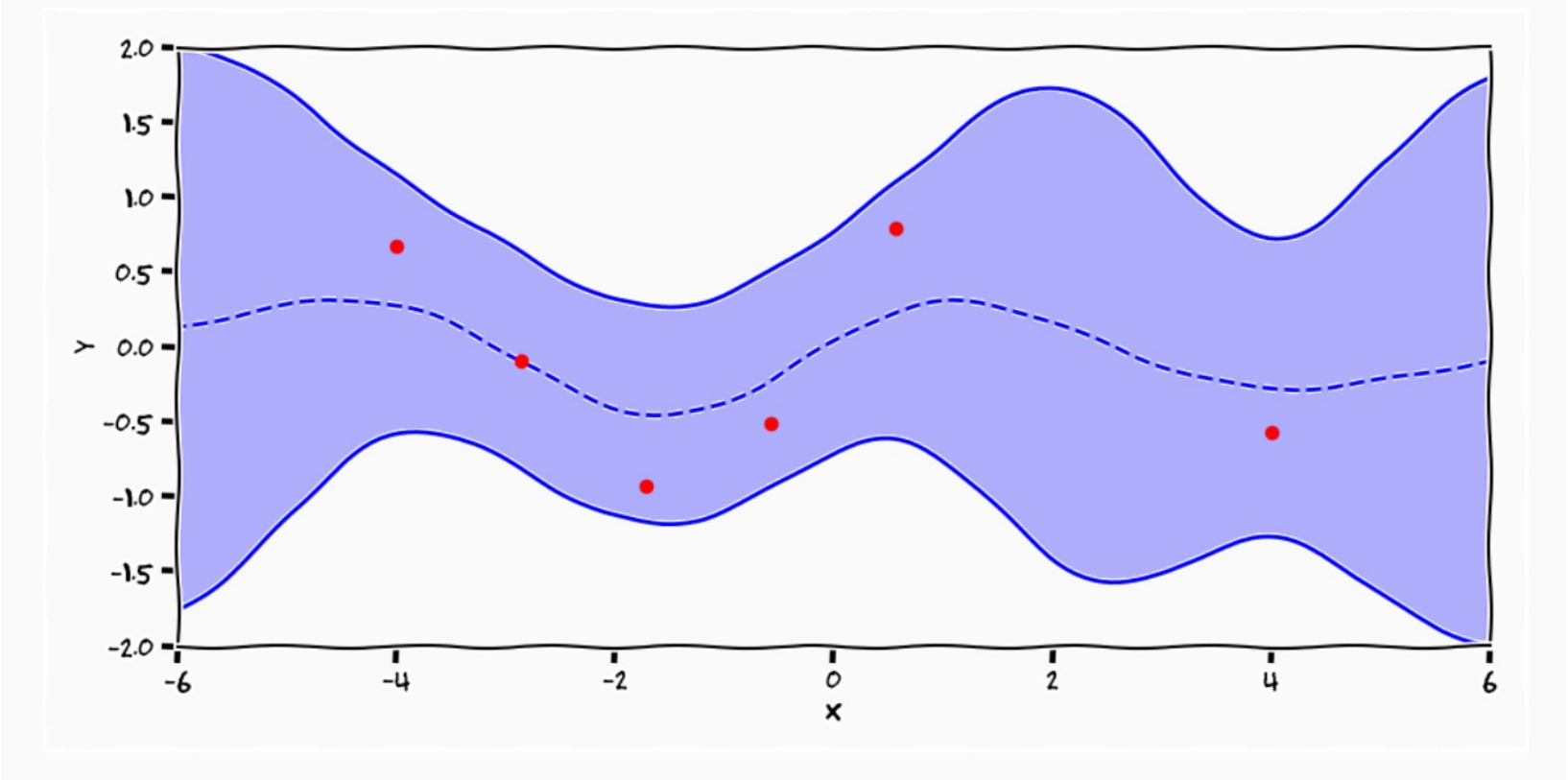
高斯过程的条件更新

- 噪声比较小时



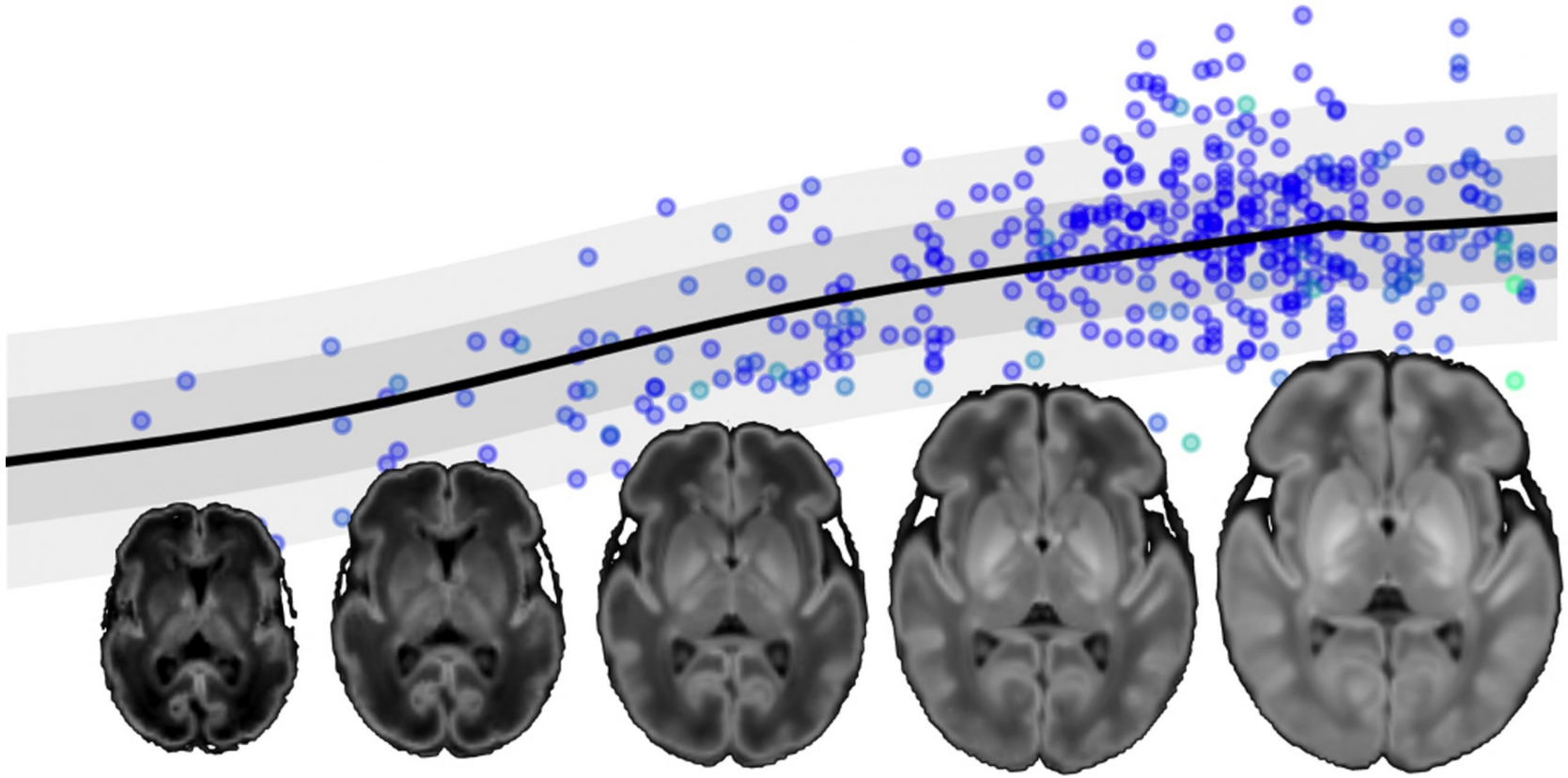
高斯过程的条件更新

- 噪声比较大时



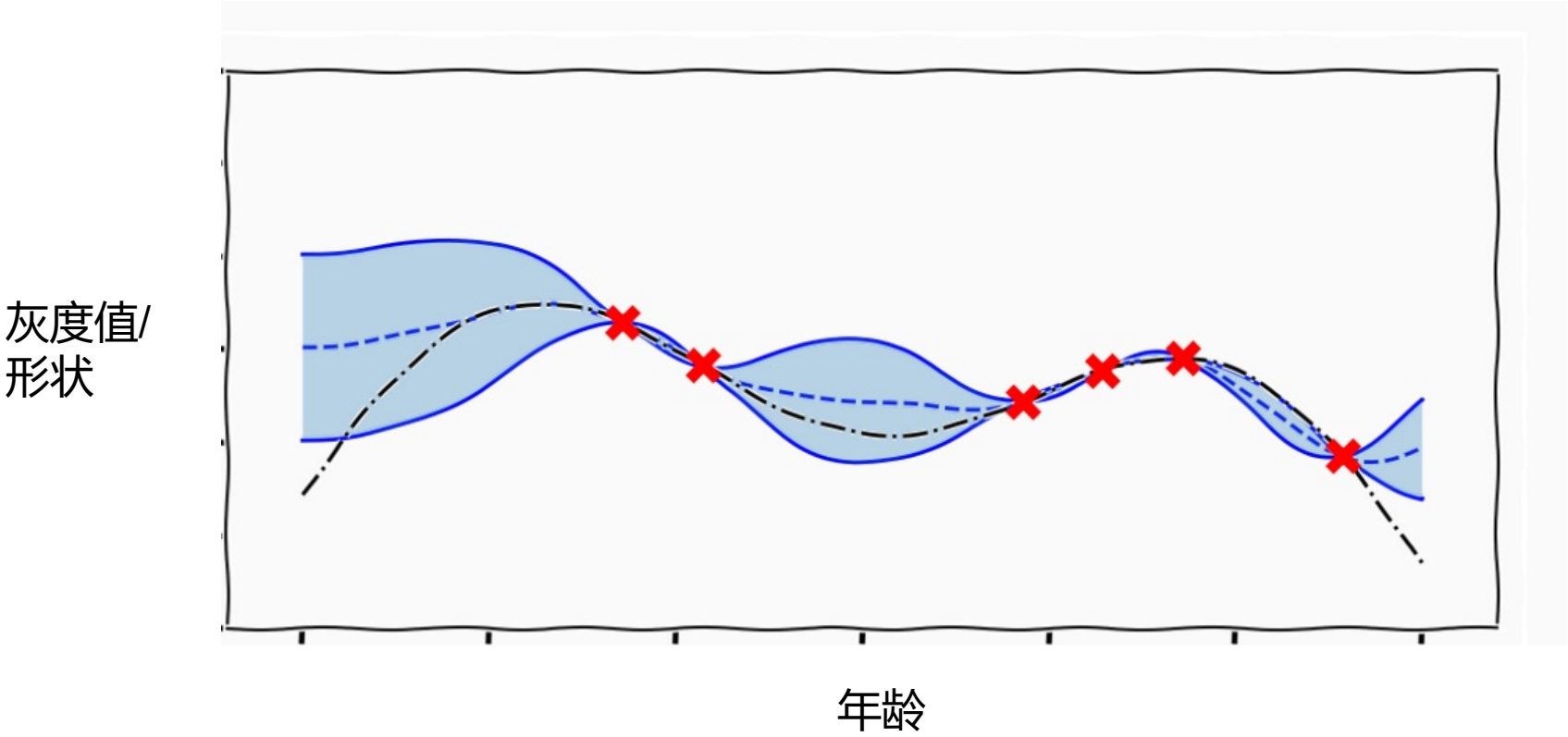
高斯过程的应用

- 非常适合时序、空间连续建模



高斯过程的应用

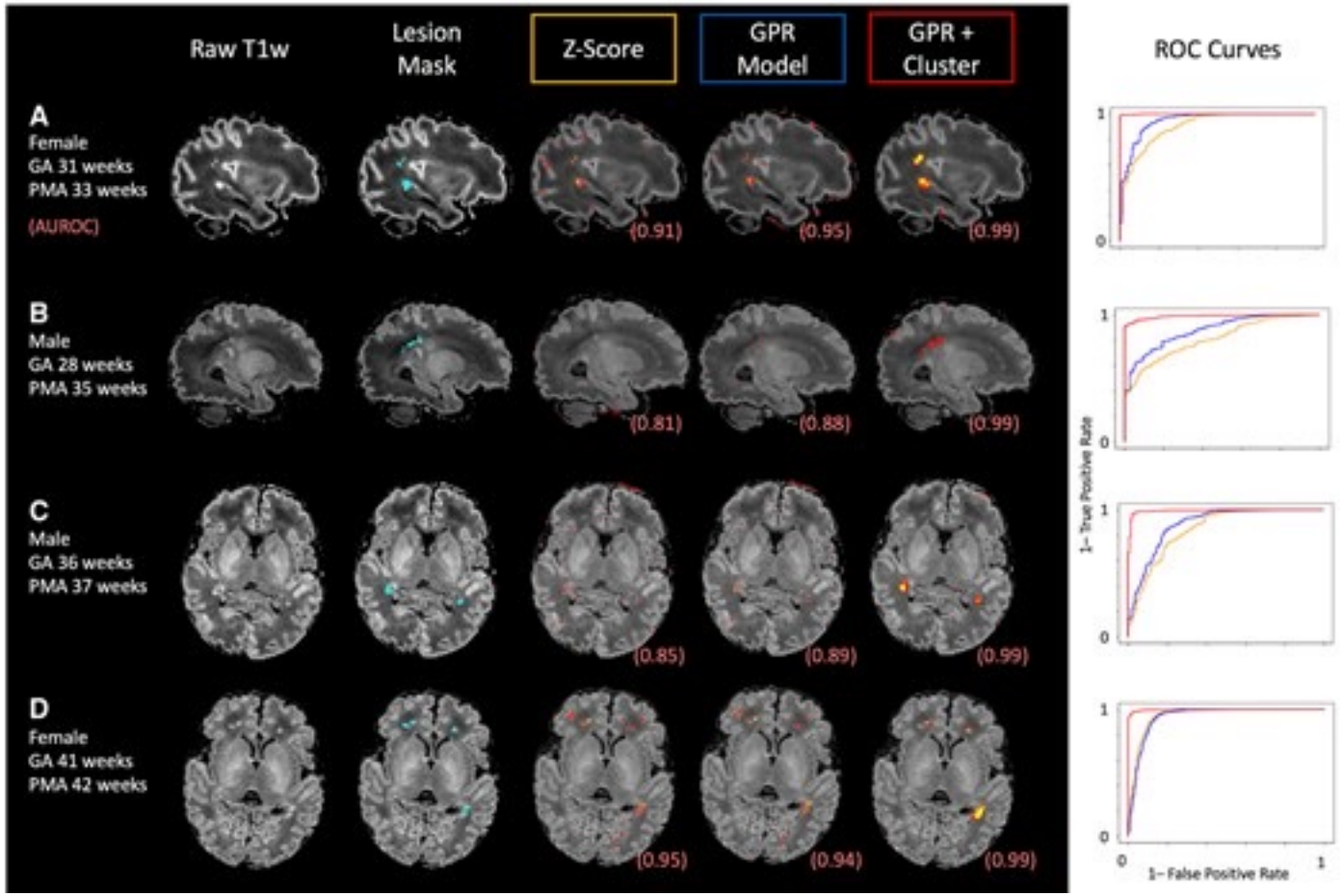
- 利用高斯过程来对脑成长过程建模



Modelling brain development to detect white matter injury in term and preterm born neonates

高斯过程的应用

- 与真实结果对比，检测白质病变

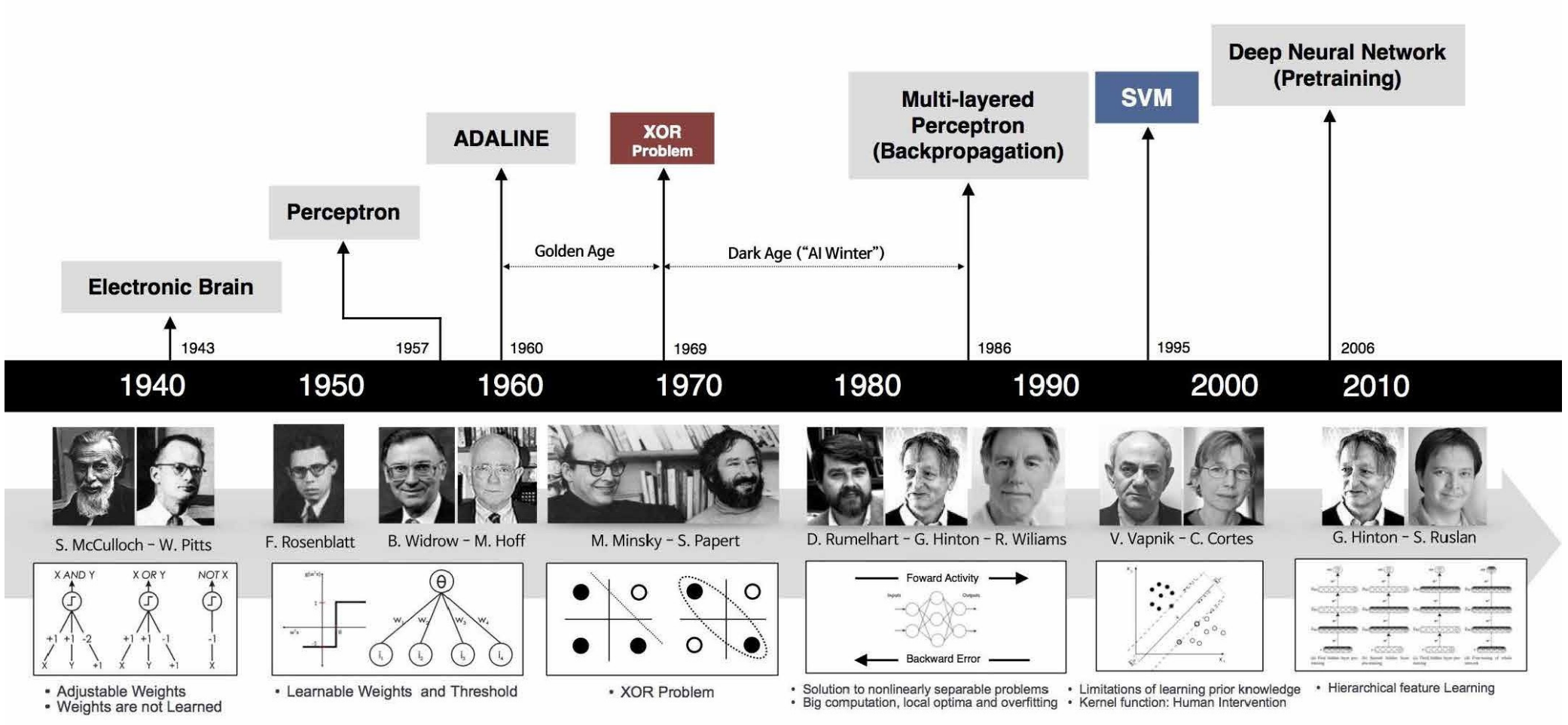


Modelling brain development to detect white matter injury in term and preterm born neonates

目录

神经网络

- 人工神经网络就是AI的历史

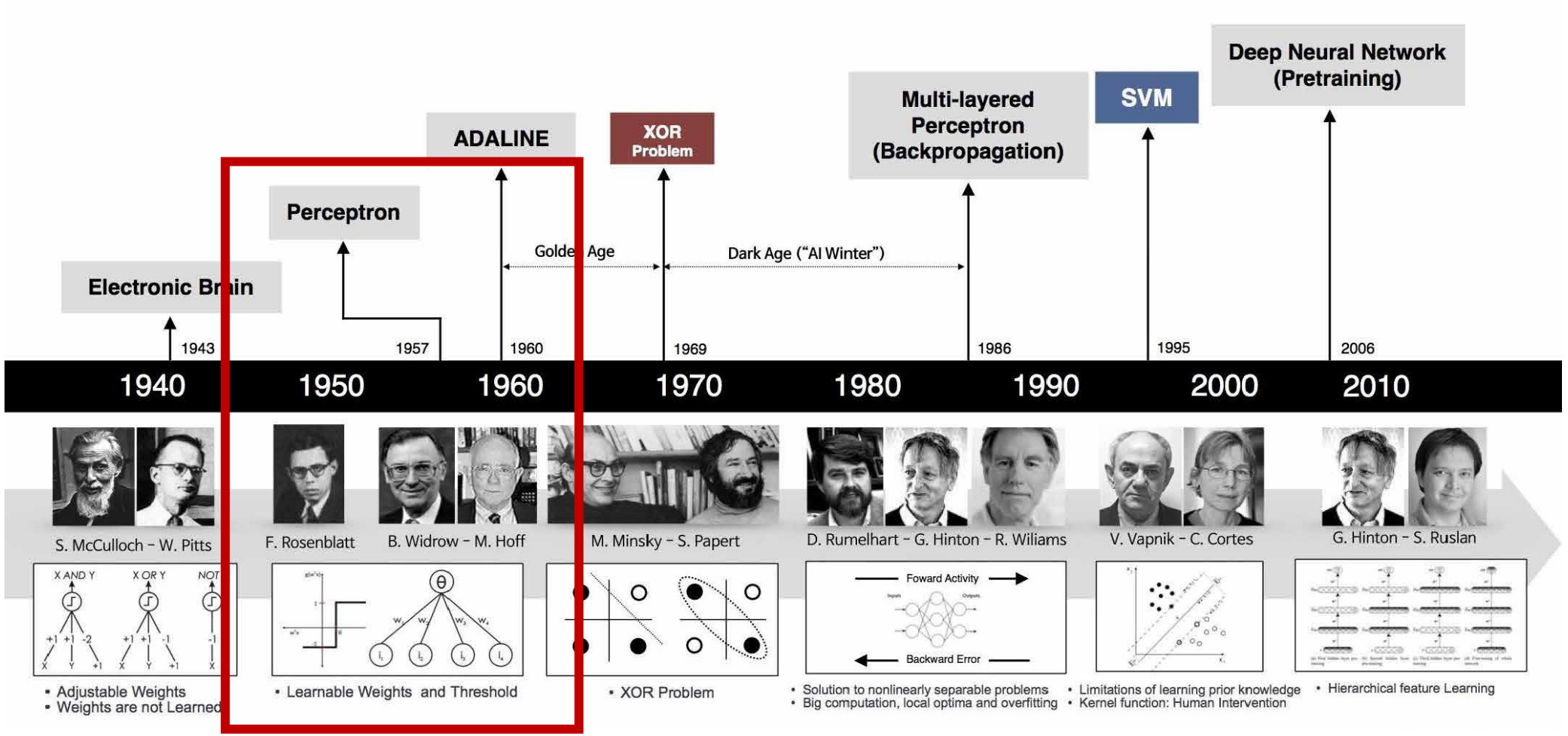


神经网络

- 生物神经网络（Biological Neural Network）
- 神经元（Neuron）：主体
- 轴突（Axon）：传递冲动
- 突触（Synapse）：神经元之间传递冲动
- 树突（Dendrite）：接受冲动

神经网络

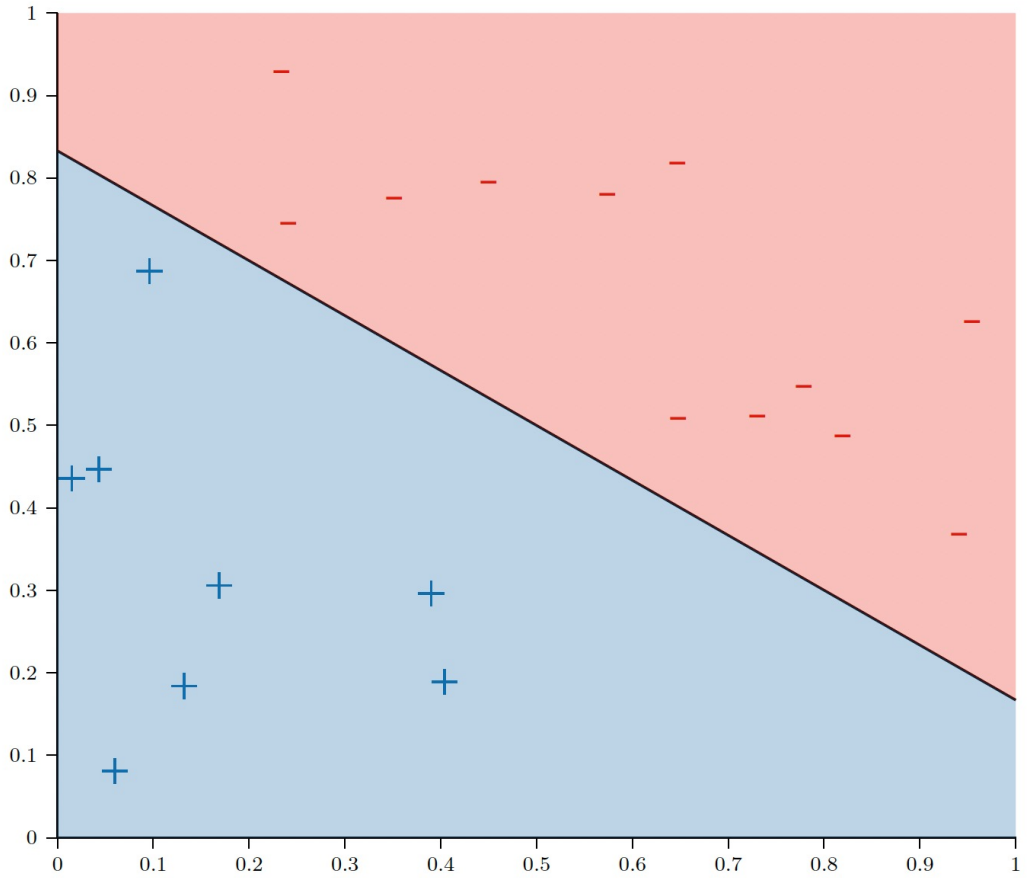
- 人工神经网络就是AI的历史



线性模型

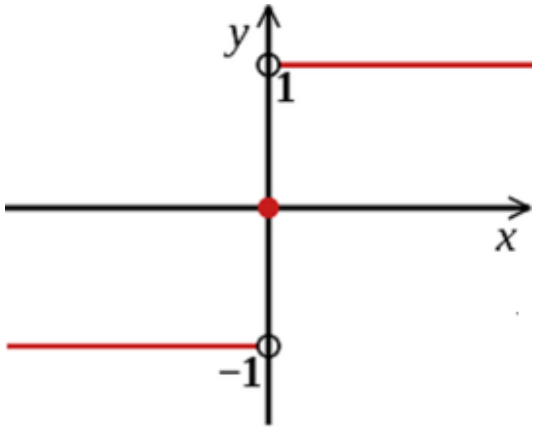
- 基础线性模型

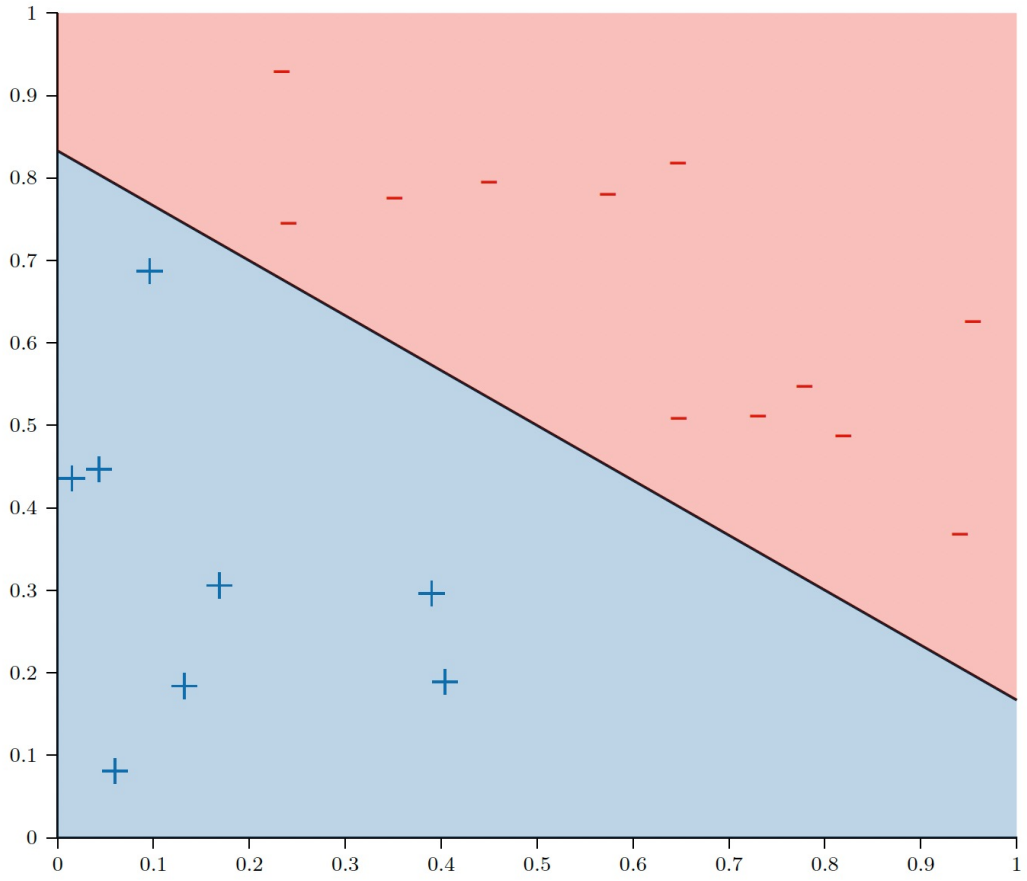
- $w^T x = b$



感知器

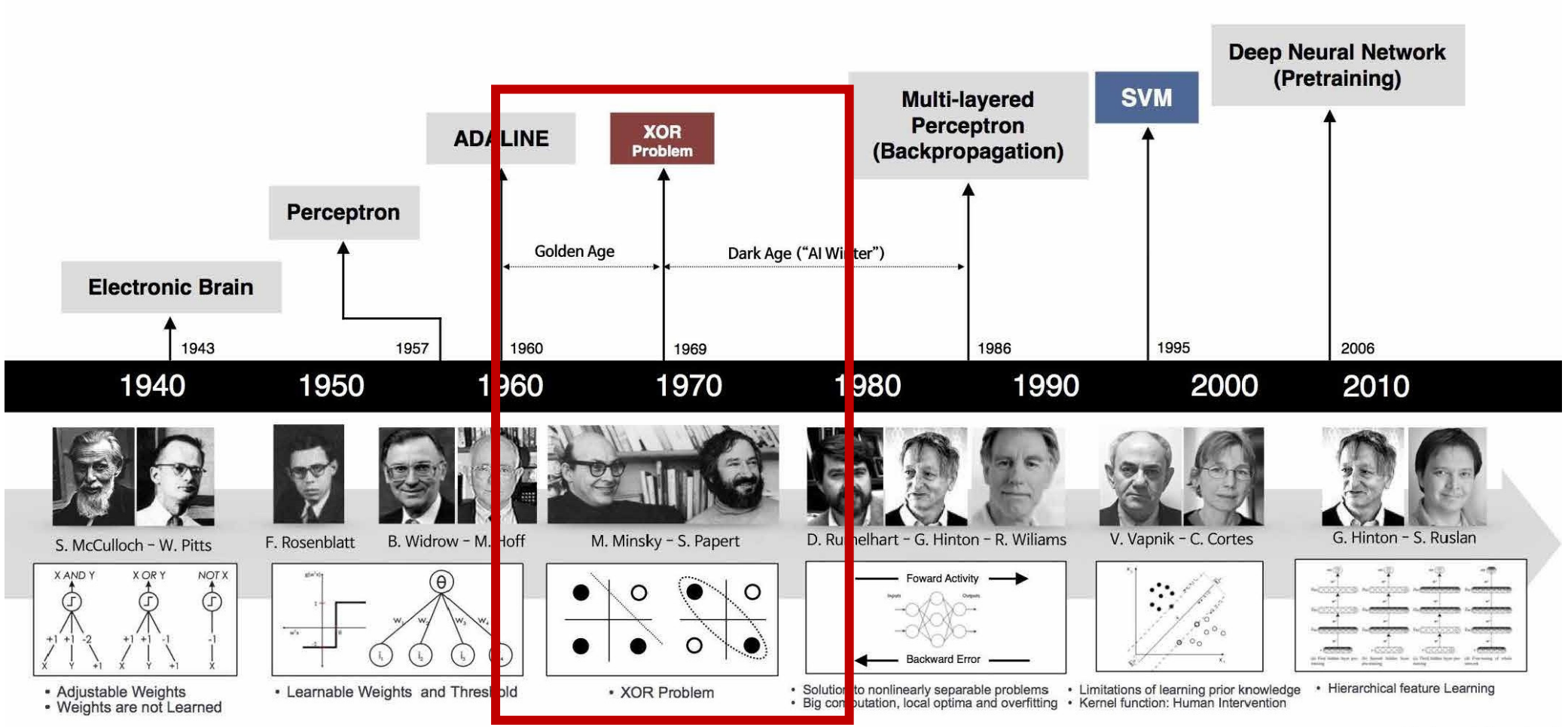
- 决策边界为$h(x) = sign(w^T x + b)$
- 感知器（Perception）
- $sign(x)$





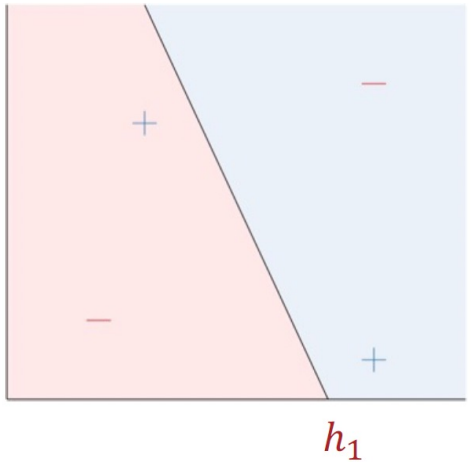
神经网络

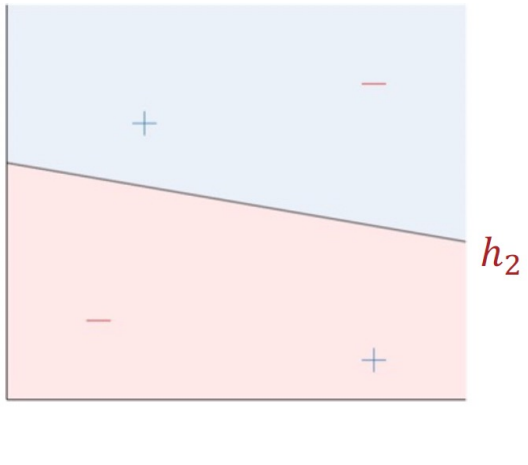
- 人工神经网络就是AI的历史

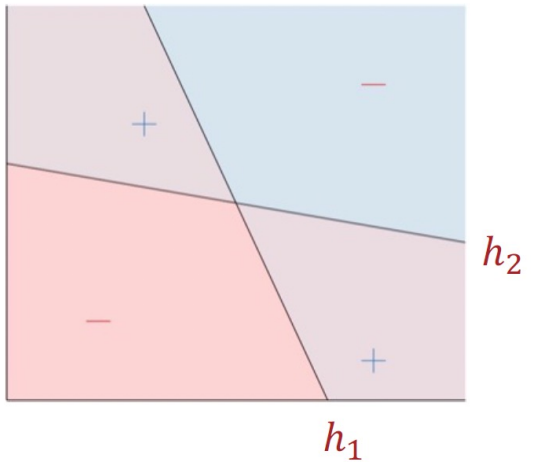


神经网络

- 多个线性模型



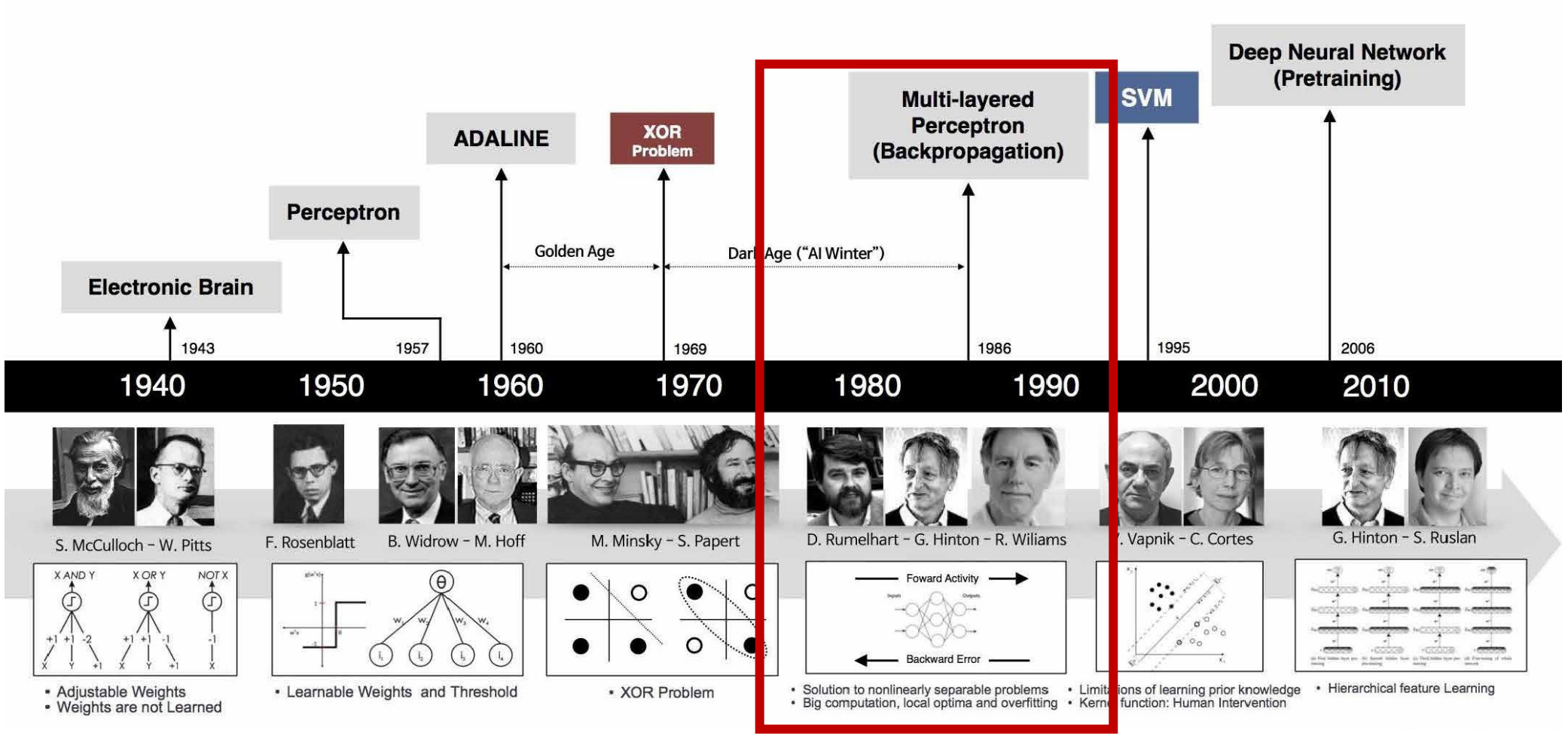




$$h(\boldsymbol{x}) = \begin{cases} +1 \text{ if } (h_1(\boldsymbol{x}) = +1 \text{ and } h_2(\boldsymbol{x}) = -1) \text{ or } (h_1(\boldsymbol{x}) = -1 \text{ and } h_2(\boldsymbol{x}) = +1) \\ -1 \text{ otherwise} \end{cases}$$

神经网络

- 人工神经网络就是AI的历史

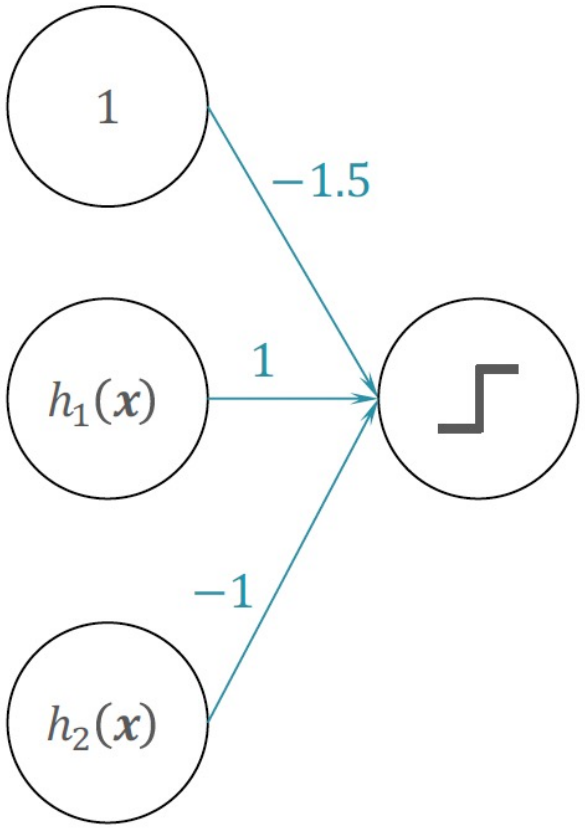


多层感知器 Multi-Layer Perceptron (MLP)

$$h(\boldsymbol{x}) = OR\left(AND\left(h_1(\boldsymbol{x}), \neg h_2(\boldsymbol{x})\right), AND\left(\neg h_1(\boldsymbol{x}), h_2(\boldsymbol{x})\right)\right)$$

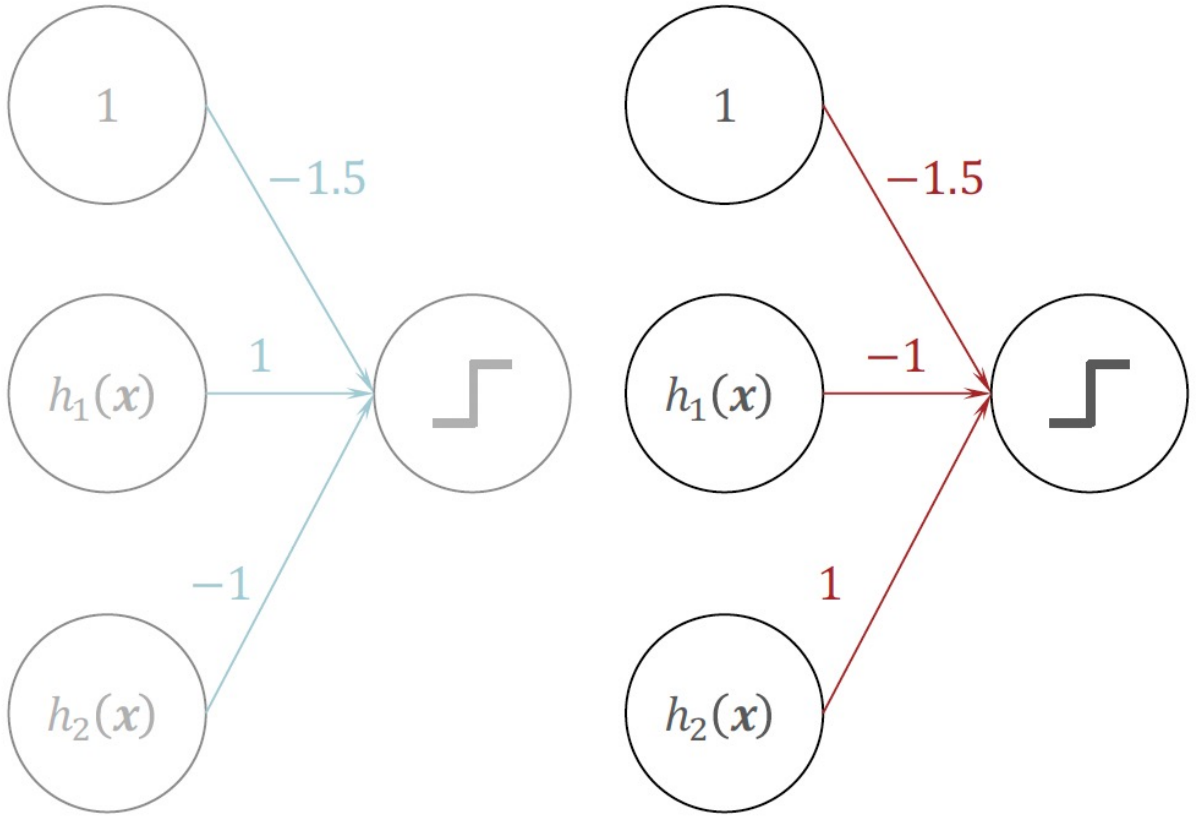
多层感知器 Multi-Layer Perceptron (MLP)

$$h(x) = OR\left(AND\left(h_1(\boldsymbol{x}), \neg h_2(\boldsymbol{x})\right), AND\left(\neg h_1(x), h_2(\boldsymbol{x})\right)\right)$$



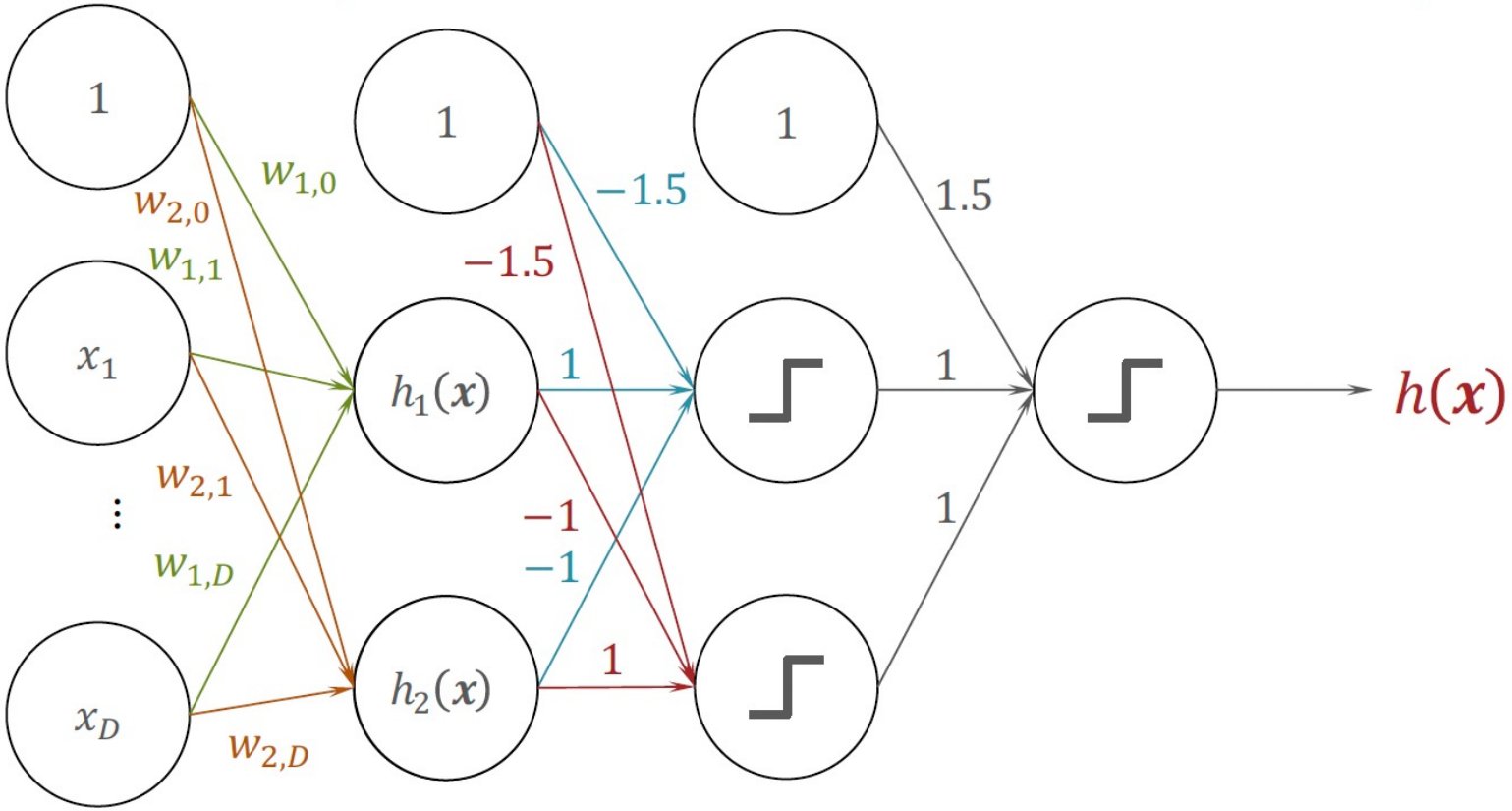
多层感知器 Multi-Layer Perceptron (MLP)

$$h(\boldsymbol{x}) = OR\left(AND\left(h_1(\boldsymbol{x}), \neg h_2(\boldsymbol{x})\right), AND\left(\neg h_1(\boldsymbol{x}), h_2(\boldsymbol{x})\right)\right)$$



多层感知器 Multi-Layer Perceptron (MLP)

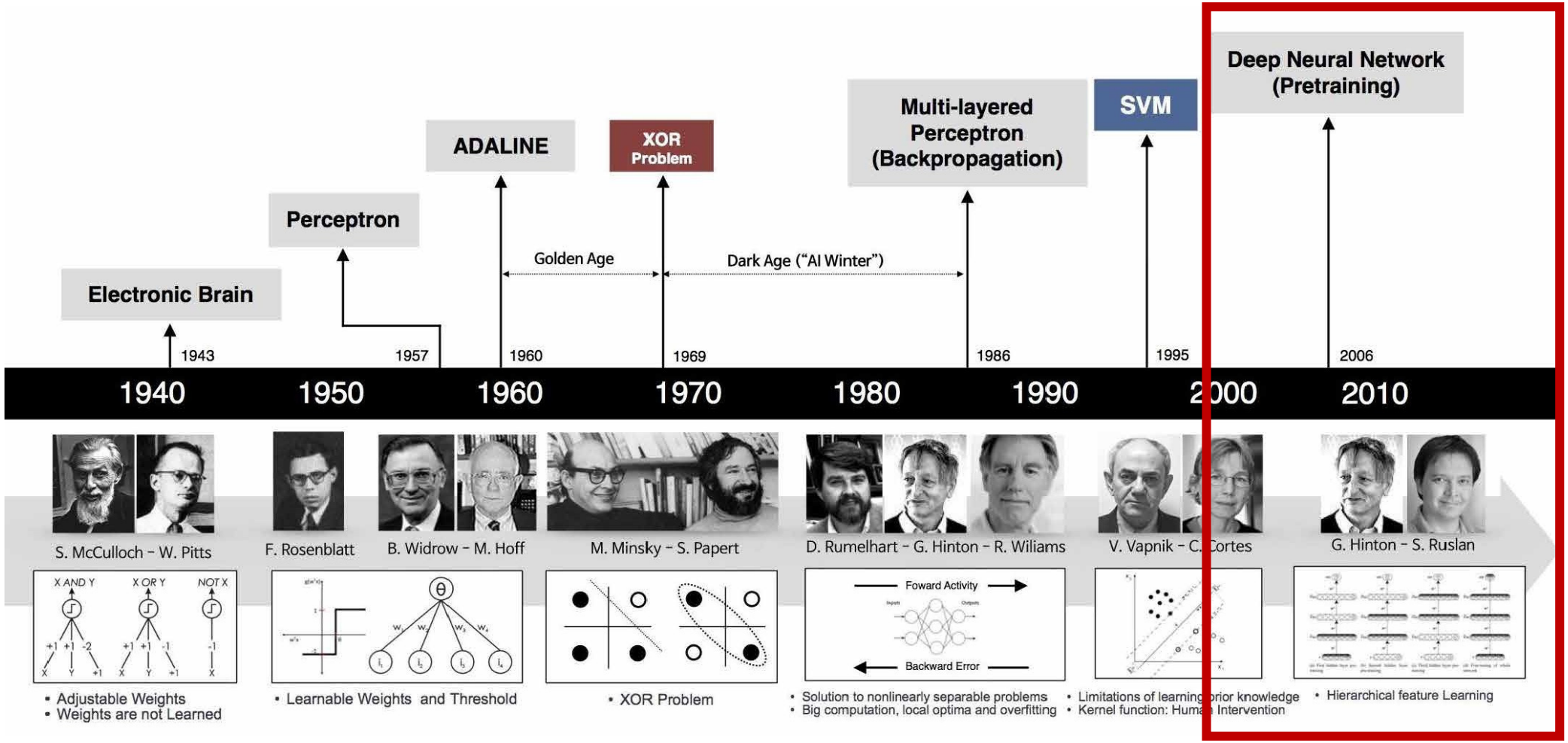
$$h(\boldsymbol{x}) = OR\left(AND\left(h_1(\boldsymbol{x}), \neg h_2(\boldsymbol{x})\right), AND\left(\neg h_1(\boldsymbol{x}), h_2(\boldsymbol{x})\right)\right)$$



$$h(\boldsymbol{x}) = \text{sign}(\text{sign}(\text{sign}(\boldsymbol{w}_1^T\boldsymbol{x}) - \text{sign}(\boldsymbol{w}_2^T\boldsymbol{x}) - 1.5) + \text{sign}(-\text{sign}(\boldsymbol{w}_1^T\boldsymbol{x}) + \text{sign}(\boldsymbol{w}_2^T\boldsymbol{x}) - 1.5) + 1.5)$$

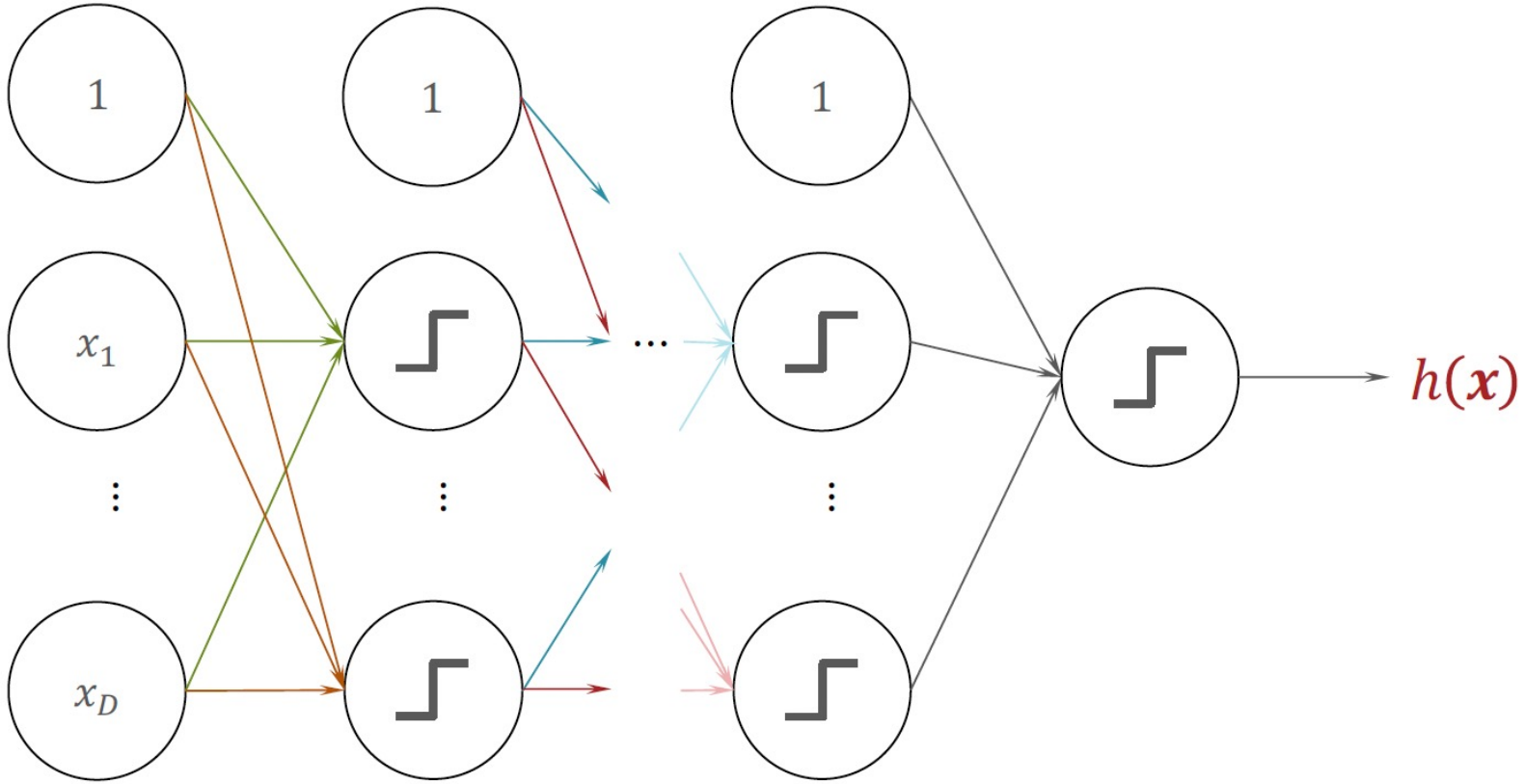
神经网络

- 人工神经网络就是AI的历史



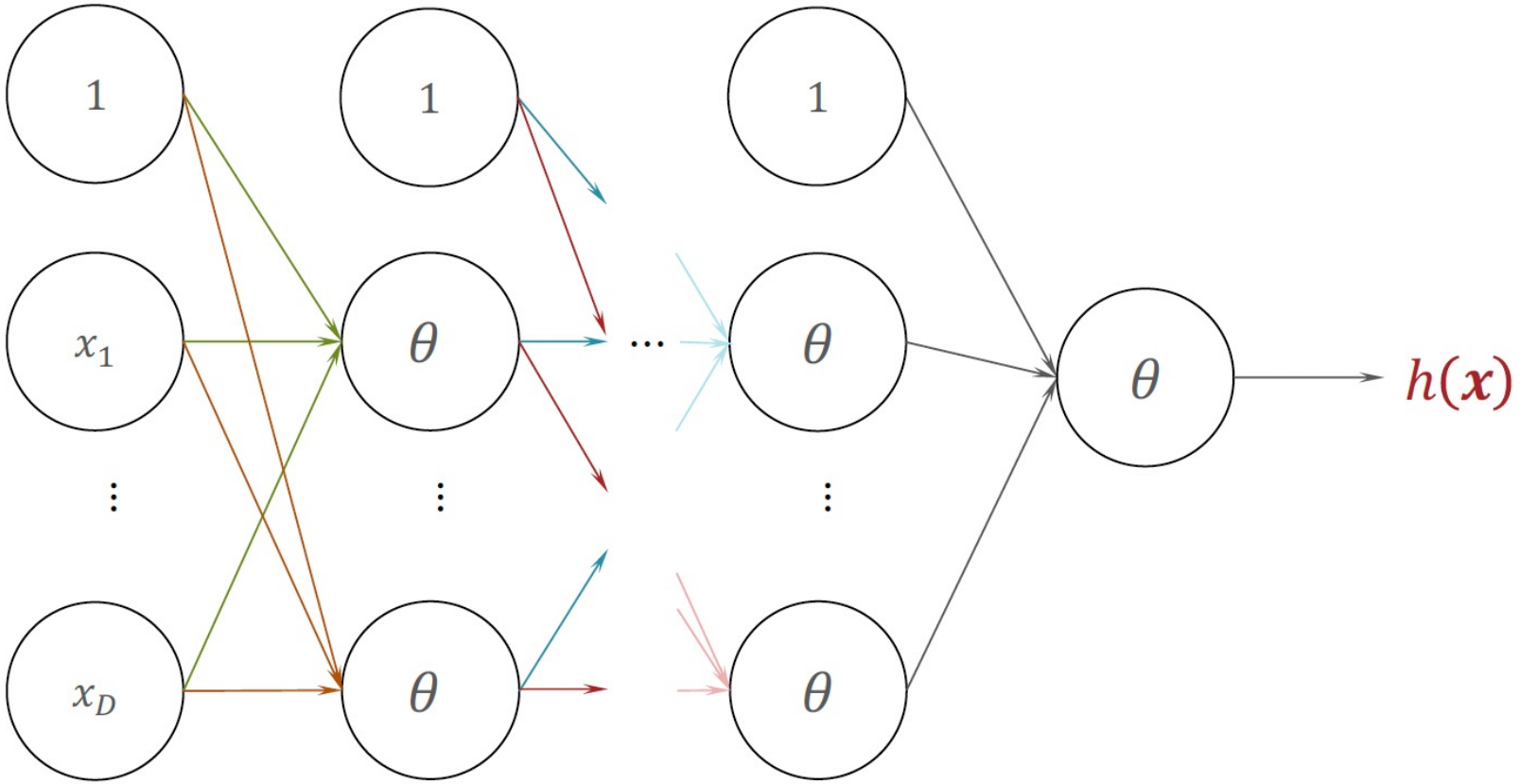
神经网络

- 多层感知器



神经网络

- 神经网络

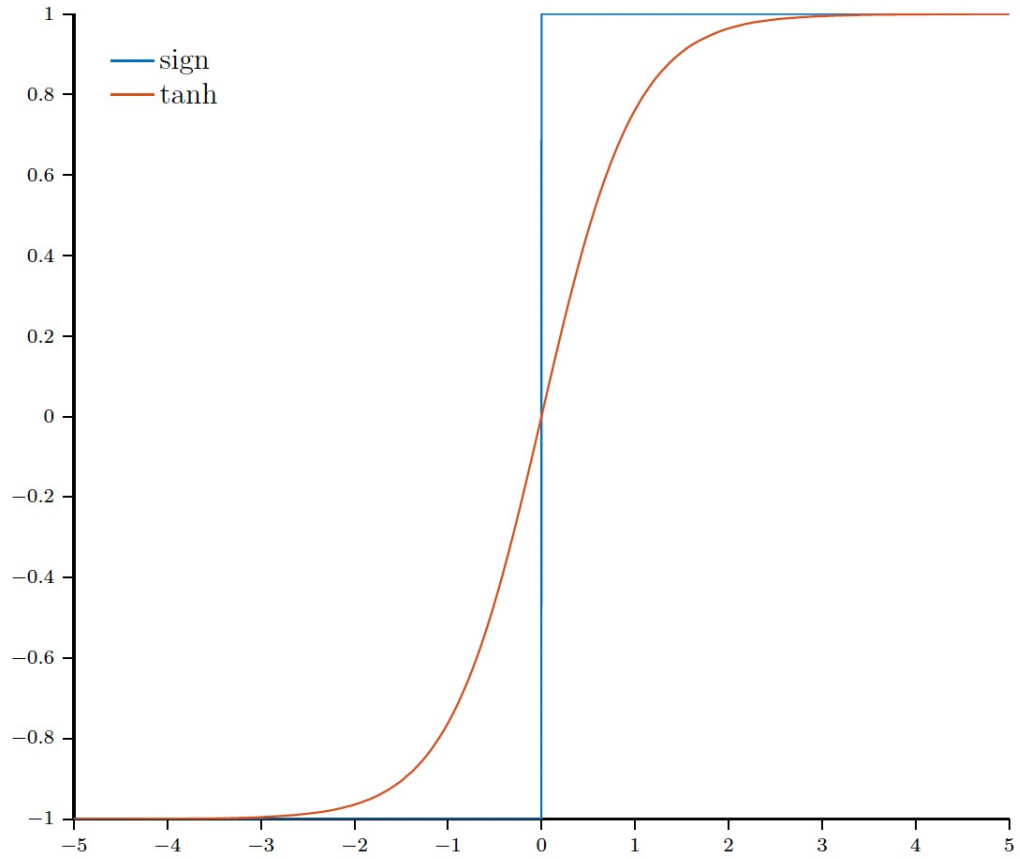


神经网络

- $\theta(z)$：激活函数
- 有明确定义的梯度

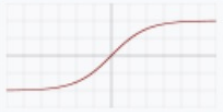
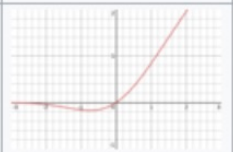
$$\tanh(z) = \frac{\sinh(z)}{\cosh(z)} = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

$$\bullet \ \frac{\partial \tanh(z)}{\partial z} = 1 - \tanh(z)^2$$



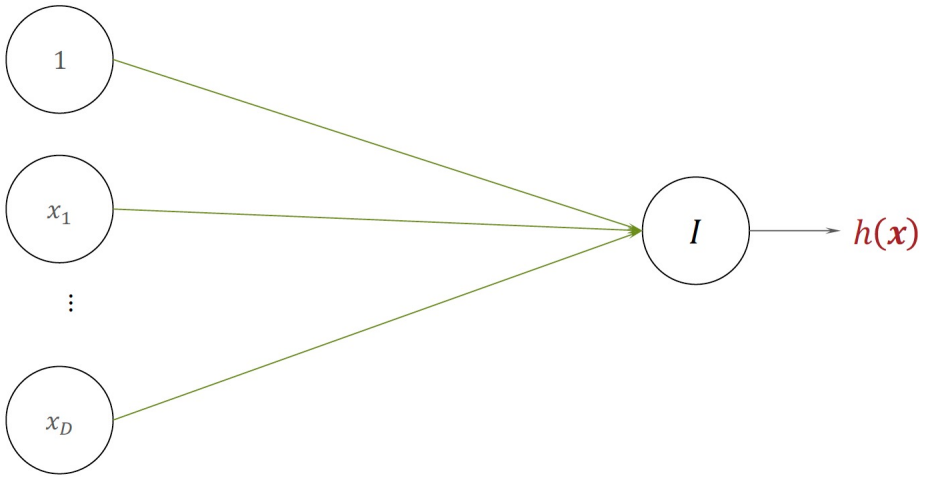
神经网络

- $\theta(z)$：其他激活函数

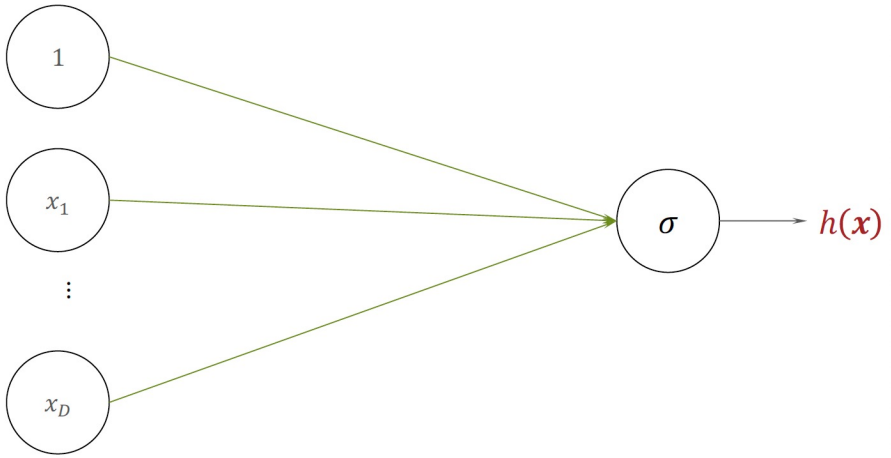
Logistic, sigmoid, or soft step		$\sigma(x) = \frac{1}{1+e^{-x}}$
Hyperbolic tangent (tanh)		$\tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$
Rectified linear unit (ReLU)[7]		$\begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$ $= \max\{0, x\} = x\mathbf{1}_{x>0}$
Gaussian Error Linear Unit (GELU)[4]		$\frac{1}{2}x\left(1+\operatorname{erf}\left(\frac{x}{\sqrt{2}}\right)\right)$ $= x\Phi(x)$
Softplus[8]		$\ln(1+e^{x})$
Exponential linear unit (ELU)[9]		$\begin{cases} \alpha\left(e^{x}-1\right) & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$ with parameter α
Leaky rectified linear unit (Leaky ReLU)[11]		$\begin{cases} 0.01x & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$
Parametric rectified linear unit (PReLU)[12]		$\begin{cases} \alpha x & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$ with parameter α

神经网络

- 线性回归的神经网络实现（回归）

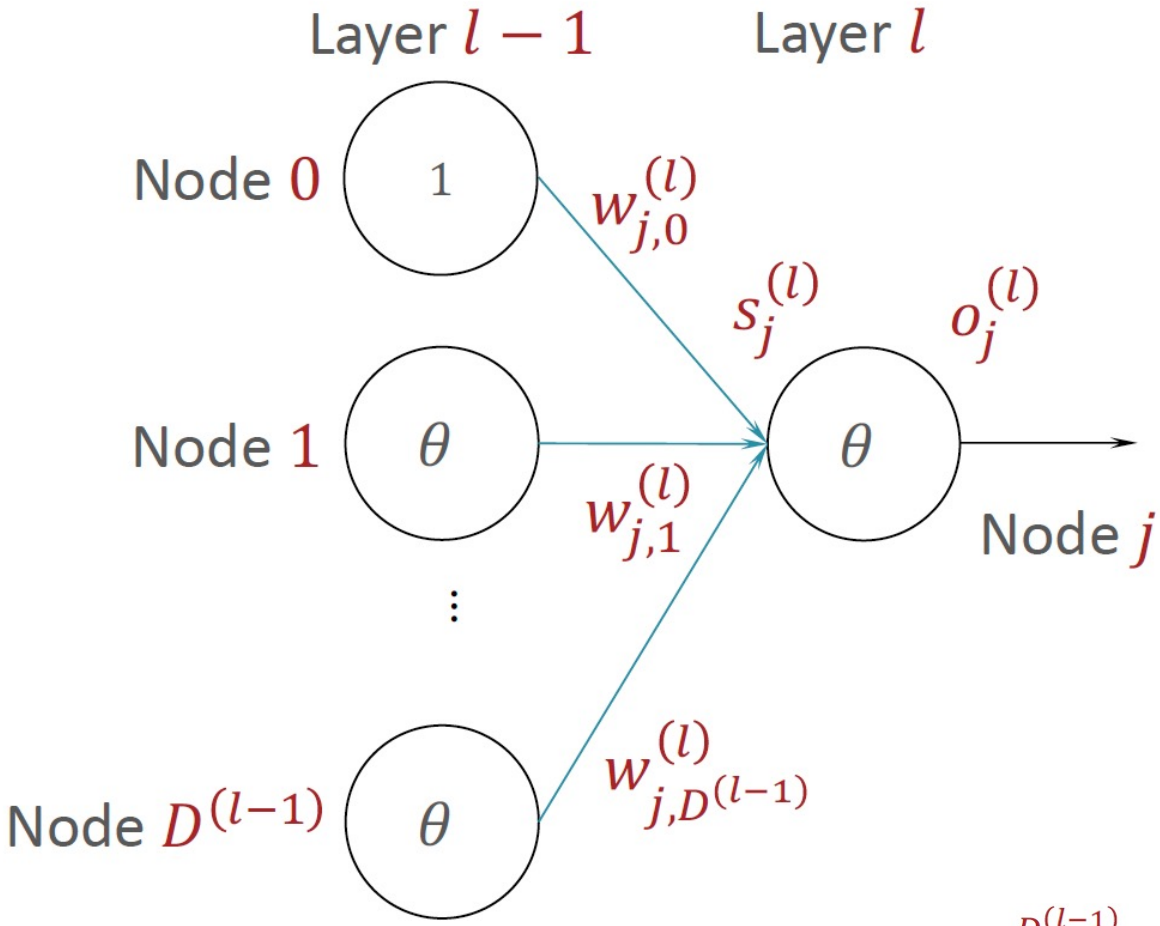


- 逻辑回归的神经网络实现（分类）



神经网络

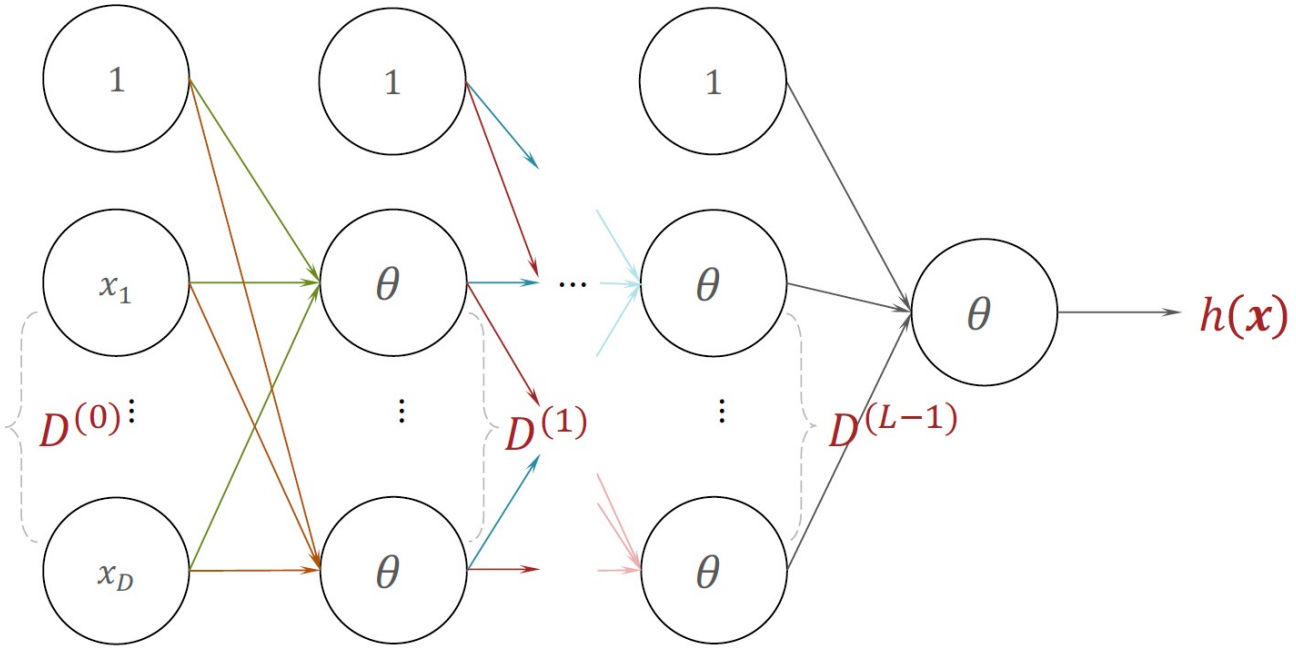
- 前向传播



$$s_j^{(l)} = \sum_{i=0}^{D^{(l-1)}} w_{j,i}^{(l)} o_i^{(l-1)} \text{ and } o_j^{(l)} = \theta\left(s_j^{(l)}\right)$$

神经网络

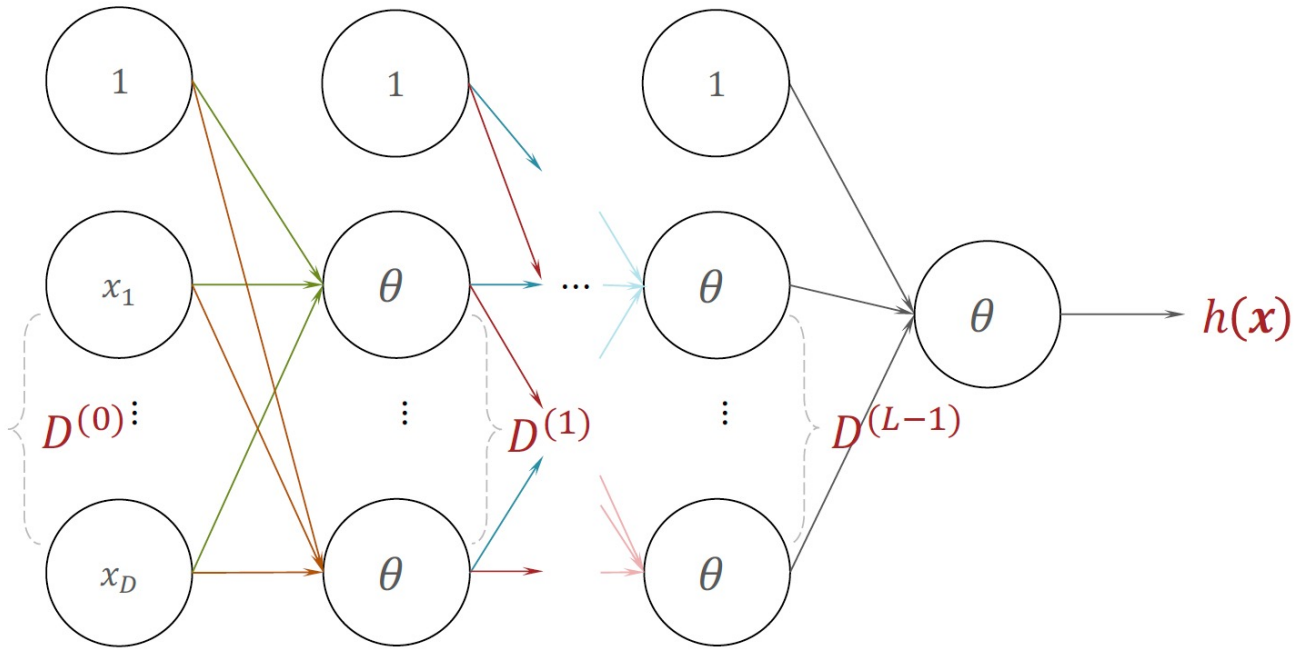
- 前向传播



- Input: weights $W^{(1)}, \dots, W^{(L)}$ and a query data point $\boldsymbol{x}$
- Initialize $\boldsymbol{o}^{(0)} = \begin{bmatrix} 1 \\ \boldsymbol{x} \end{bmatrix}$
- For $l = 1, \dots, L$
  - $\boldsymbol{s}^{(l)} = W^{(l)}\boldsymbol{o}^{(l-1)}$
  - $\boldsymbol{o}^{(l)} = \begin{bmatrix} 1 \\ \theta(\boldsymbol{s}^{(l)}) \end{bmatrix}$
- Output: $h_{W^{(1)},\dots,W^{(L)}}(\boldsymbol{x}) = \boldsymbol{o}^{(L)}$

神经网络

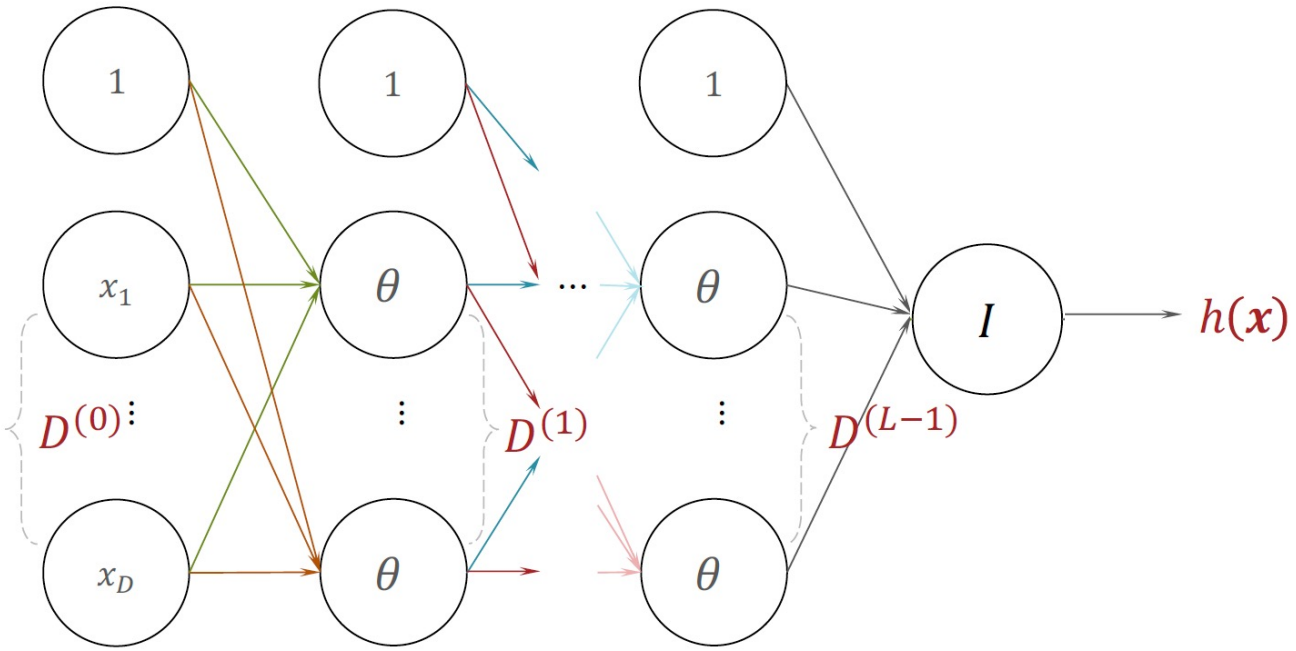
- 模型优化：梯度下降



- Input: $\mathcal{D} = \{(\boldsymbol{x}^{(n)}, y^{(n)})\}_{n=1}^{N}, \eta^{(0)}$
- Initialize all weights $W_{(0)}^{(1)}, \dots, W_{(0)}^{(L)}$ to small, random numbers and set $t = 0$
- While TERMINATION CRITERION is not satisfied
  - For $i \in \text{shuffle}(\{1, \dots, N\})$
    - For $l = 1, \dots, L$
      - Compute $G^{(l)} = \nabla_{W^{(l)}} \ell^{(i)} \left(W_{(t)}^{(1)}, \dots, W_{(t)}^{(L)}\right)$
      - Update $W^{(l)}$: $W_{(t+1)}^{(l)} = W_{(t)}^{(l)} - \eta^{(0)} G^{(l)}$
    - Increment t: $t = t + 1$
- Output: $W_{(t)}^{(1)}, \dots, W_{(t)}^{(L)}$

神经网络

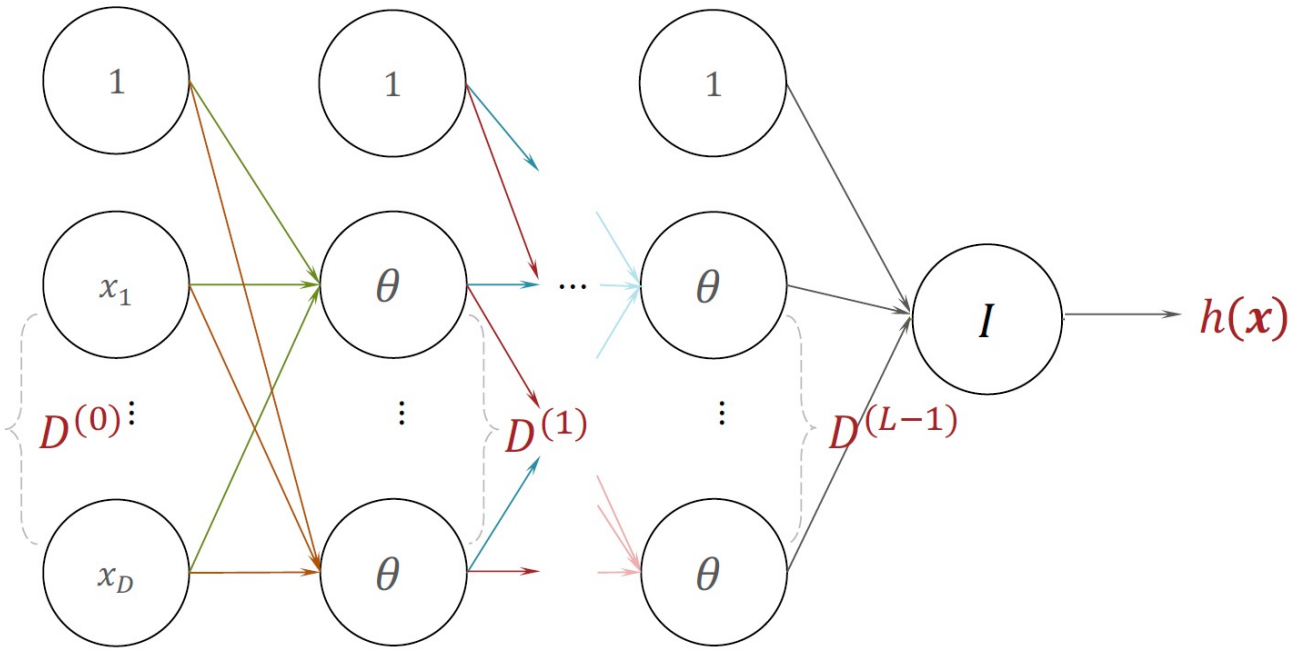
- 模型优化常规实现：反向传播



- Input: $W^{(1)}, \ldots, W^{(L)}$ and $\left(\boldsymbol{x}^{(i)}, y^{(i)}\right)$
- Run forward propagation with $\boldsymbol{x}^{(i)}$ to get $\boldsymbol{o}^{(1)}, \ldots, \boldsymbol{o}^{(L)}$
- (Optional) Compute $\ell^{(i)} = \left(o^{(L)} - y^{(i)}\right)^2$
- Initialize: $\boldsymbol{\delta}^{(L)} = 2\left(o_1^{(L)} - y^{(i)}\right)$

神经网络

- 模型优化常规实现：反向传播



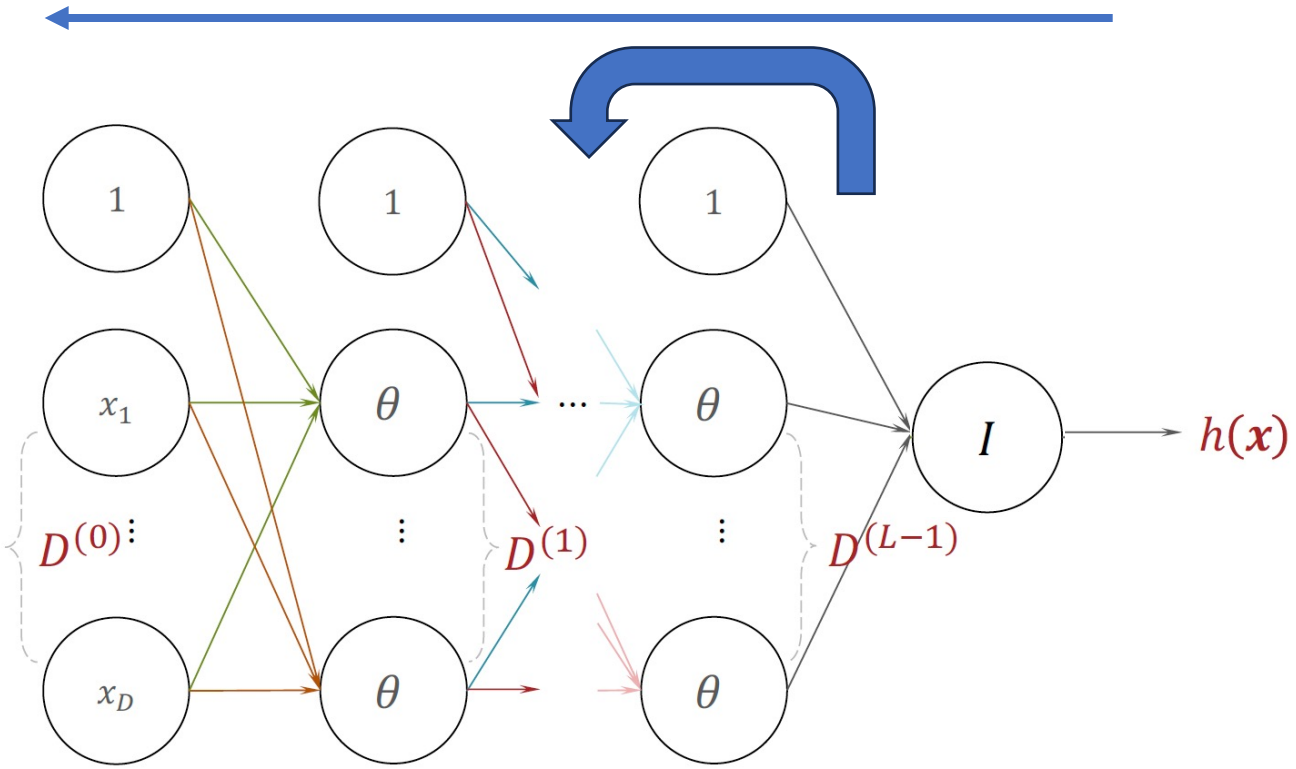
- Input: $W^{(1)}, \ldots, W^{(L)}$ and $\left(\boldsymbol{x}^{(i)}, y^{(i)}\right)$
- Run forward propagation with $\boldsymbol{x}^{(i)}$ to get $\boldsymbol{o}^{(1)}, \ldots, \boldsymbol{o}^{(L)}$
- (Optional) Compute $\ell^{(i)} = \left(o^{(L)} - y^{(i)}\right)^2$
- Initialize: $\boldsymbol{\delta}^{(L)} = 2\left(o_1^{(L)} - y^{(i)}\right)$

$$\delta^{(l+1)} = \frac{\partial \ell}{\partial z^{(l+1)}}$$

$$z^{(l+1)} = W^{(l+1)} o^{(l)}$$

神经网络

- 模型优化常规实现：反向传播



$$\delta^{(l)} = \frac{\partial \ell}{\partial z^{(l)}} = \left(\frac{\partial \ell}{\partial o^{(l)}}\right) \odot \left(\frac{\partial o^{(l)}}{\partial z^{(l)}}\right)$$

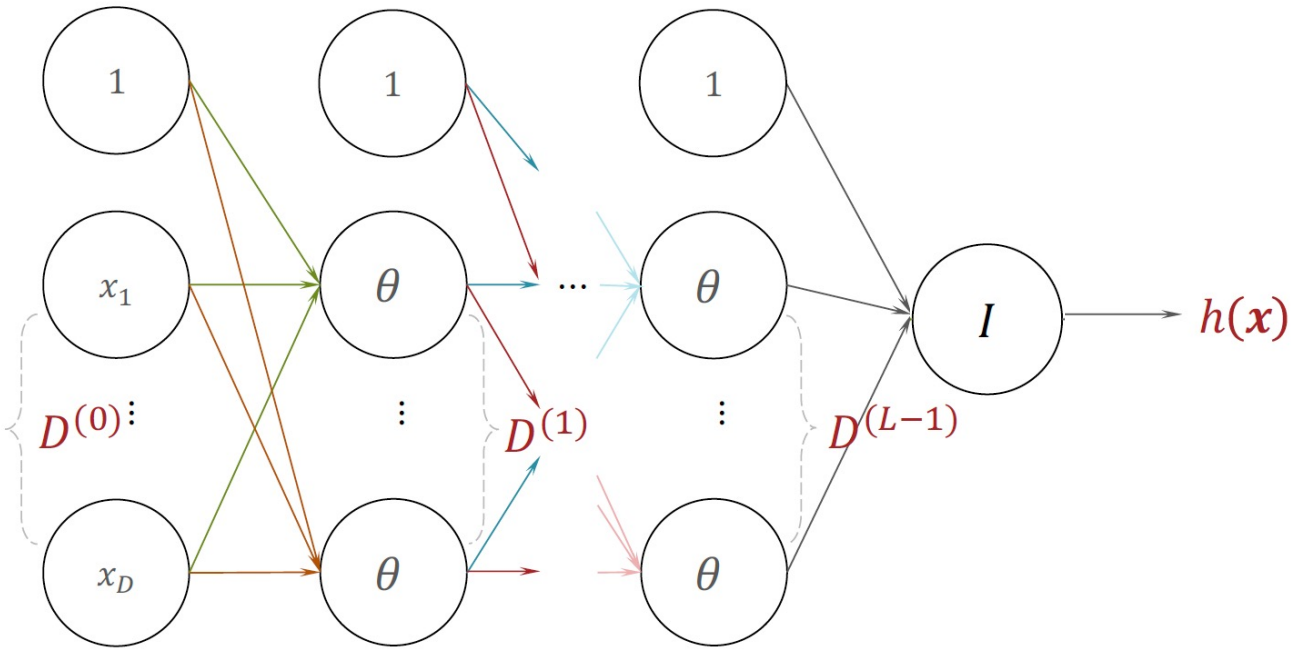
- Compute $\boldsymbol{\delta}^{(l)} = W^{(l+1)^T}\boldsymbol{\delta}^{(l+1)} \odot \left(1 - \boldsymbol{o}^{(l)} \odot \boldsymbol{o}^{(l)}\right)$

$$\frac{\partial \ell}{\partial o^{(l)}} = W^{(l+1)T}\delta^{(l+1)}$$

- $\frac{\partial \tanh(z)}{\partial z} = 1 - \tanh(z)^2$

神经网络

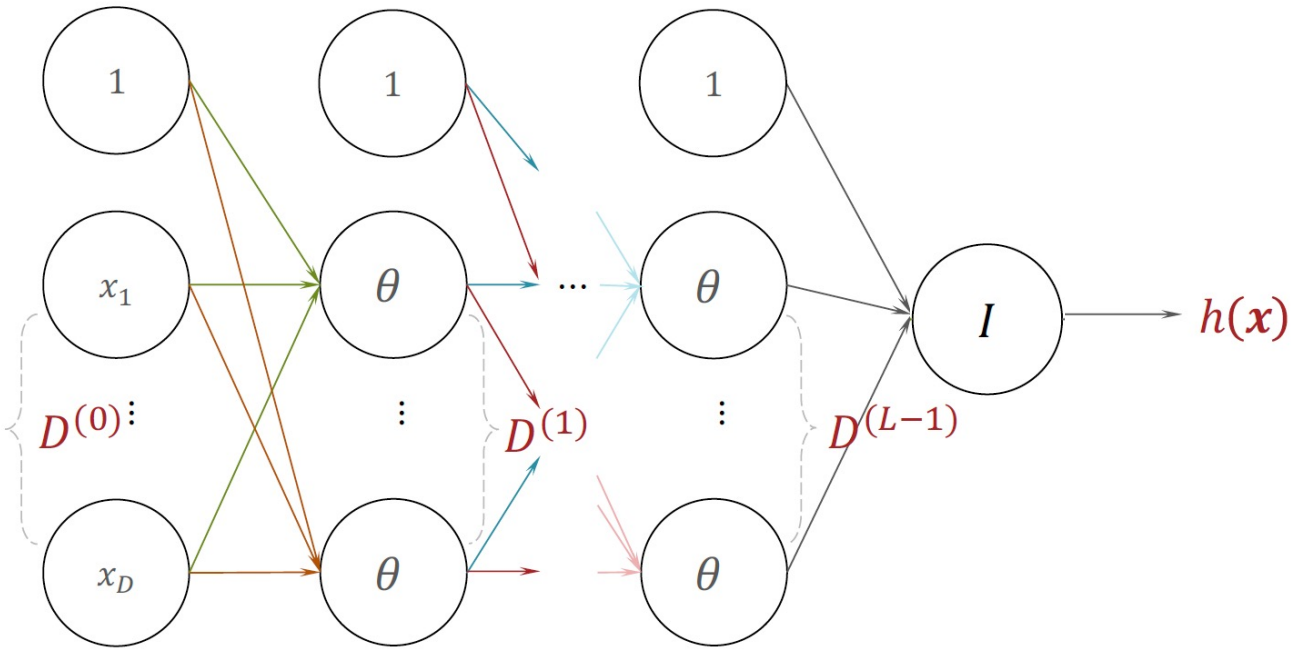
- 模型优化常规实现：反向传播



- Input: $W^{(1)}, \ldots, W^{(L)}$ and $\left(\boldsymbol{x}^{(i)}, y^{(i)}\right)$
- Run forward propagation with $\boldsymbol{x}^{(i)}$ to get $\boldsymbol{o}^{(1)}, \ldots, \boldsymbol{o}^{(L)}$
- (Optional) Compute $\ell^{(i)} = \left(o^{(L)} - y^{(i)}\right)^2$
- Initialize: $\boldsymbol{\delta}^{(L)} = 2\left(o_1^{(L)} - y^{(i)}\right)$
- For $l = L-1, \ldots, 1$
  - Compute $\boldsymbol{\delta}^{(l)} = {W^{(l+1)}}^T \boldsymbol{\delta}^{(l+1)} \odot \left(1 - \boldsymbol{o}^{(l)} \odot \boldsymbol{o}^{(l)}\right)$
  - Compute $G^{(l)} = \boldsymbol{\delta}^{(l)} {\boldsymbol{o}^{(l-1)}}^T$

神经网络

- 模型优化常规实现：反向传播



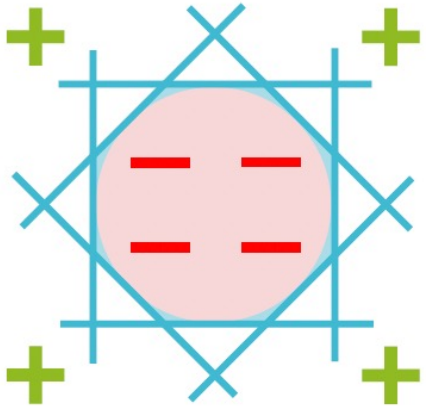
- Input: $W^{(1)}, \ldots, W^{(L)}$ and $\left(\boldsymbol{x}^{(i)}, y^{(i)}\right)$
- Run forward propagation with $\boldsymbol{x}^{(i)}$ to get $\boldsymbol{o}^{(1)}, \ldots, \boldsymbol{o}^{(L)}$
- (Optional) Compute $\ell^{(i)} = \left(o^{(L)} - y^{(i)}\right)^2$
- Initialize: $\boldsymbol{\delta}^{(L)} = 2\left(o_1^{(L)} - y^{(i)}\right)$
- For $l = L-1, \ldots, 1$
  - Compute $\boldsymbol{\delta}^{(l)} = {W^{(l+1)}}^T \boldsymbol{\delta}^{(l+1)} \odot \left(1 - \boldsymbol{o}^{(l)} \odot \boldsymbol{o}^{(l)}\right)$
  - Compute $G^{(l)} = \boldsymbol{\delta}^{(l)} {\boldsymbol{o}^{(l-1)}}^T$
- Output: $G^{(1)}, \ldots, G^{(L)}$, the gradients of $\ell^{(i)}$ w.r.t $W^{(1)}, \ldots, W^{(L)}$

torch.Tensor.backward

```
Tensor.backward(gradient=None, retain_graph=None, create_graph=False, inputs=None)
```

神经网络

- 理论上，无限宽的3层MLP可以拟合任何函数
  - Theorem: any function that can be decomposed into perceptrons can be modelled exactly using a 3-layer MLP
  - Any smooth decision boundary can be approximated to an arbitrary precision using a finite number of perceptrons



神经网络

- 深度学习的定义：more than one layer

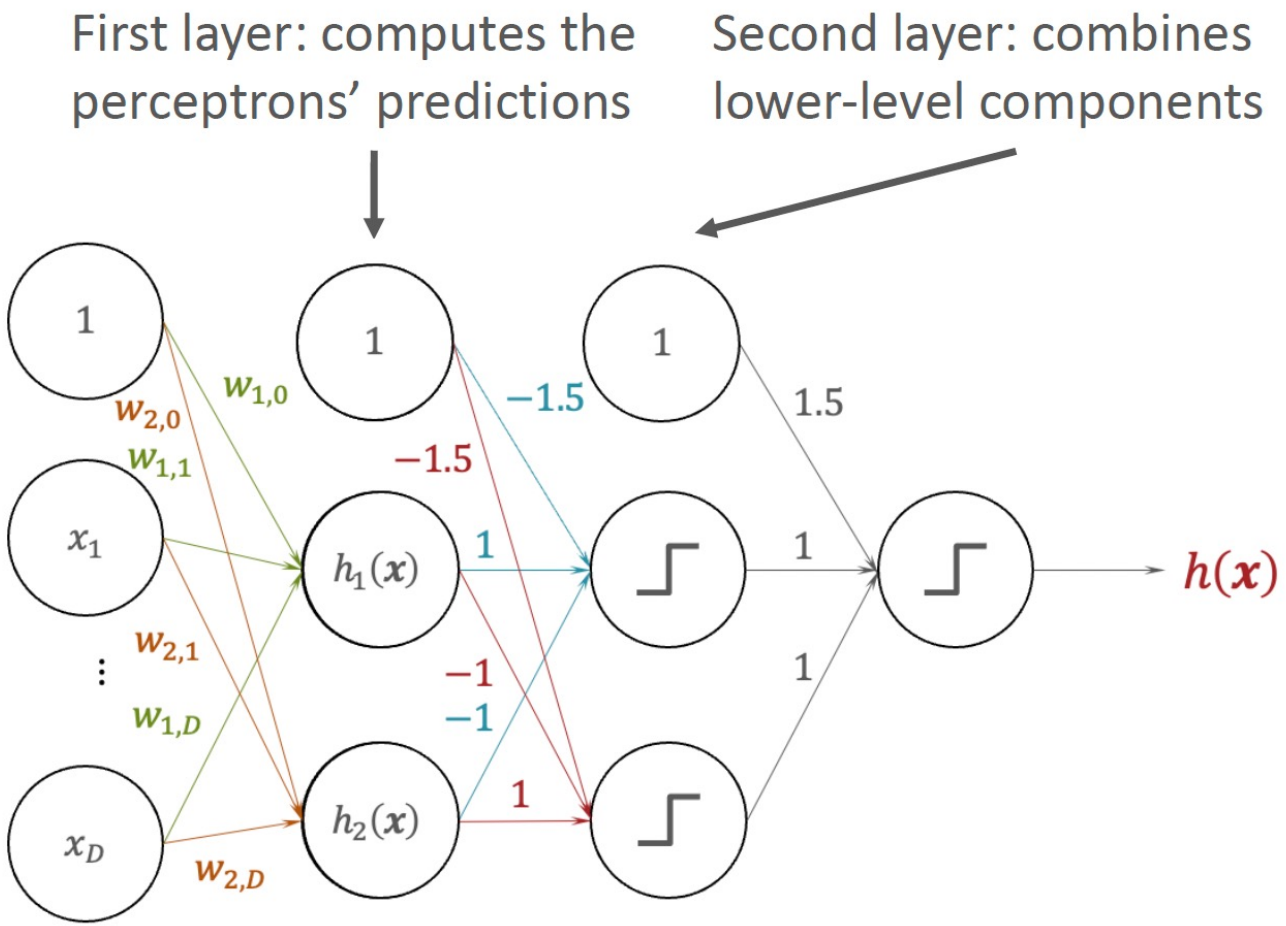
Definition [edit]

Deep learning is a class of machine learning algorithms that[11](pp199–200) uses multiple layers to progressively extract higher level features from the raw input. For example, in image processing, lower layers may identify edges, while higher layers may identify the concepts relevant to a human such as digits or letters or faces.

- 本质原因：无限宽不现实，深度可以替代宽度（因为ResNet, BN etc...）

神经网络

- 深度学习：


First layer: computes the perceptrons' predictions
Second layer: combines lower-level components
1
1
1
x_1
x_D
$w_{1,0}$
$w_{2,0}$
$w_{1,1}$
$w_{2,1}$
$w_{1,D}$
$w_{2,D}$
$h_1(x)$
$h_2(x)$
−1.5
−1.5
1
−1
−1
1
1.5
1
1
$h(x)$

神经网络

- 卷积神经网络（CNN）：加入卷积操作的神经网络
- 老婆饼没有老婆，卷积神经网络没有卷积
- 本质是卷积核的互相关

0	0	0	0	0	0
0	1	2	2	1	0
0	2	4	4	2	0
0	1	3	3	1	0
0	1	2	3	1	0
0	0	1	1	0	0

$*$

0	1	0
1	-4	1
0	1	0

$=$

0			

$$(0*0)+(0*1)+(0*0)+(0*1)+(1*-4)+(2*1)+(0*0)+(2*1)+(4*0)=0$$

神经网络

- 卷积神经网络（CNN）：加入卷积操作的神经网络
- 老婆饼没有老婆，卷积神经网络没有卷积
- 本质是卷积核的互相关

0	0	0	0	0	0
0	1	2	2	1	0
0	2	4	4	2	0
0	1	3	3	1	0
0	1	2	3	1	0
0	0	1	1	0	0

$*$

0	1	0
1	-4	1
0	1	0

$=$

0	-1		

$$(0*0)+(0*1)+(0*0)+(1*1)+(2*-4)$$
$$+(2*1)+(2*0)+(4*1)+(4*0)=-1$$

神经网络

- 卷积神经网络（CNN）：加入卷积操作的神经网络
- 老婆饼没有老婆，卷积神经网络没有卷积
- 本质是卷积核的互相关

0	0	0	0	0	0
0	1	2	2	1	0
0	2	4	4	2	0
0	1	3	3	1	0
0	1	2	3	1	0
0	0	1	1	0	0

$*$

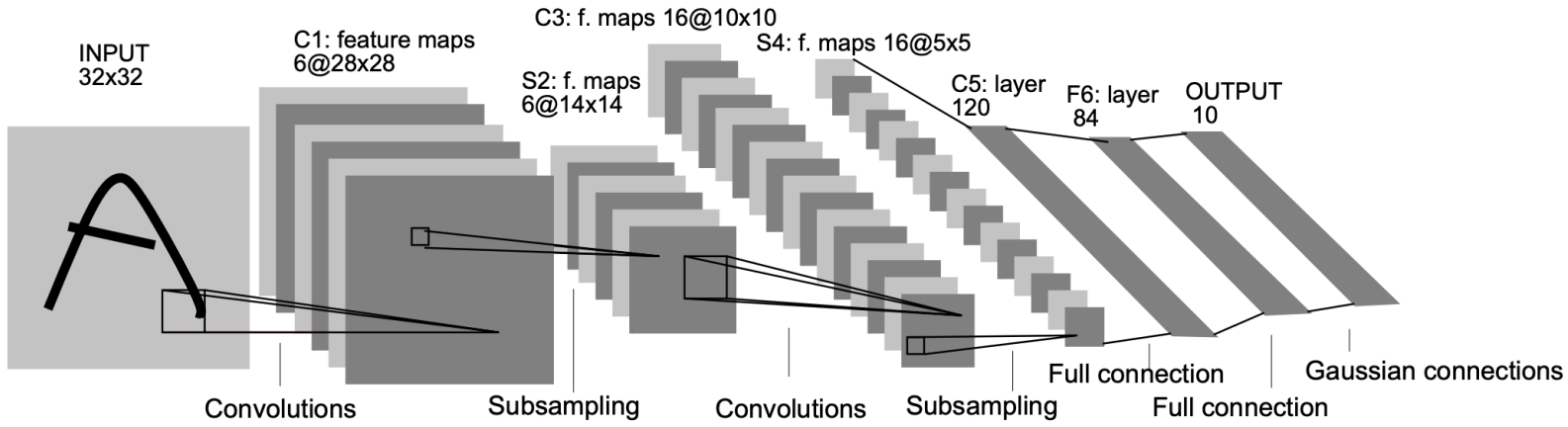
0	1	0
1	-4	1
0	1	0

$=$

0	-1	-1	0
-2	-5	-5	-2
2	-2	-1	3
-1	0	-5	0

神经网络

- 现有的网络结构30年前就已经完成了

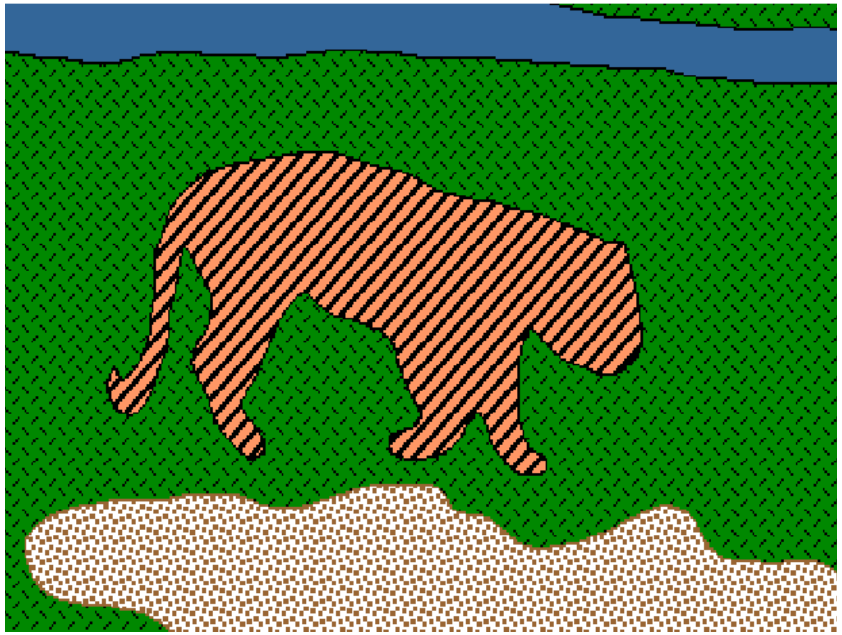


Y. Lecun, *et al*. “Gradient-based learning applied to document recognition“, P IEEE, 1998.

目录

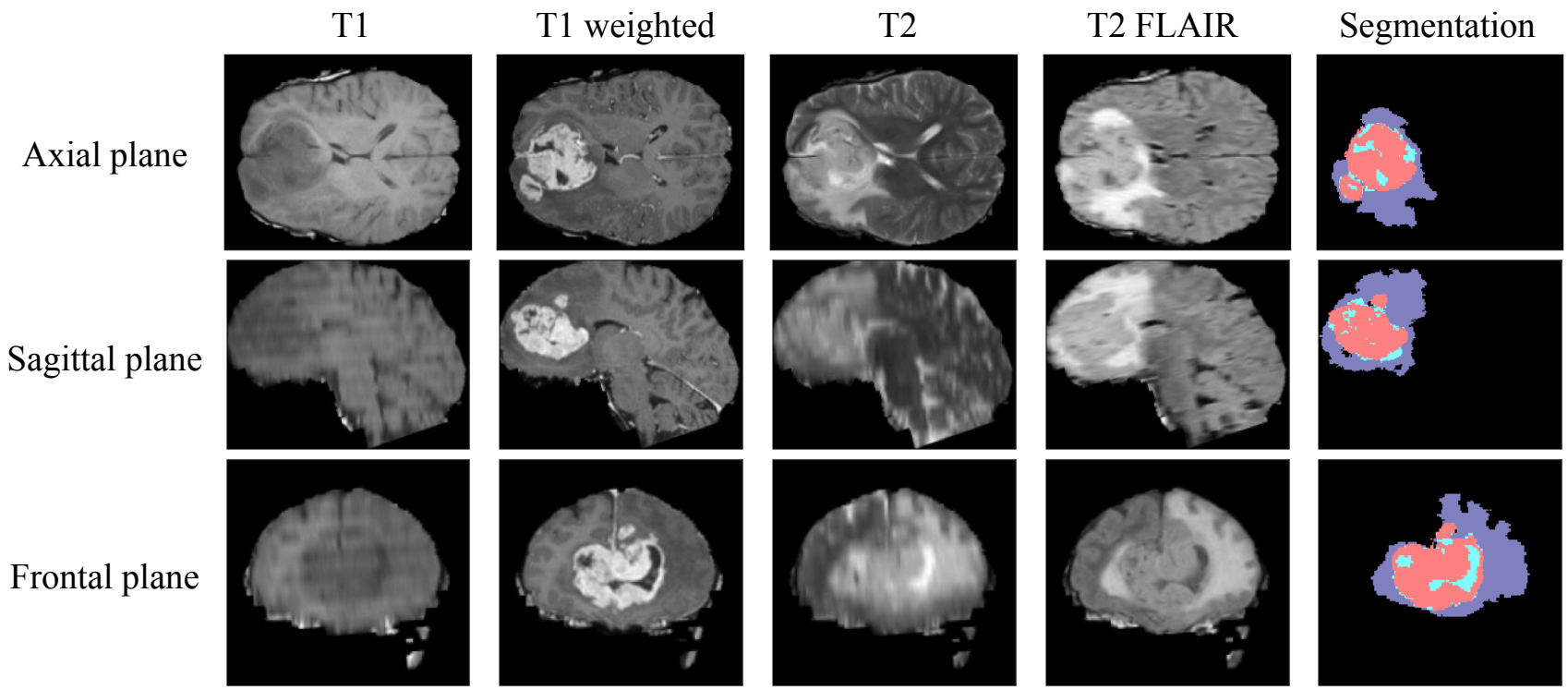
图像分割

- 将图像划分成多个结构意义的区域



医学图像分割

- 将医学图像（磁共振，CT，超声等）划分成多个结构意义的区域

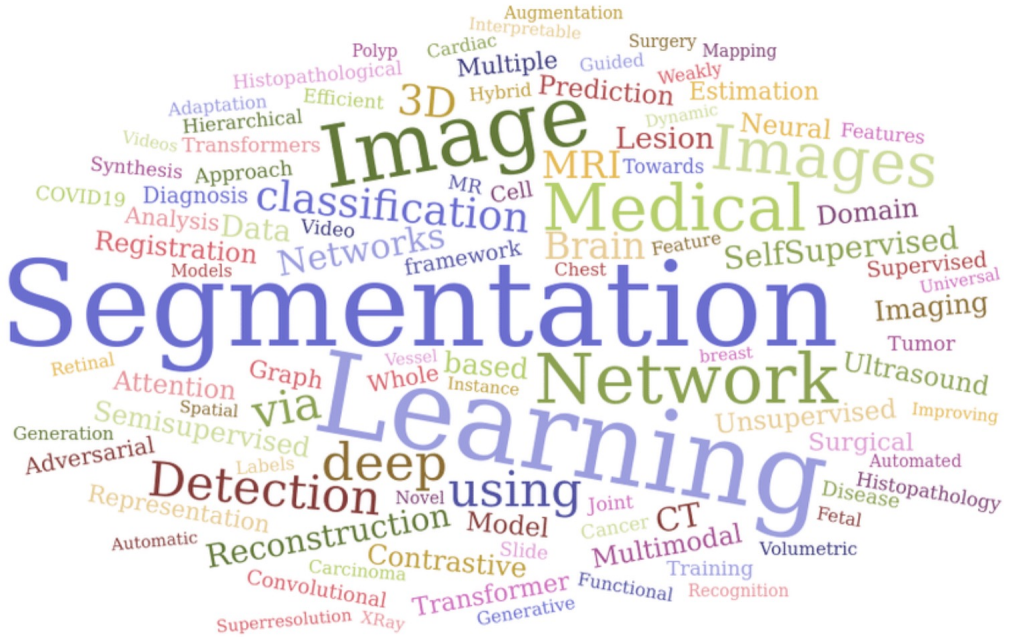


医学图像分割

- 医学图像分割是最普遍的医学图像处理任务
- **40%**的MICCAI文章是关于医学图像分割

Task type

- Classification 60
- Detection 38
- Modeling 5
- Reconstruction 7
- Registration 6
- Regression 11
- Segmentation 96



https://grand-challenge.org/challenges/

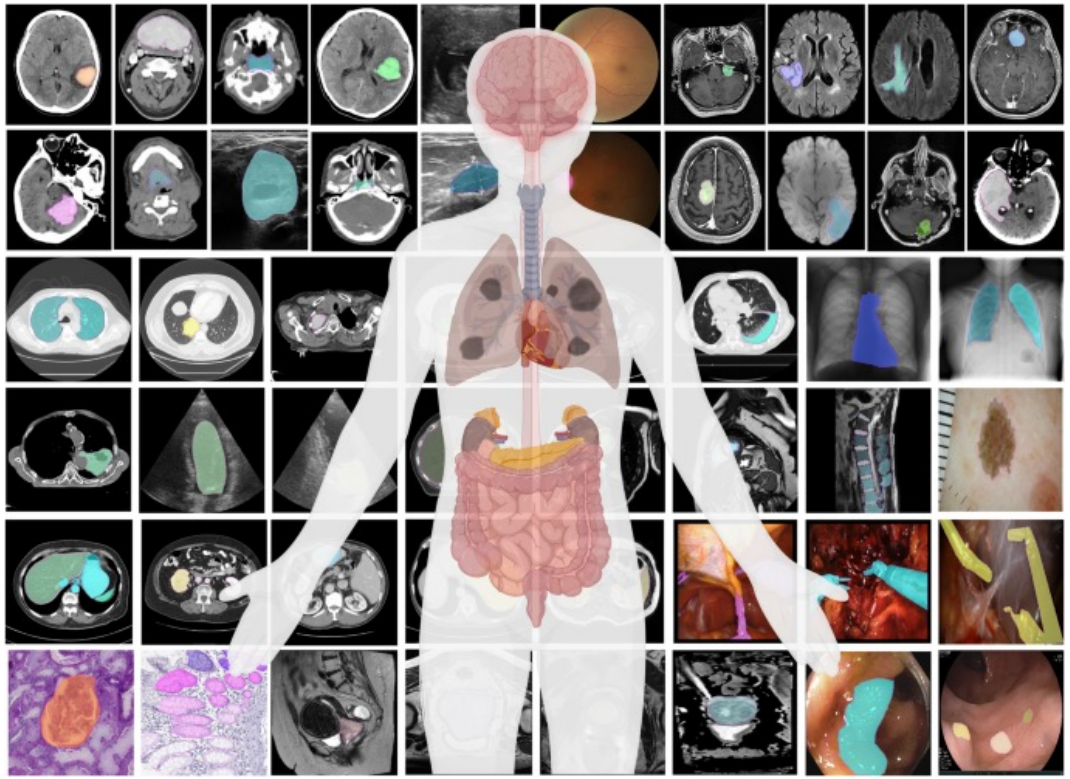
Frequency maps for MICCAI'21

医学图像分割

- 多用于简化繁琐的标注过程
- 后续任务的先决条件

- Abdomen 15
  - Colon 1
  - Kidney 6
  - Liver 11
  - Pancreas 3
  - Spleen 3
- Cardiac 13
  - Heart 13
- Head and Neck 54
  - Brain 41
  - Cranium 0
  - Retina 12
  - Teeth 1
- Lower Limb 4
  - Knee 4
- Pelvis 10
  - Cervix 2
  - Prostate 8
- Skin 2
  - Skin 2
- Spine 3
  - Spinal Cord 0
  - Vertebral Column 3
- Thorax 25
  - Breast 9
  - Lung 18
- Upper Limb 0
  - Hand 0

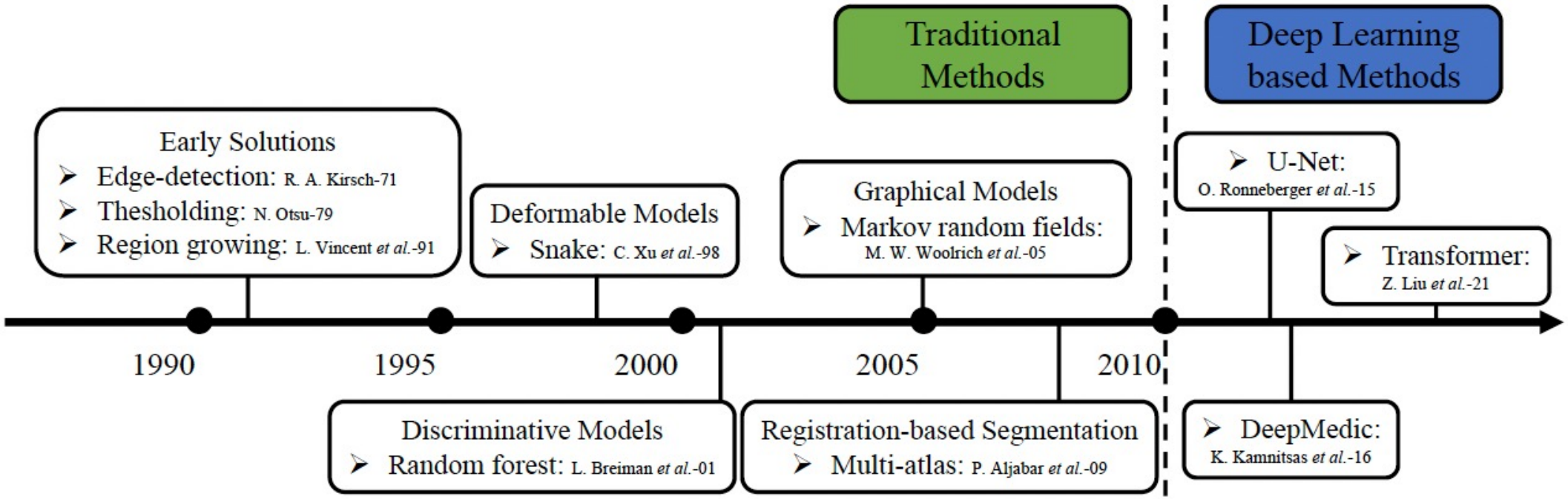
https://grand-challenge.org/challenges/



Jun Ma, *et al.* " Segment anything in medical images", Nat. Com., 2024.

医学图像分割

- 研究已有**50年**历史
- 2015年（U-Net）以后迅速收敛到神经网络

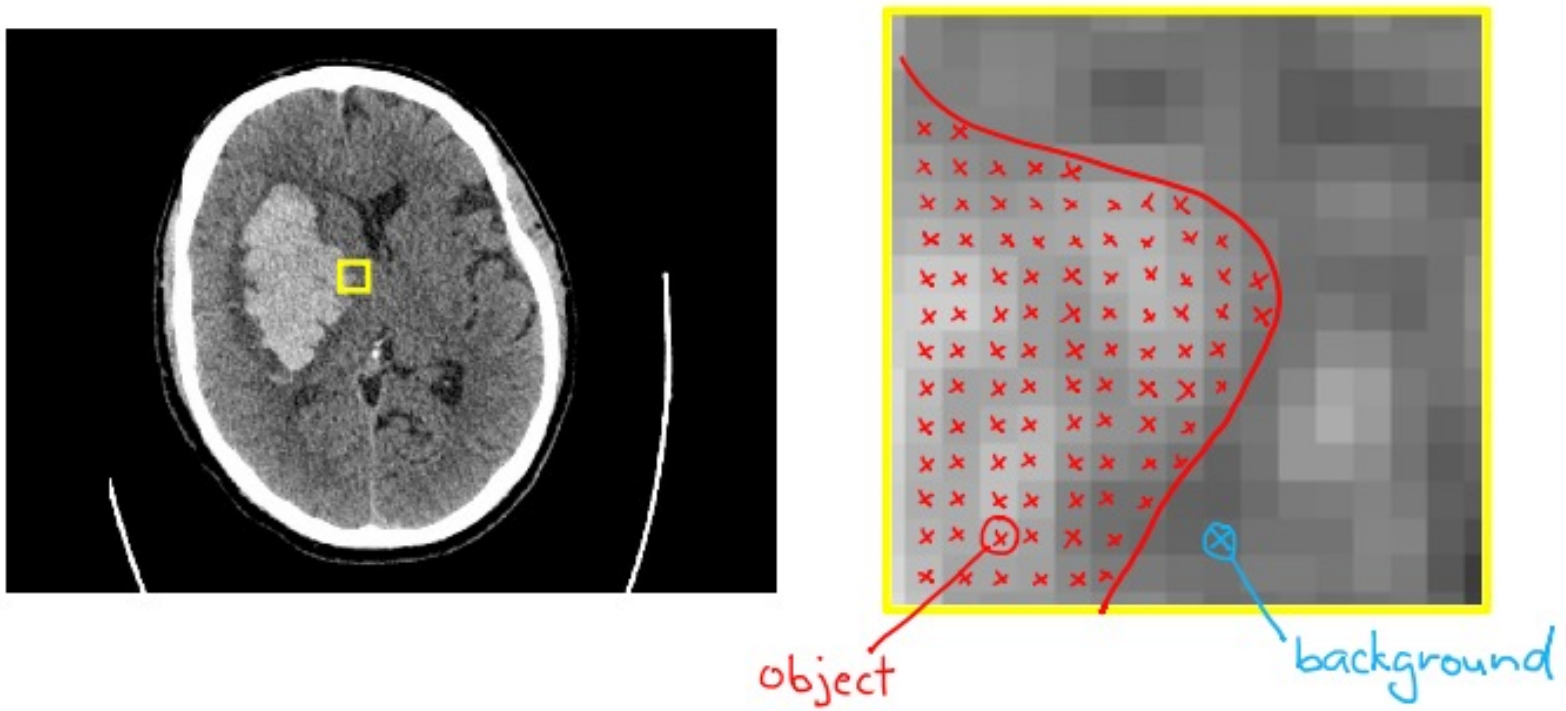


O. Ronneberger, P. Fischer and T. Brox. “U-net: Convolutional networks for biomedical image segmentation“, MICCAI, 2015

Most cited paper in the MICCAI history

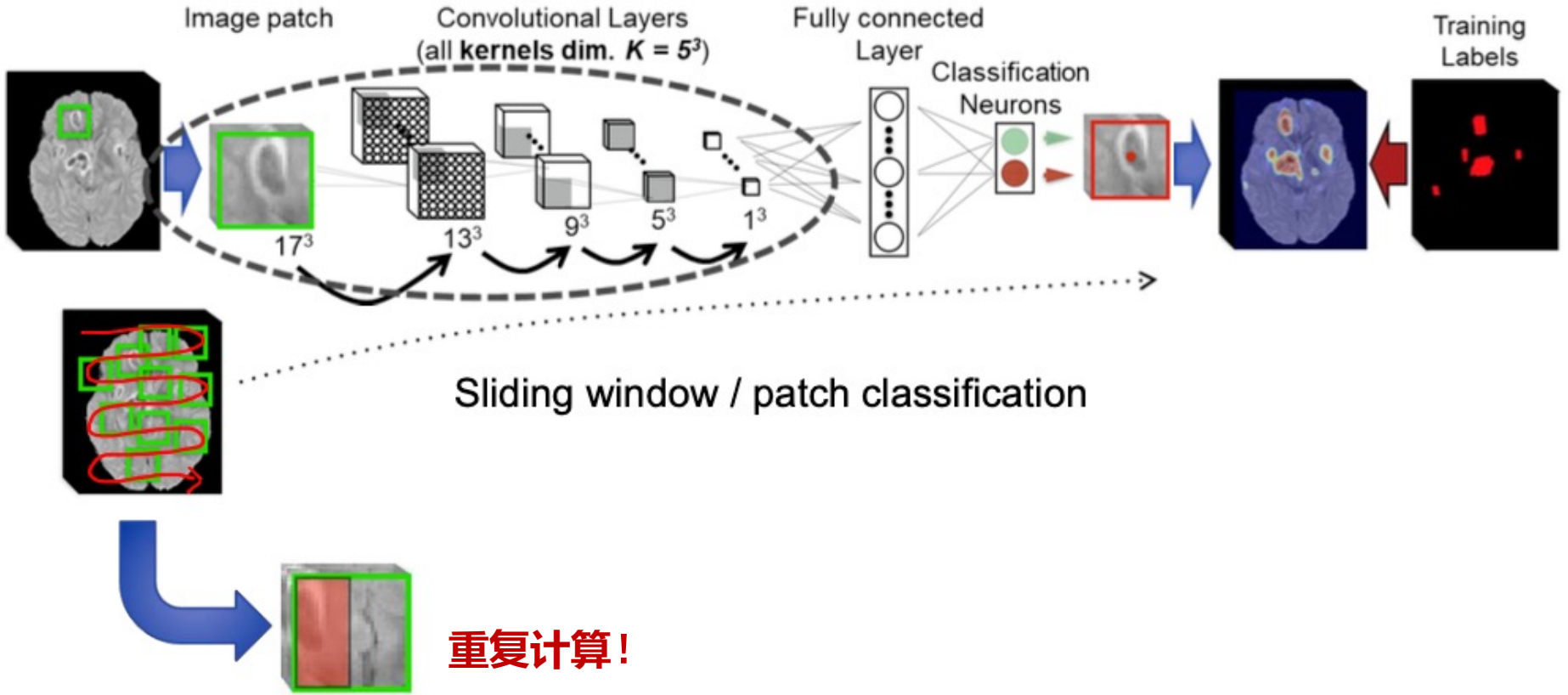
医学图像分割

- 基于稠密分类的图像分割



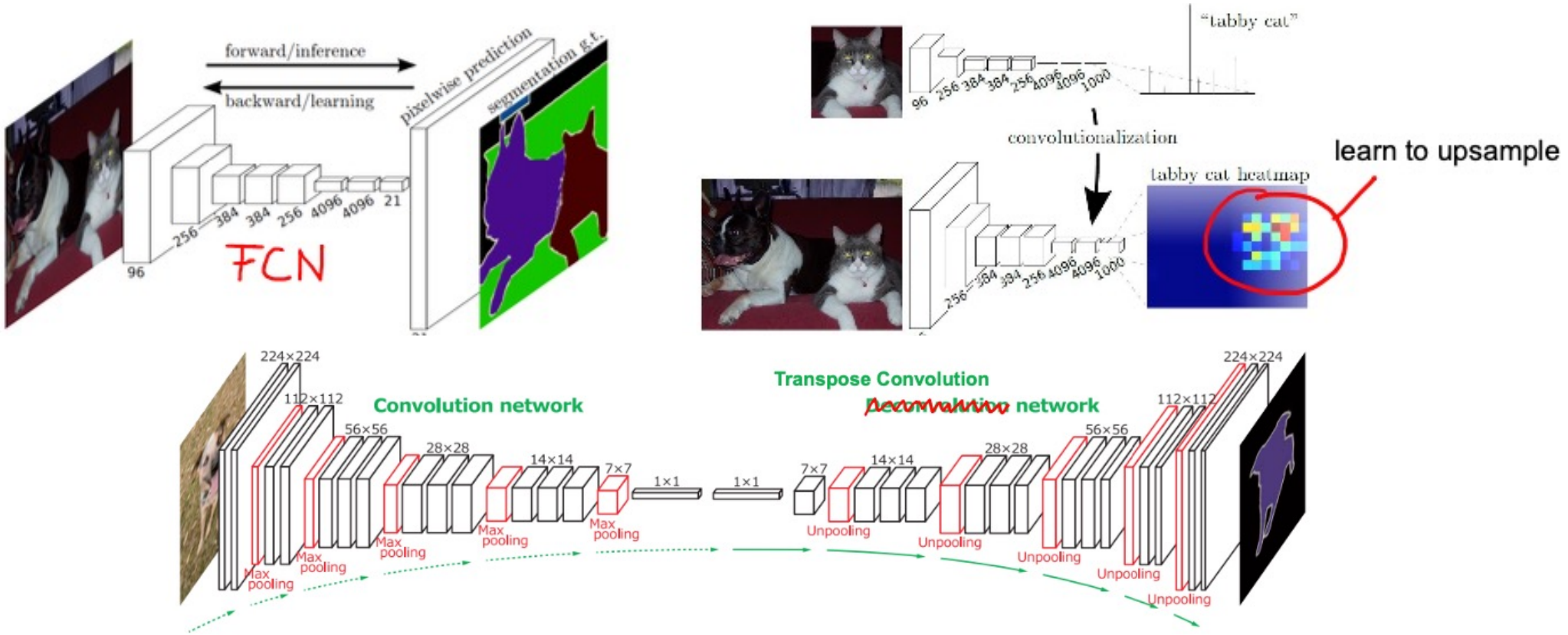
医学图像分割

- 基于稠密分类的图像分割



医学图像分割

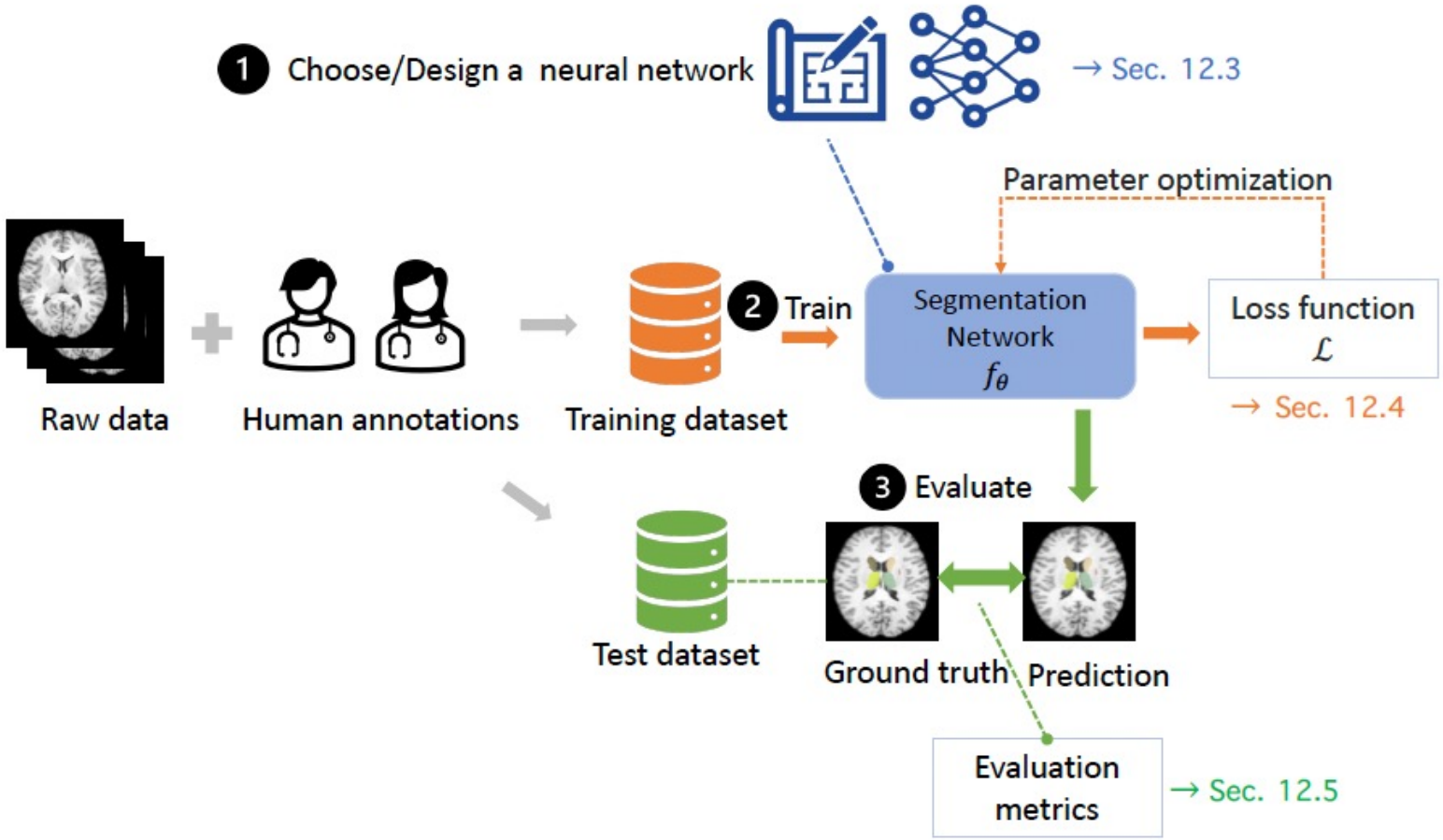
- 全卷积神经网络



J. Long, *et al*. “Fully convolutional networks for semantic segmentation“, CVPR, 2015.

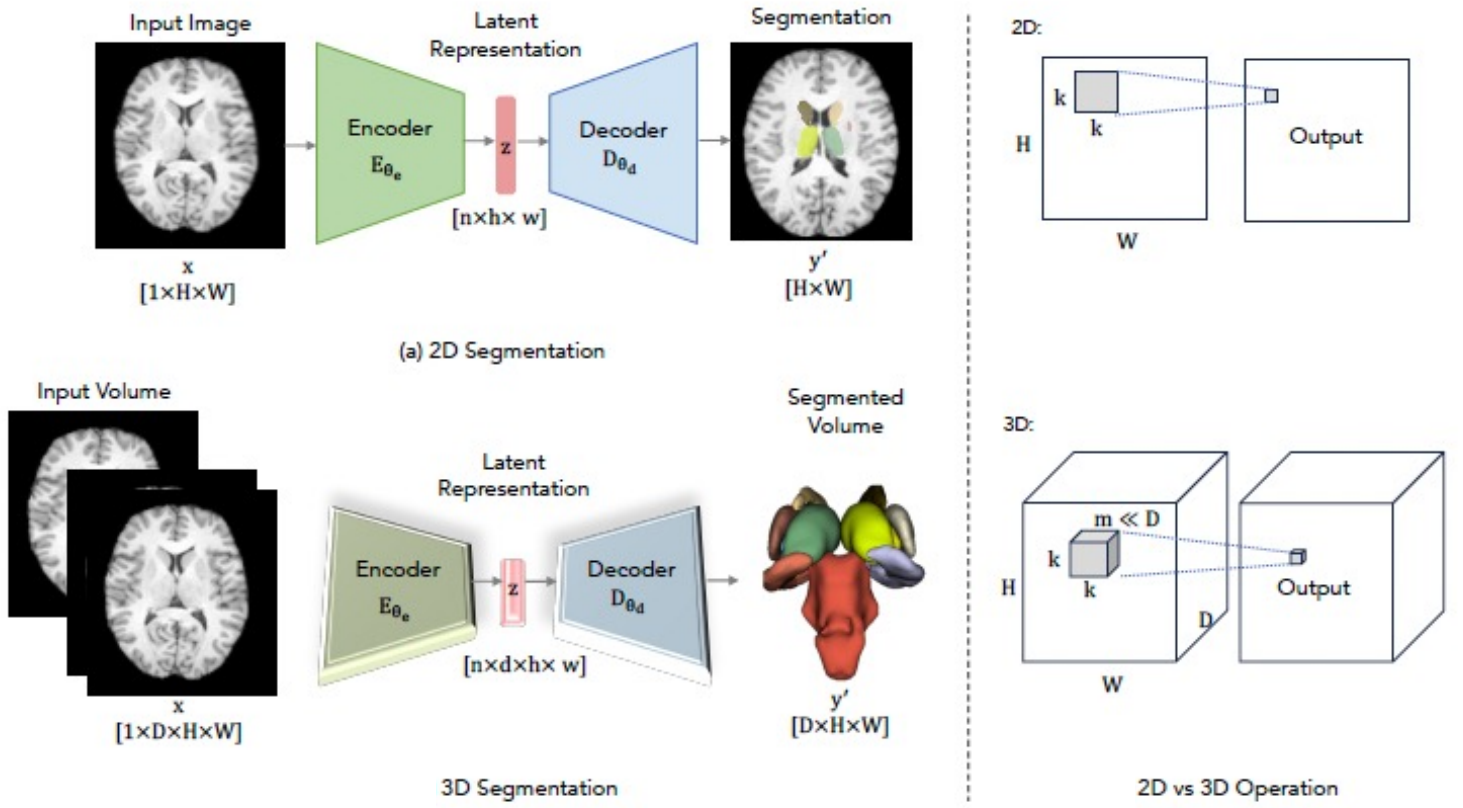
从训练到评估

- 分割算法的开发过程



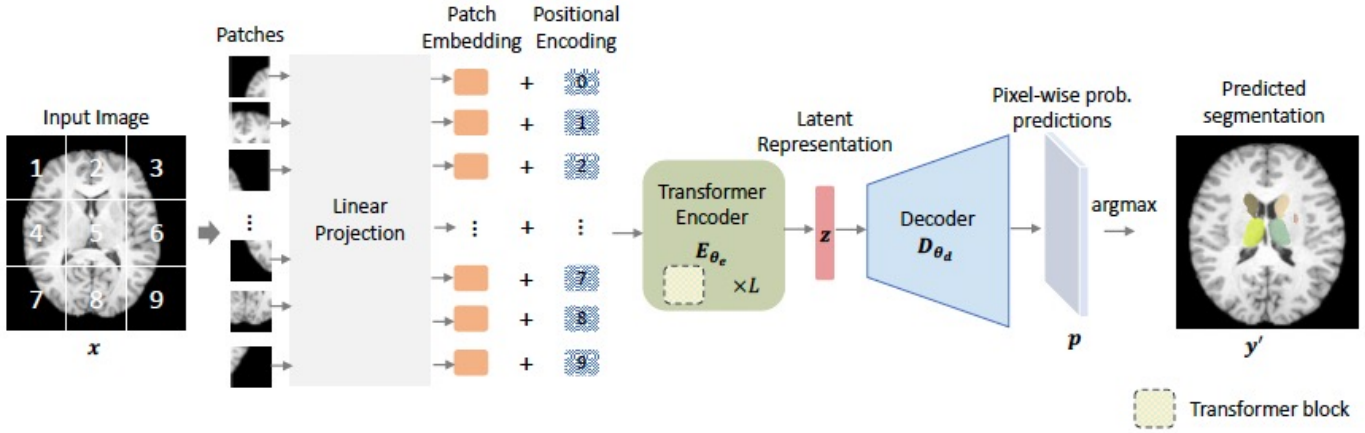
模型结构设计

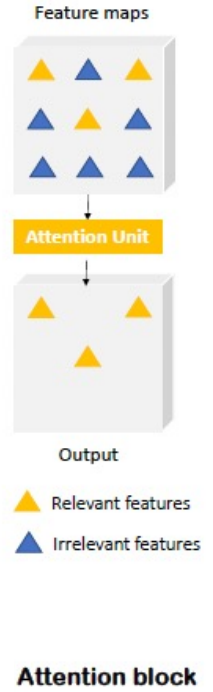
- 与自然图像不同，医学图像大多为3D



模型结构设计

- Transformer和注意力机制强调空间相关性

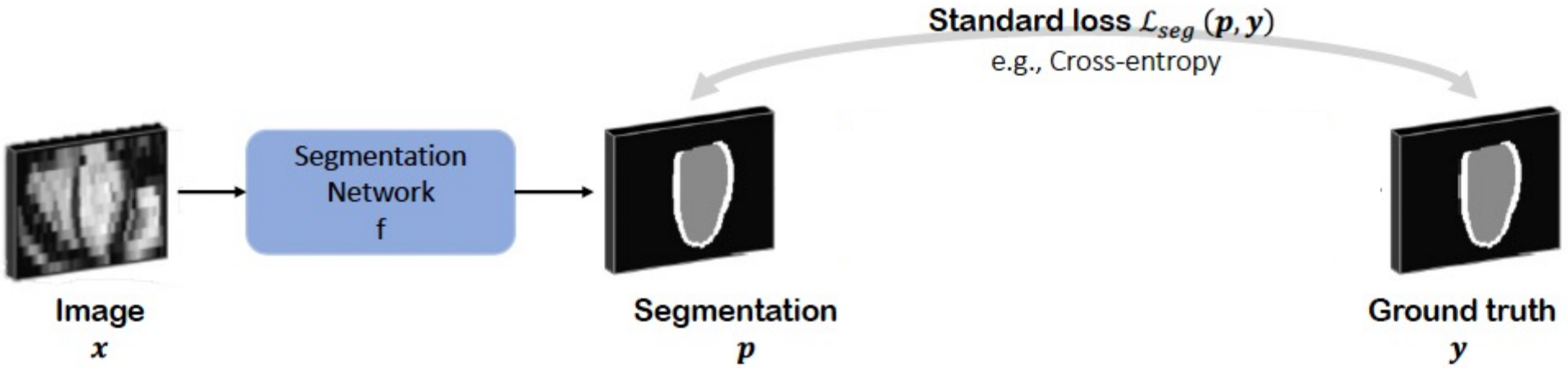




Attention block

模型训练

- 全监督学习常规为Cross-entropy和Soft-Dice损失的平均

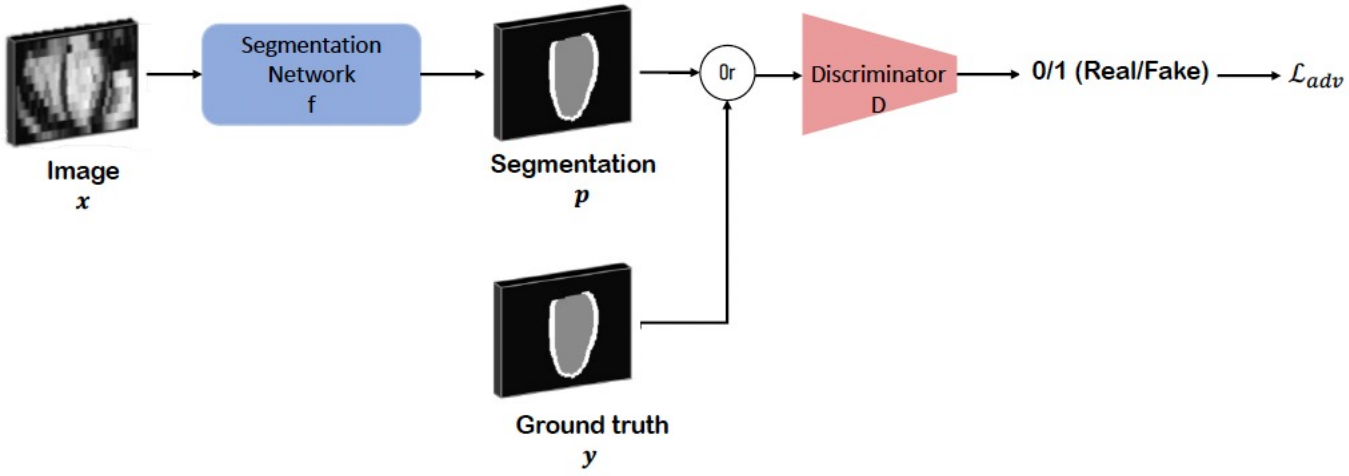


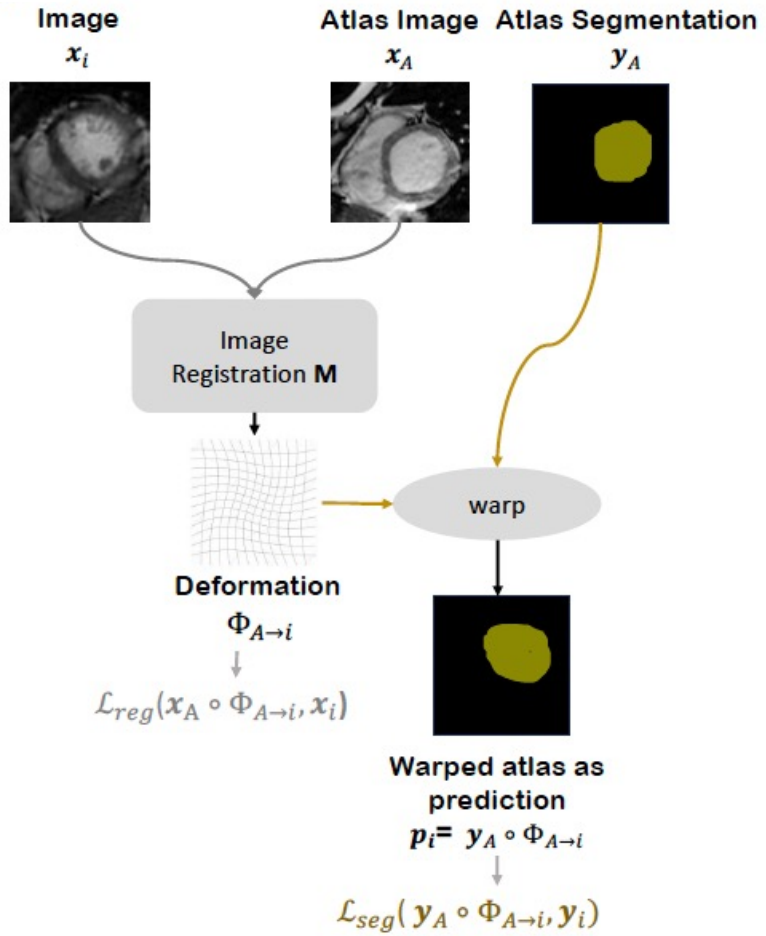
$$\mathcal{L}_{\mathrm{CE}} = -\frac{1}{n}\sum_{i=1}^{n}\sum_{c=1}^{C} y_i^c \log(p_i^c),$$

$$\mathcal{L}_{Dice} = 1 - \frac{2\sum_{i=1}^{n}\sum_{c=1}^{C} y_i^c p_i^c}{\sum_{i=1}^{n}\sum_{c=1}^{C}(y_i^c + p_i^c)}.$$

模型训练

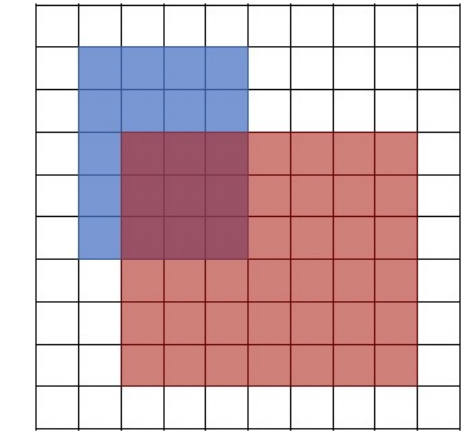
- 加入分割结果的形状先验知识





模型评估

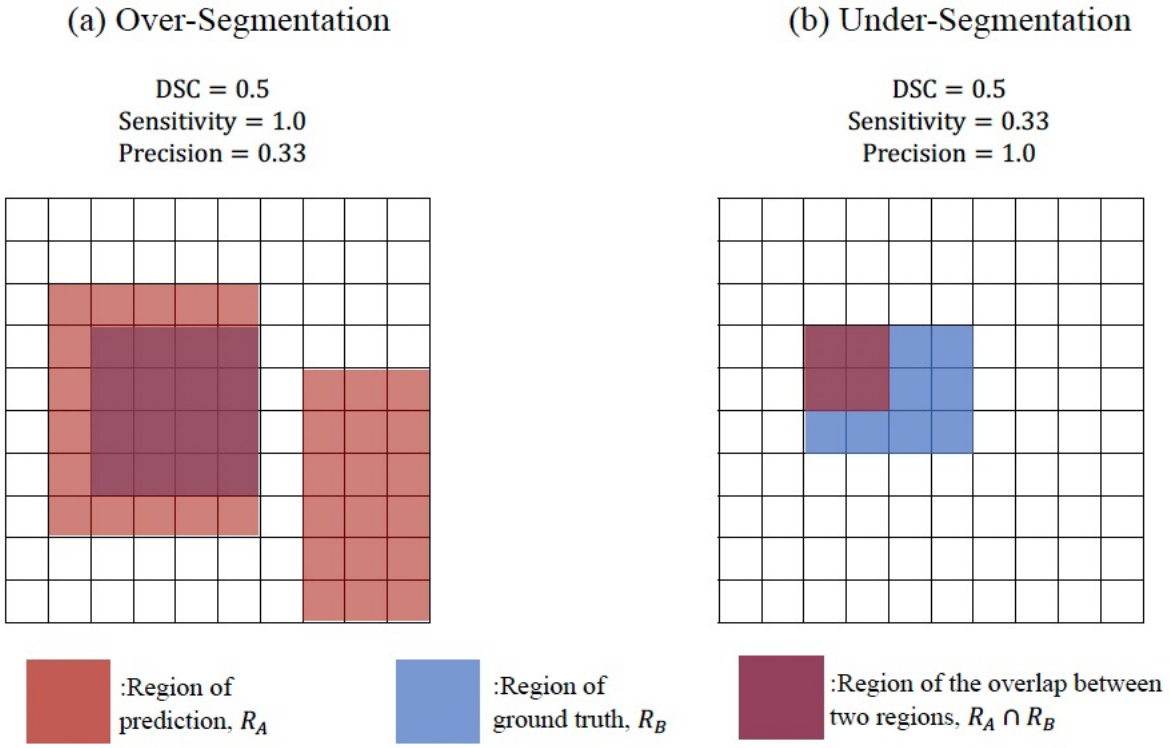
- 基于区域的评估方法



$$\mathrm{DSC}(R_A, R_B) = \frac{2|R_A \cap R_B|}{|R_A| + |R_B|}.$$

模型评估

- 基于区域的评估方法

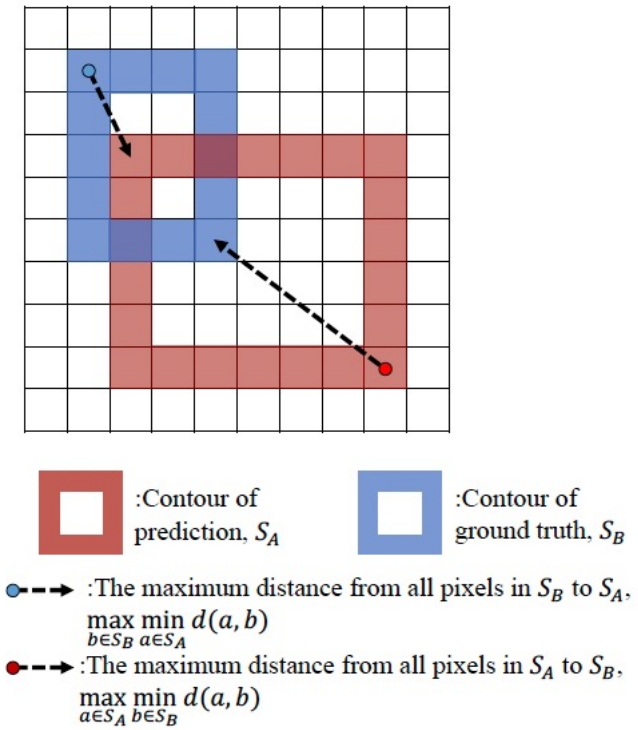


$$\text{Sensitivity}(R_A, R_B) = \frac{|R_A \cap R_B|}{|R_B|}$$

$$\text{Precision}(R_A, R_B) = \frac{|R_A \cap R_B|}{|R_A|}.$$

模型评估

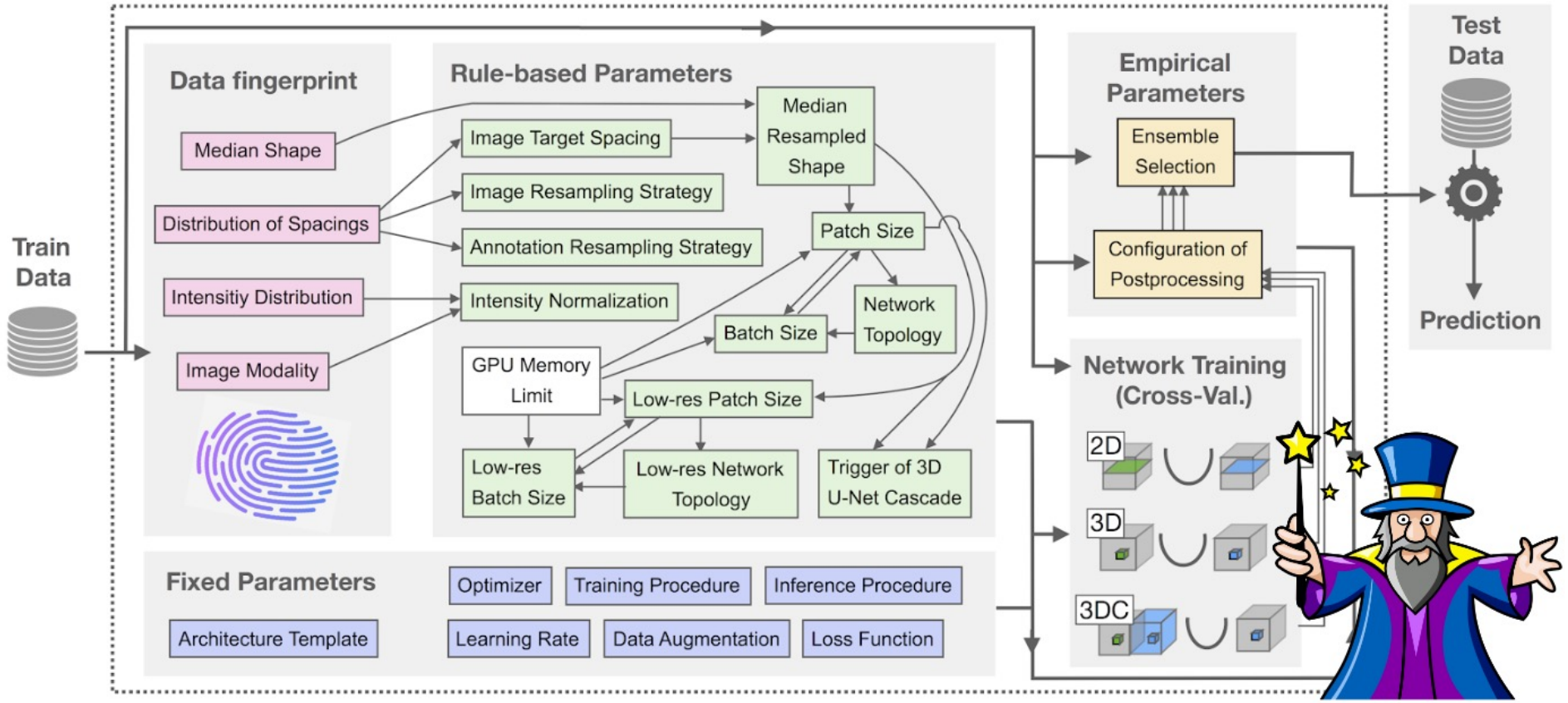
- 基于距离的评估方法



$$\mathrm{HD}(S_A, S_B) = \max\left\{ \max_{b \in S_B} \min_{a \in S_A} d(a,b), \max_{a \in S_A} \min_{b \in S_B} d(a,b) \right\}$$

通用的解决方案

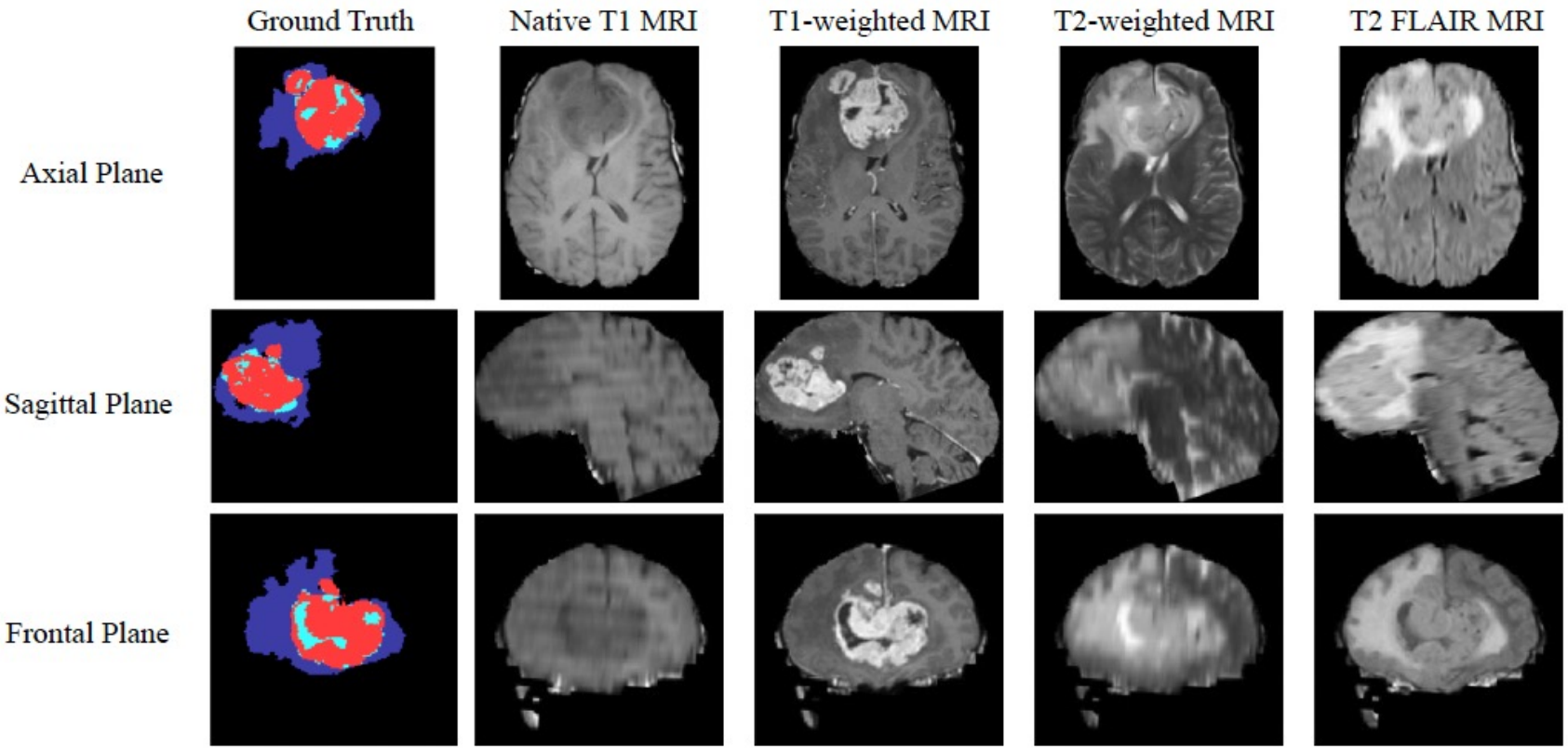
- Q：是否所有任务都可以用同一个框架解决？
- A：大部分可以



F. Isensee, *et al*. “nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation“, Nat Method, 2020.

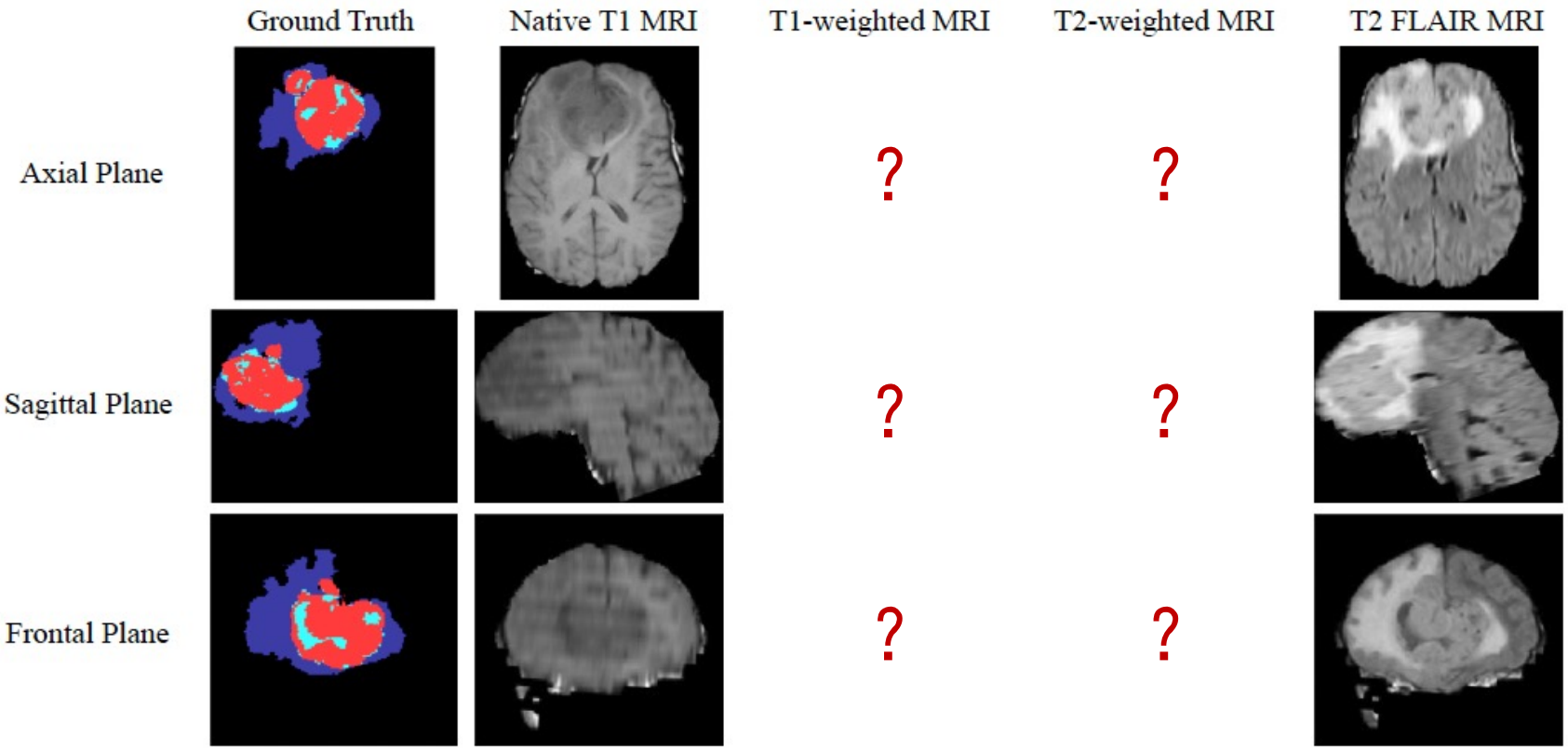
脑部磁共振图像分割

- 磁共振多序列分割



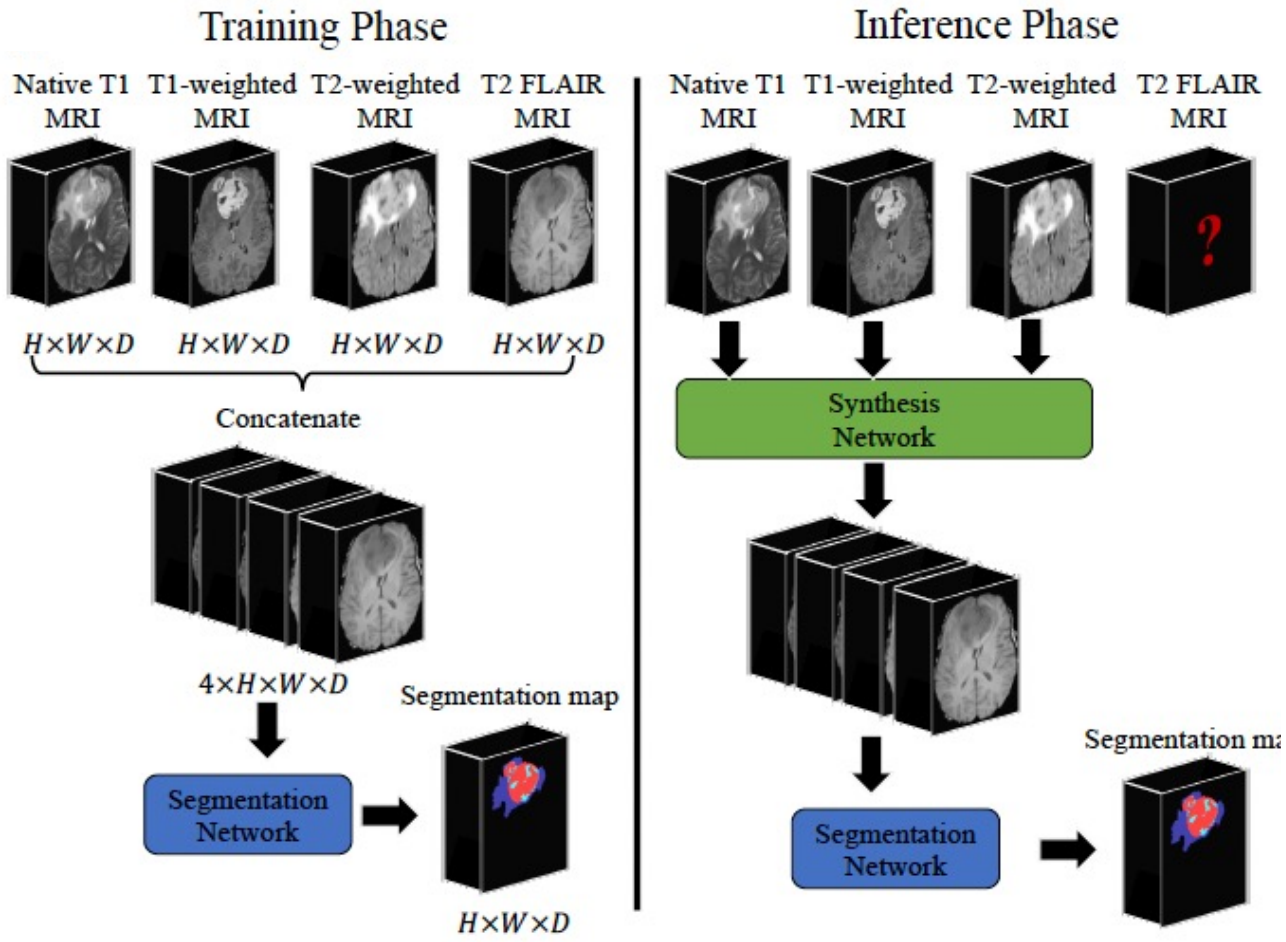
脑部磁共振图像分割

- 问题：测试时**缺少配对序列**



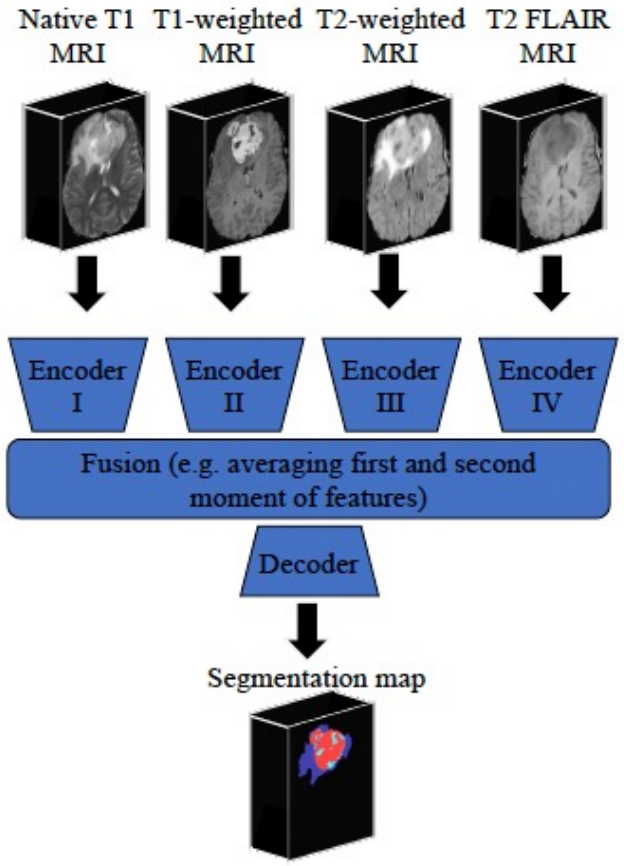
脑部磁共振图像分割

- 基于模态插补的分割方法
- **优势：**
  - 可以利用先进的生成模型技术
  - 从而效果更好



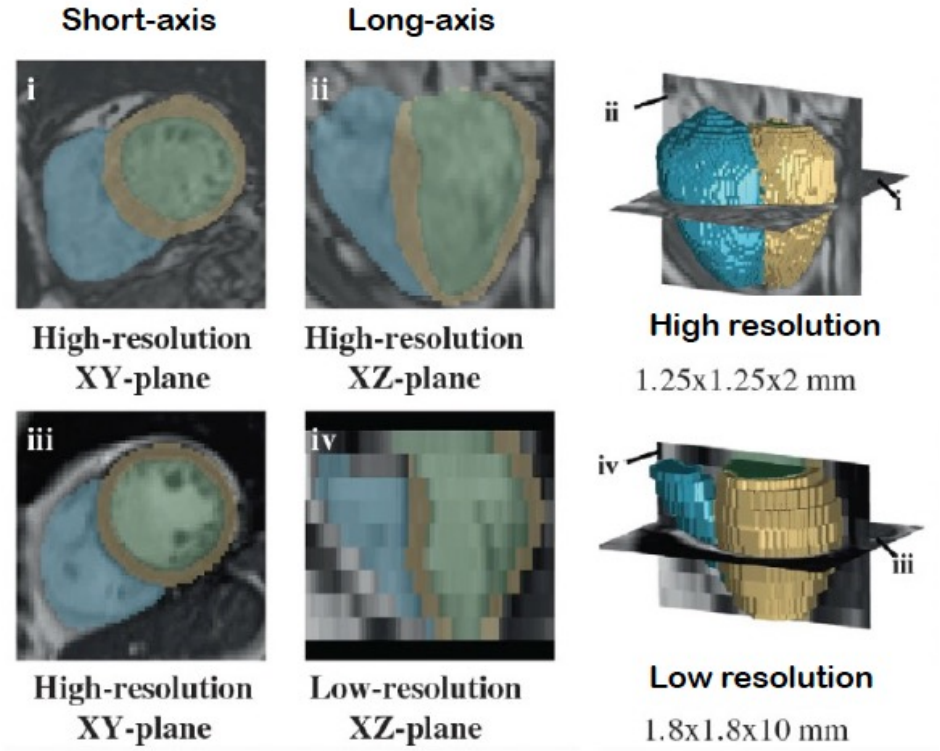
脑部磁共振图像分割

- 基于特征融合的分割方法
- **优势：**
  - 可以处理多种模型丢失情况
  - 从而应用更为灵活



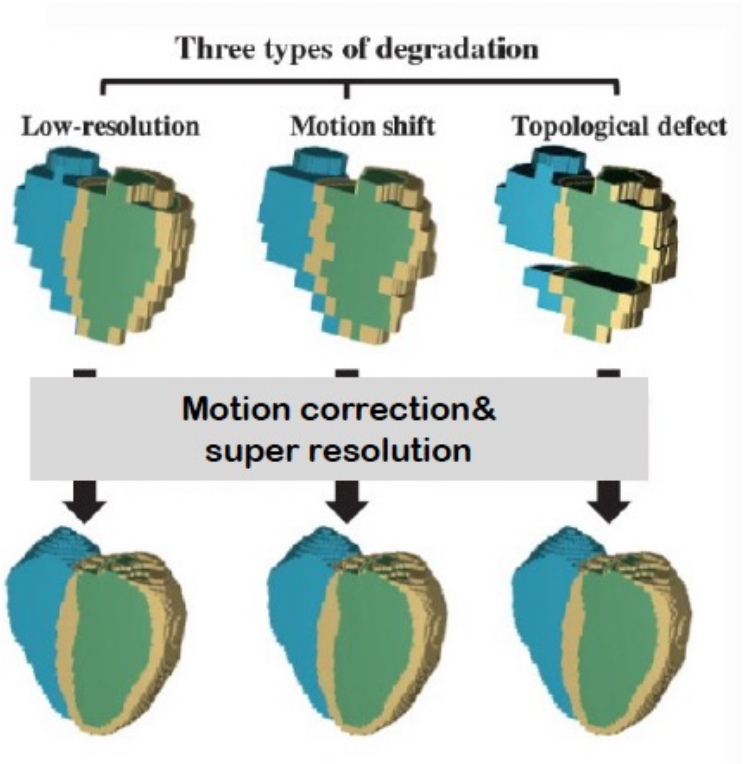
心脏磁共振图像分割

- 心脏在磁共振扫描中变动快，分辨率低



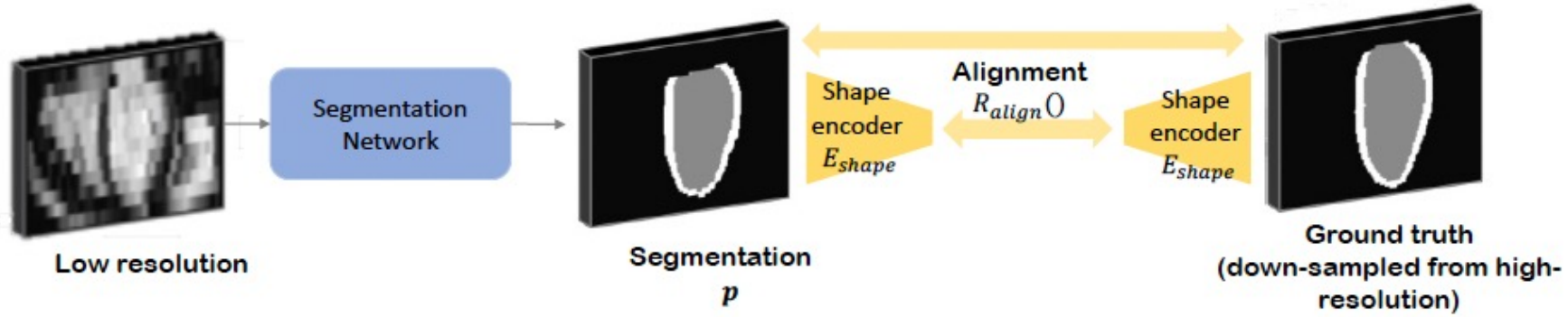
心脏磁共振图像分割

- 问题：心脏**形状**信息容易**丢失**



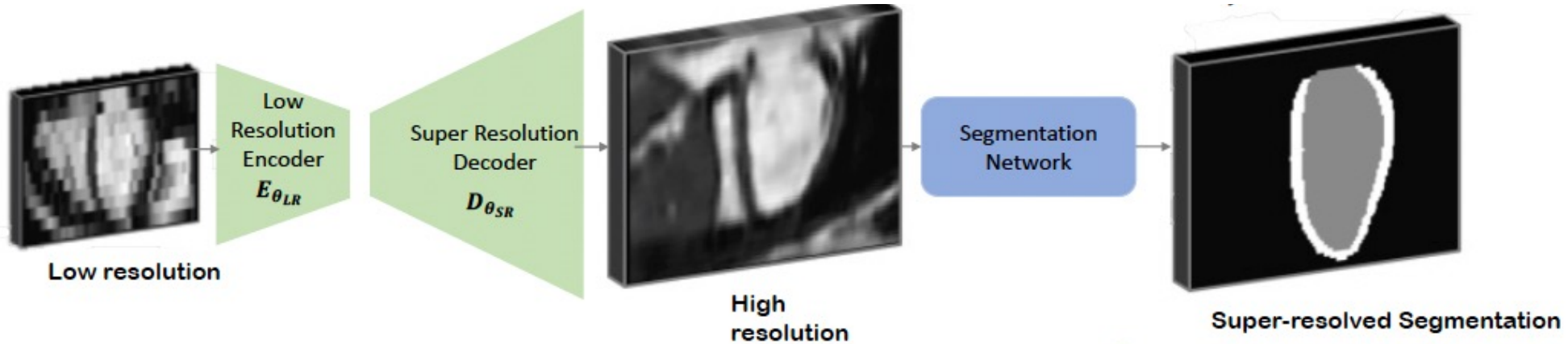
心脏磁共振图像分割

- 基于形状先验的分割方法
- **优势：**
  - 可以轻量化改进分割结果



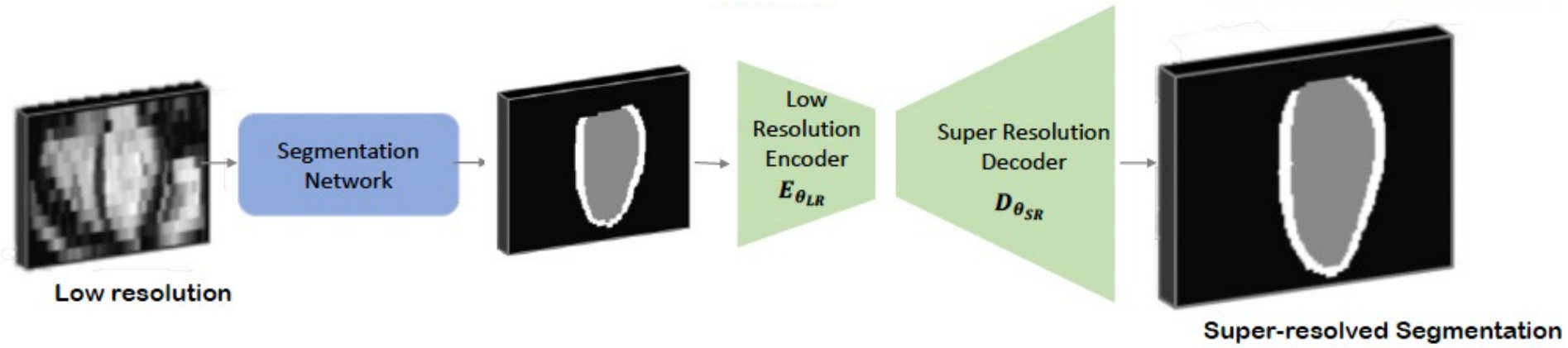
心脏磁共振图像分割

- 基于图像超分辨率的分割方法
- **优势：**
  - 可以利用高分辨率的全部信息



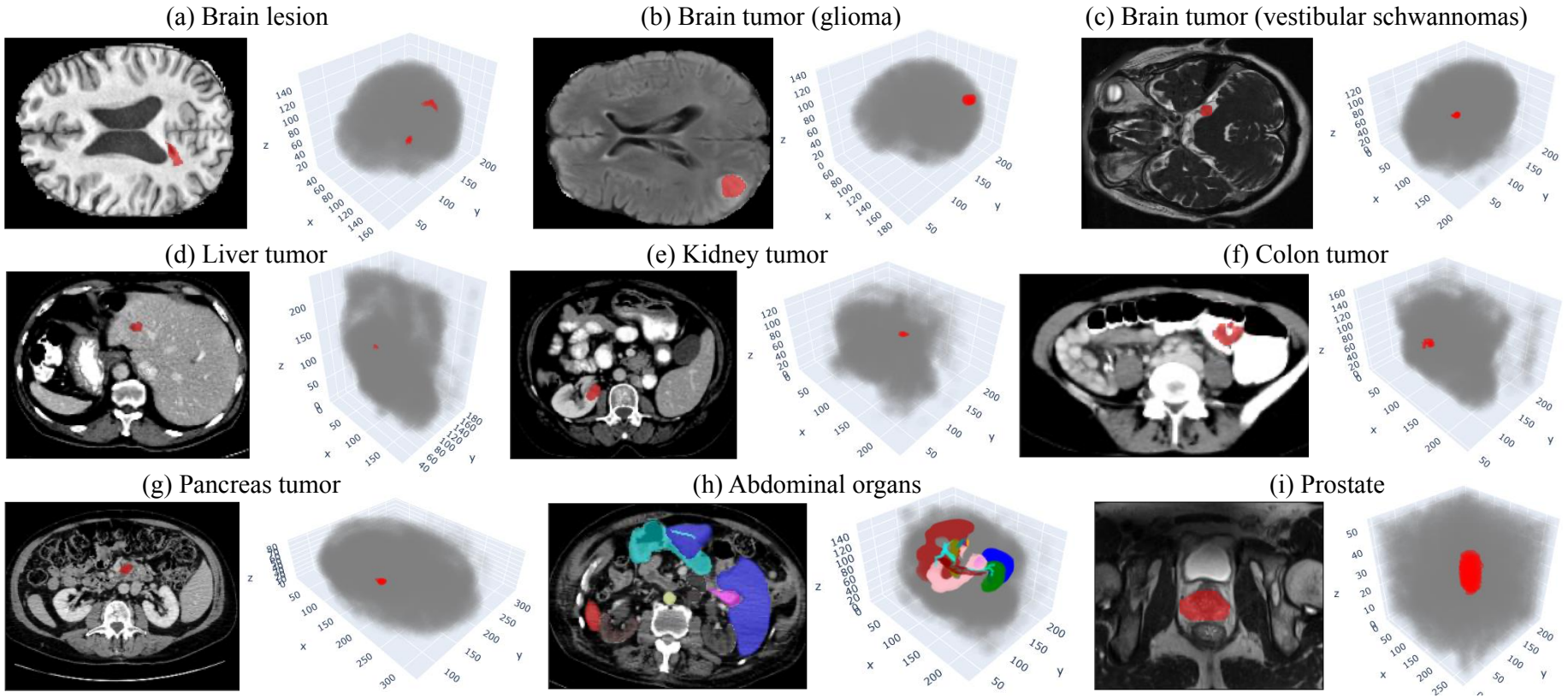
心脏磁共振图像分割

- 基于形状超分辨率的分割方法
- **优势：**
  - 作为一种后处理方法，可以应用于已训练好的分割网络



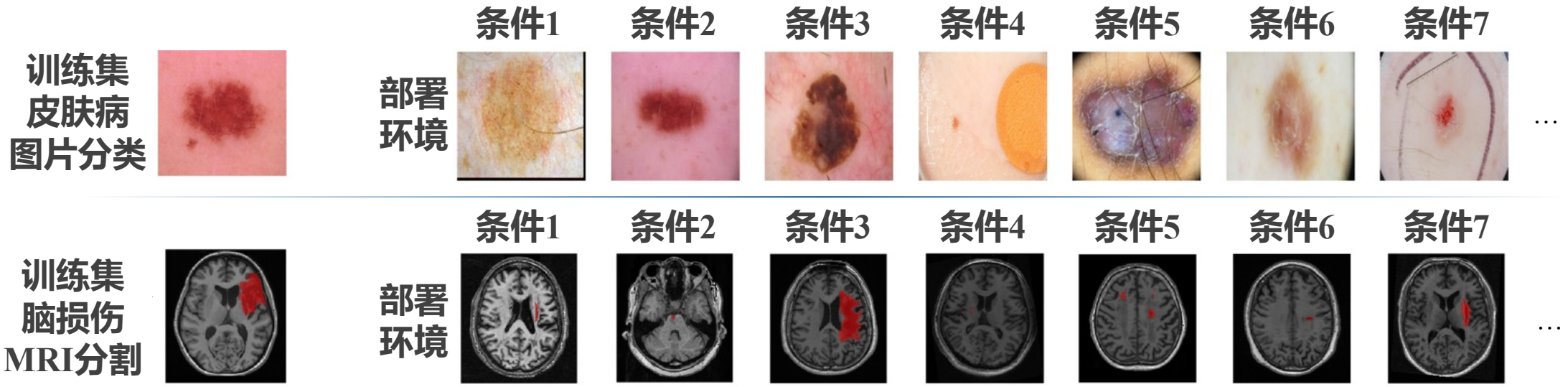
样本不均衡

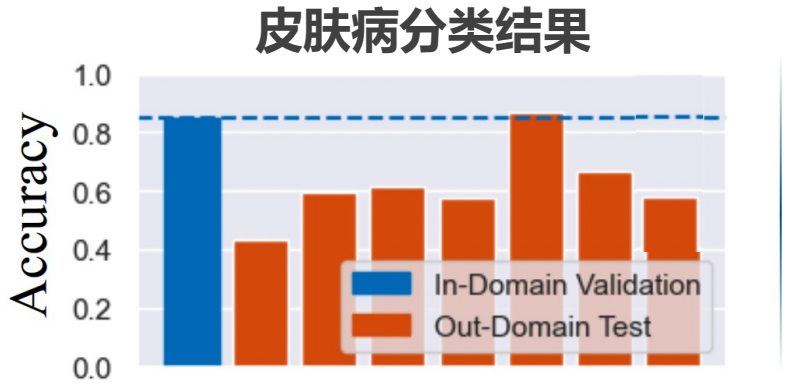
- 样本不均衡是医学图像中的一个常见问题

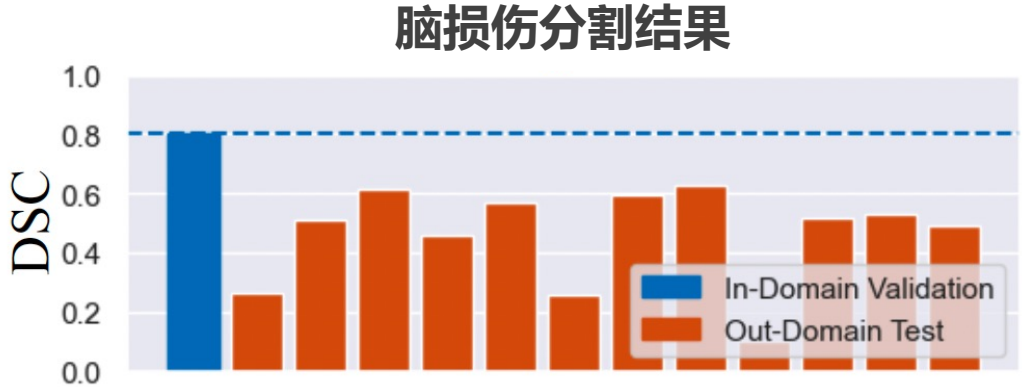


算法泛化性问题

- 因为采集仪器不同，深度学习算法在不同成像条件下泛化性差



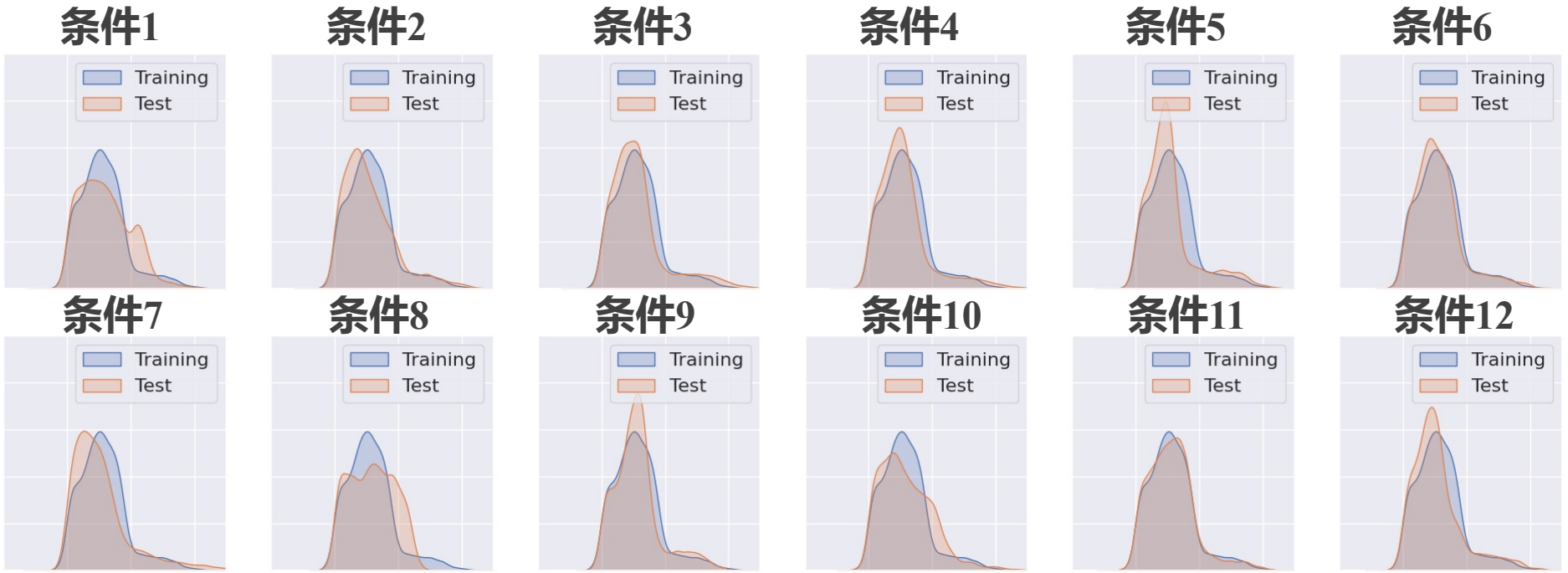


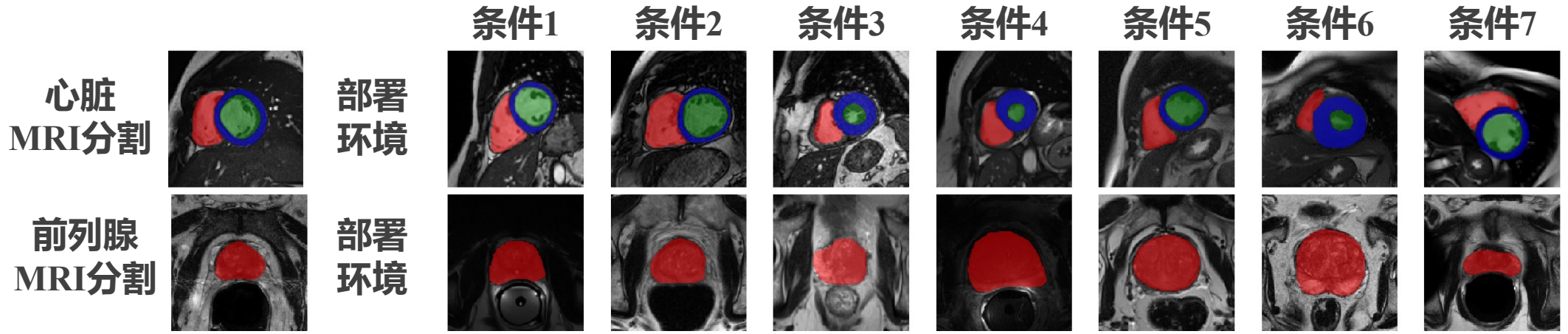


部署时性能下降
且下降程度未知

磁共振成像参数对图像质量影响显著

不同采集条件
脑部MRI
灰度直方图





类别不平衡导致的模型误差累积

目标未对齐 → 模型不可靠

模型表现

部署环境

优化算法

数据分布

可靠性保障

泛化性

模型拟合

小结

- 判别模型和生成模型本质上是一个问题的两个方面
- 高斯过程是一种基于贝叶斯定理的非参数模型，适用于连续空间的建模
- 深度学习的核心结构源于多层感知机
- CNN是一种通过卷积核进行局部连接和参数共享的特化神经网络
- 医学图像分割是医学图像分析领域的核心任务，主流方法基于深度学习技术