自动驾驶汽车的决策规划技术（Part II）

李升波

清华大学·车辆与运载学院

Email: lishbo@tsinghua.edu.cn

总体概要

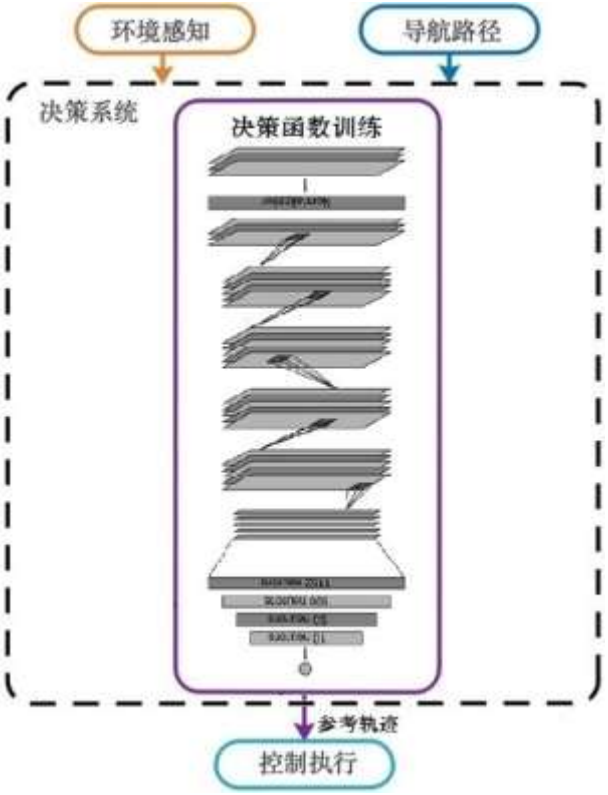
1 决策规划技术概述

2 分解式决策方案

3 集中式决策方案

两类基本决策方案

	分解式	集中式
定义	将决策过程分解为独立的子问题，如：场景理解、运动预测、行为选择、路径规划等，每一个问题独立解决，然后功能组合。	以道路环境的感知结果为输入，以期望路径或执行器控制命令为输出，将决策过程视作一个不可分解的黑箱。



两类基本决策方案

	分解式	集中式
方法	排序、搜索、优化、浅层学习等	深度学习、强化学习等
代表	谷歌、百度、通用、福特、Tesla 等	英伟达、Wayve 等
优点	✓ 问题可分解、任务可分工 ✓ 节省车载存储和计算资源 ✓ 决策代码的开发可控性好	✓ 体系框架简洁明了 ✓ 环境感知信息无损失 ✓ 几乎无需手工标记数据
缺点	x 感知信息存在损失 x 涵盖场景\行为有限 x 决策目标制定困难 x 场景间难以互相迁移	x 难以嵌入已知驾驶经验 x 算法难以理解与手动改进 x 算法入门艰难，开发难度高

两类基本决策方案

分解式决策的发展挑战

- 难以应对“混杂”、“博弈”交通场景

- 不能应对尚未见过的“未知”交通场景

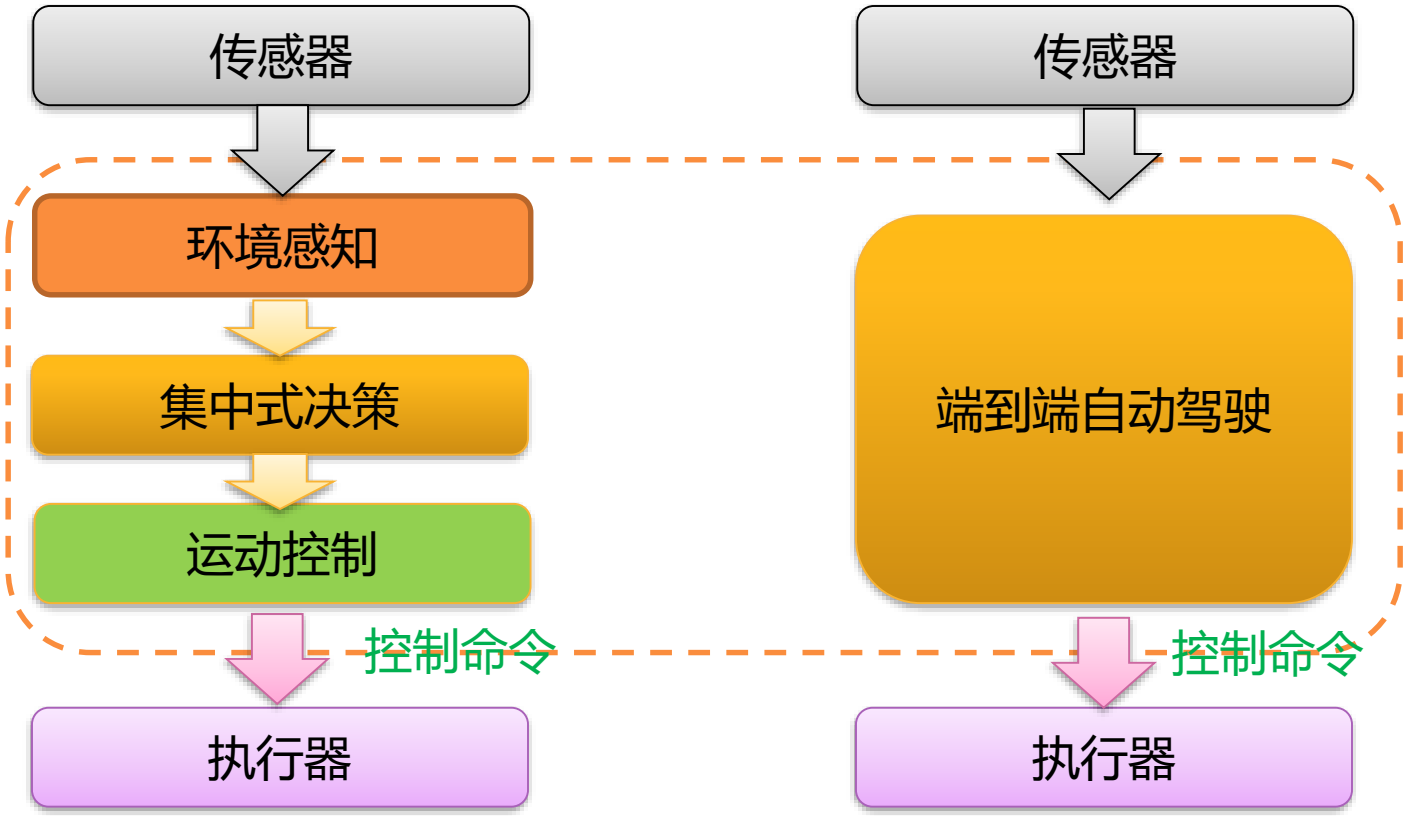
Uber自动驾驶

特斯拉AutoPilot

集中式决策 vs 端到端自动驾驶

集中式决策

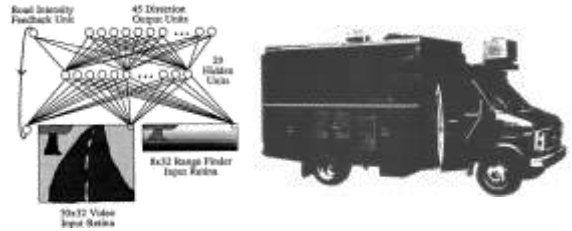
端到端自动驾驶



广义上说，集中式决策与端到端自动驾驶是不分家的。
二者所用方法是类似的！

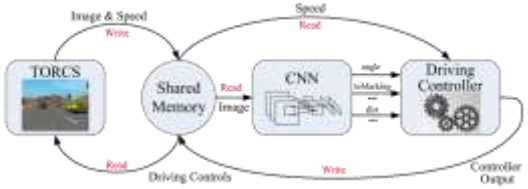
集中式决策方案的发展历程

浅层神经网络 *Pomerleau



CMU 校园静态场景(低速)

深度神经网络 *Chen



Princeton TORCS平台

深度强化学习 *Mnih

DeepMind TORCS平台

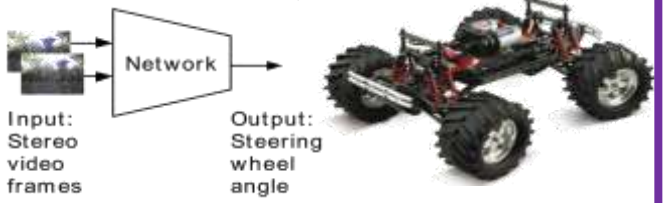
1989

2004

2015

2016

卷积神经网络 *Lecun



DARPA 静态越野场景

深度神经网络 *Bojarski

NVIDIA 高速公路

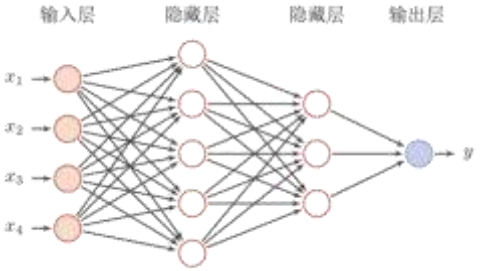
集中式决策方案：监督学习型+强化学习型

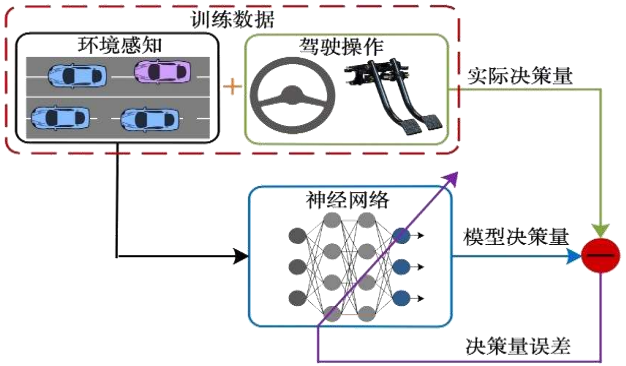
两类集中式决策方案

监督学习型

Kozakiewicz et al.，1991；
Pal P K et al.，1995；
Yang S X et al.，2000

依赖大量驾驶员数据，但不能利用车辆模型和驾驶经验等信息

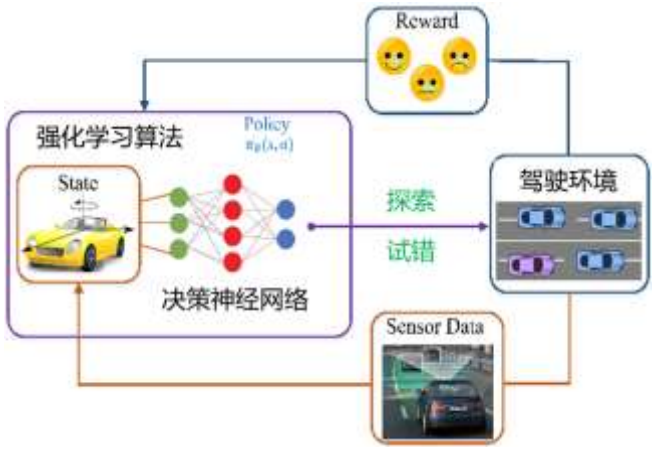


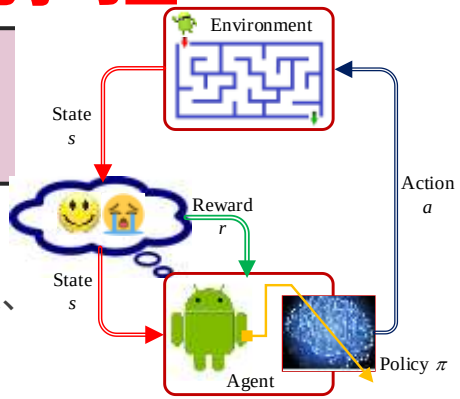


强化学习型

Fares A et al.，2014；
Mnih V et al.，2015；
Sallab A E et al.，2017；
Wang Pin et al.，2017

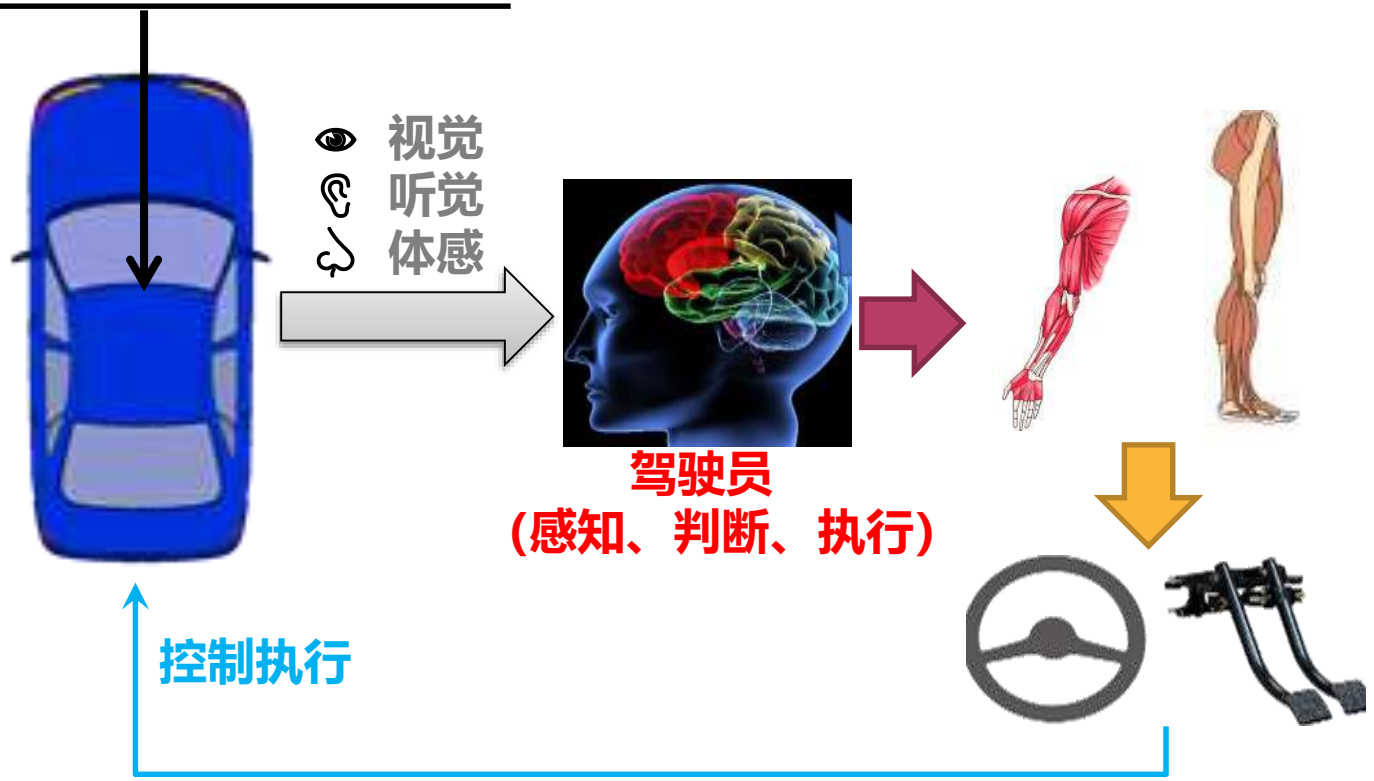
具有嵌入驾驶经验等能力，但训练速度慢、安全性差。





集中式决策：监督学习型

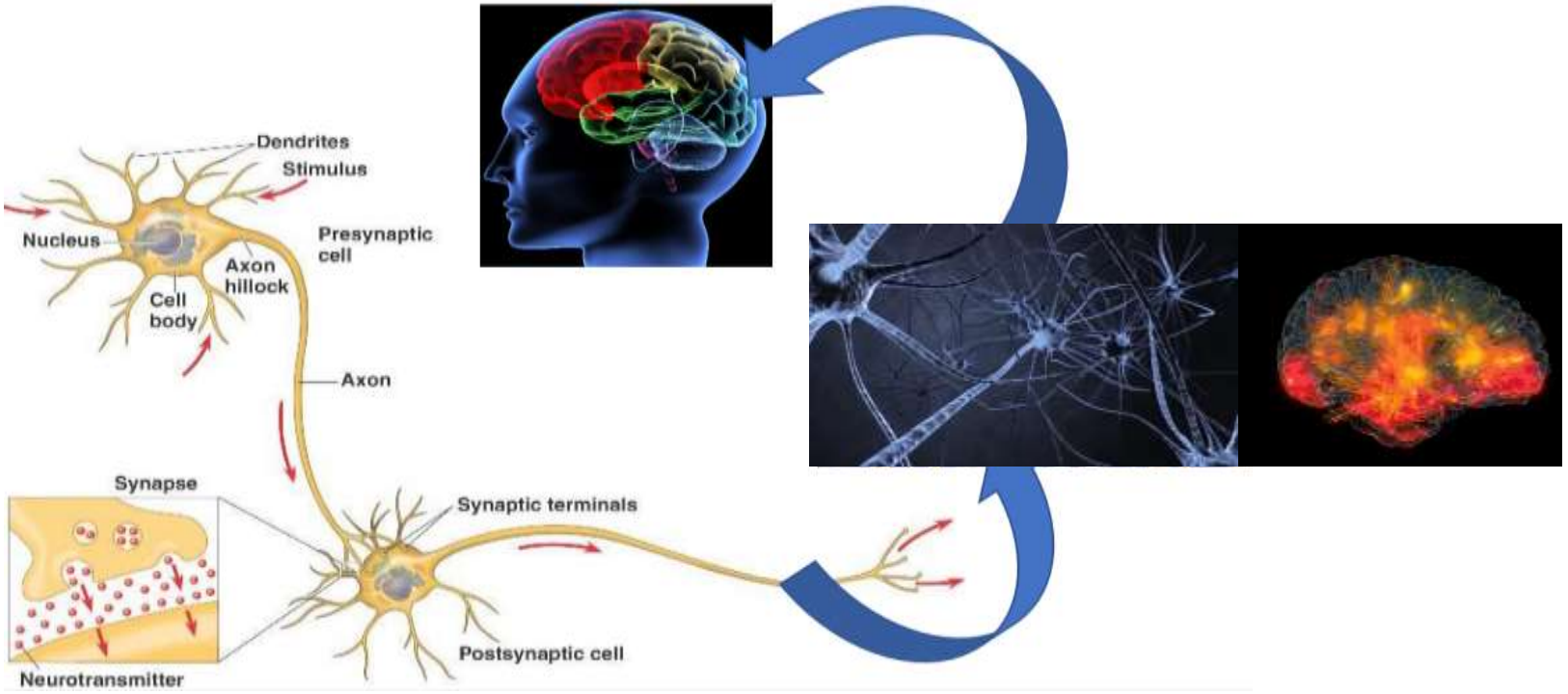
人类的驾驶原理



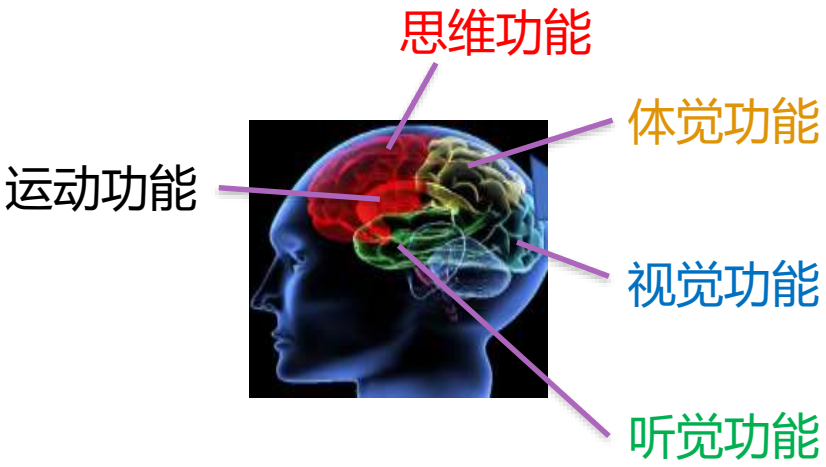
人类的驾驶原理

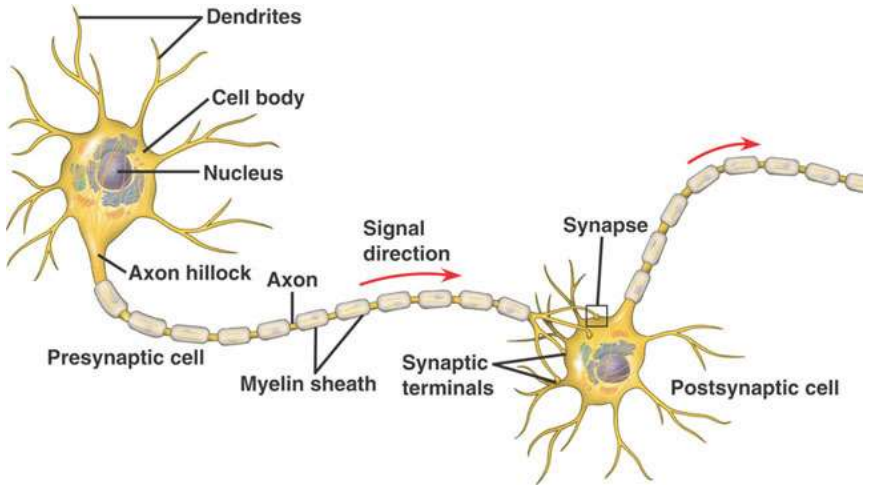
人脑

- 大量（约850亿）相互连接的神经元
- 每个神经元平均约 10^4 个连接
- 单一神经元的反应速度0.5毫秒，恢复时间约5毫秒
- 通过改变神经元之间突触连接的有效性进行学习



人类的驾驶原理

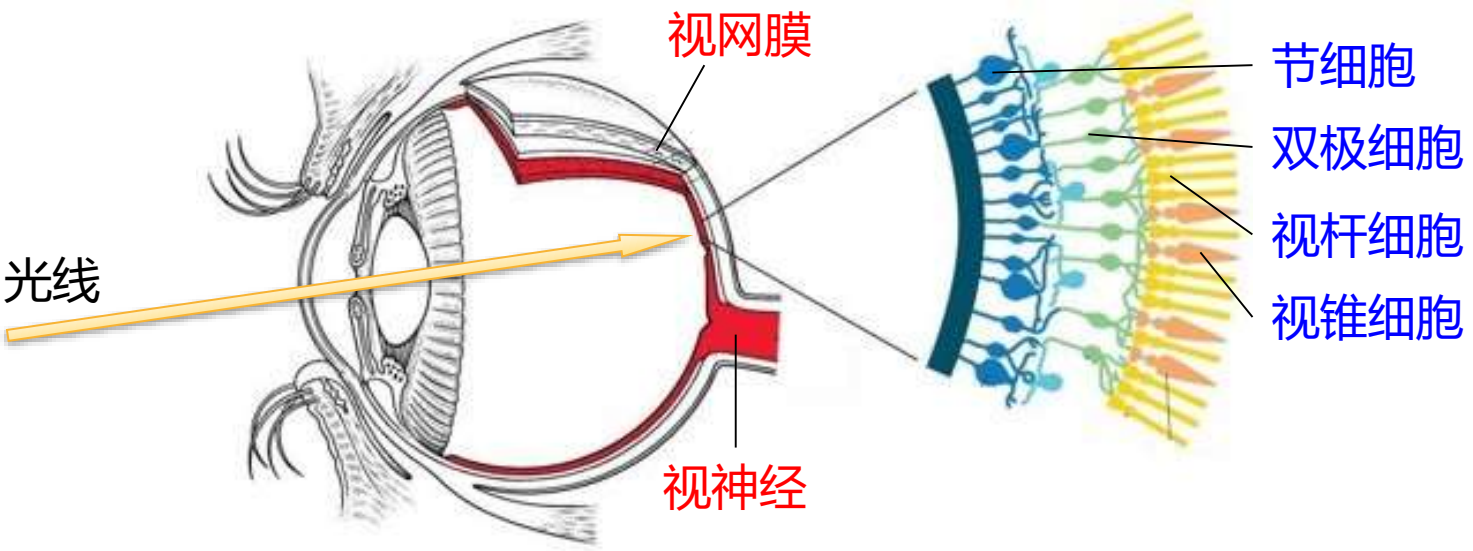




人脑特点

- 大脑功能：视觉、听觉、体觉、思维（决策、判断、规划）等
- 小脑功能：微调运动控制命令，达到准确性，协调性和连贯性
- 神经元内部是电流传导；神经元之间接触点由化学物质传导
  - A类神经纤维（感知/运动）传递速度：5~120m/s。
  - B类神经纤维（呼吸/内脏）传递速度：3~15m/s。

人类的驾驶原理



视网膜的感光处理

- 双极细胞及节细胞的中心：周围感受野。
- 夜晚时视锥细胞活力较差，立体感降低。
- 大约1.3亿个光感受器接受光信号，通过120万个节细胞轴突将信息从视网膜传递到大脑。
- 人眼最佳分辨率约等价于576000000（五亿七千万）像素的相机。

人类的驾驶原理

肢体执行能力

- 反应速度由神经反射通路的传导速度所决定
- 正常人反应速度：0.2~0.3s
- 单一驾驶动作周期：0.5~0.7s
- 经验形成肌肉记忆（+小脑/脊柱反馈），提高反应速度

手被烫反应时间：0.15s

紧急制动时间 ~ 0.7s

(反应时间 ~ 0.4s + 制动响应 ~ 0.3s)

多选题 1分

设置

请你大胆猜测：对比人类神经系统和机器神经网络，下列那一些叙述是正确的呢？

A 机器神经元的运算速度约是人类神经元的五千倍

B 最大机器神经网络的规模是人类大脑的百分之一

C 单一机器神经元的功耗约为人类神经元的一亿倍

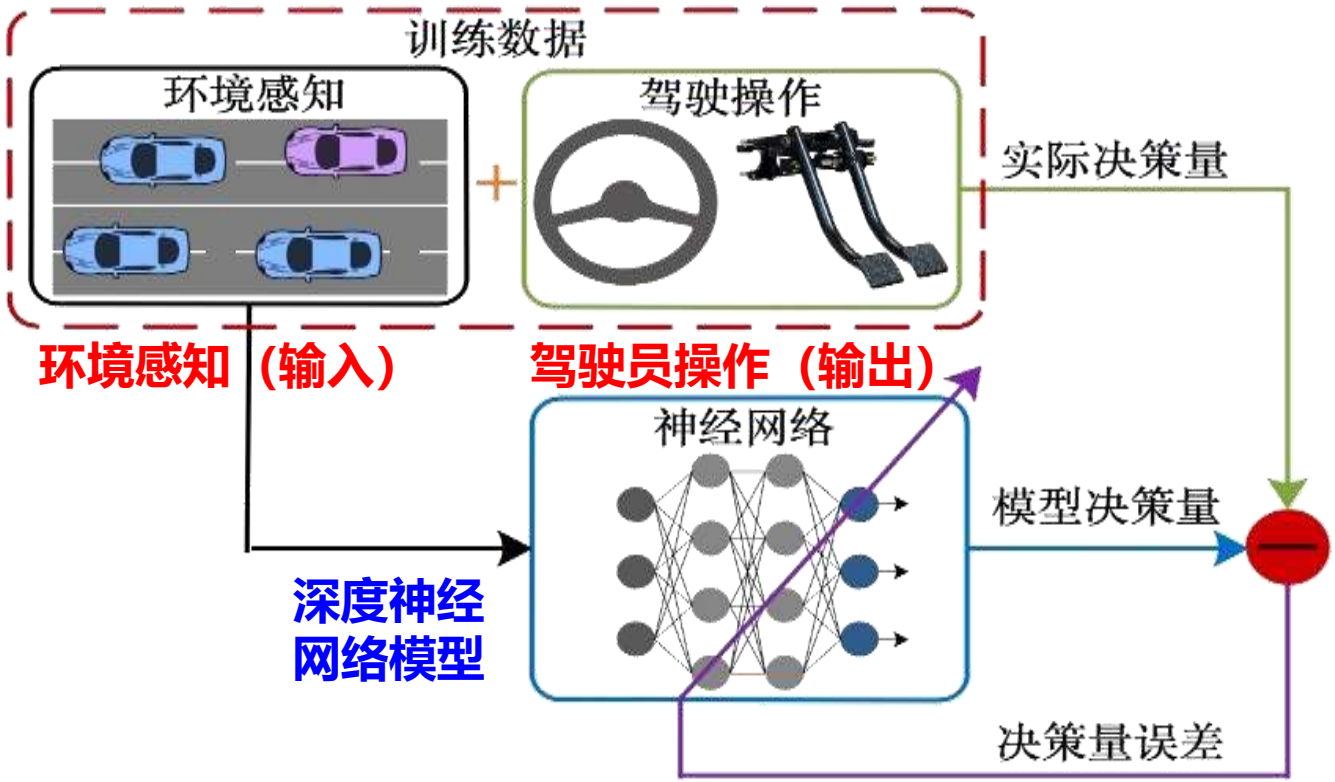
D 人类神经系统比机器神经网络的总体效率高二十万倍

提交

监督学习型决策

基本原理

- 利用环境感知信息（输入）以及驾驶员操作（输出）数据的训练深度神经网络模型
- 本质：模仿优秀驾驶员的驾驶经验



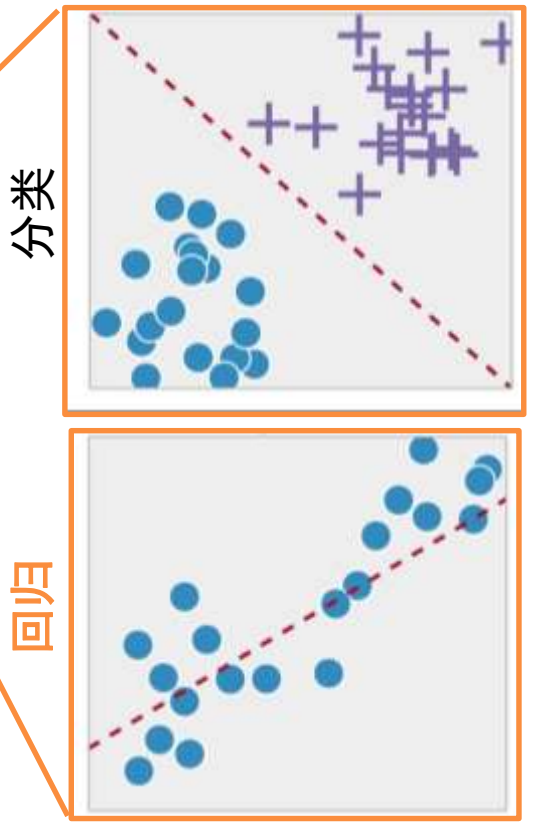
监督学习简介

监督学习的基本原理

- 给定驾驶状态 s 的驾驶数据标签 $D(s)$
- 求解**最优策略 π** ，**最小化**策略输出 $D^{\pi}(s)$**与标签数据的差异**

$$\min_{\pi} \|D^{\pi}(s) - D(s)\|, \forall s \in \mathcal{S}$$

$$\pi^*(s) \quad ?$$



监督学习简介

监督学习主要步骤

- 选取策略模型

$$\pi(s;\ \theta) \rightarrow D^{\pi}(s)$$

- 确定目标函数

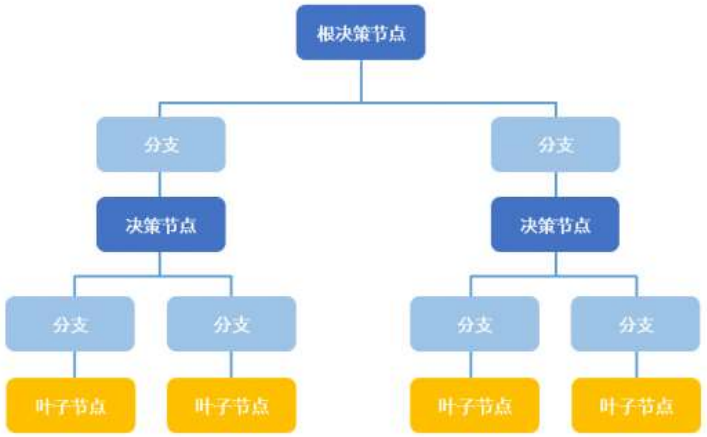
$$J(s,\theta) = \left(D^{\pi}(s) - D(s)\right)^2$$

- 迭代更新，求解策略参数（如：梯度下降）

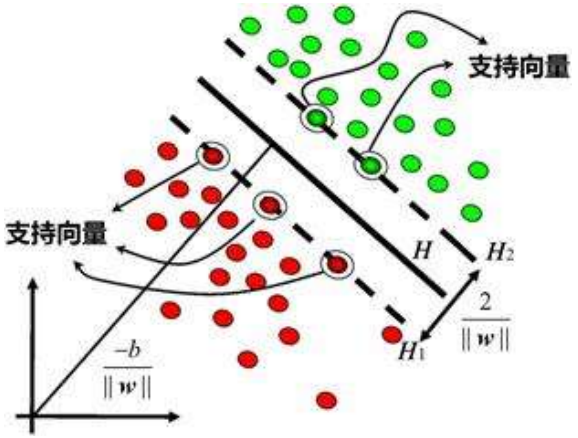
$$\theta \leftarrow \theta - \lambda \frac{\partial J(s,\theta)}{\partial \theta}$$

监督学习简介

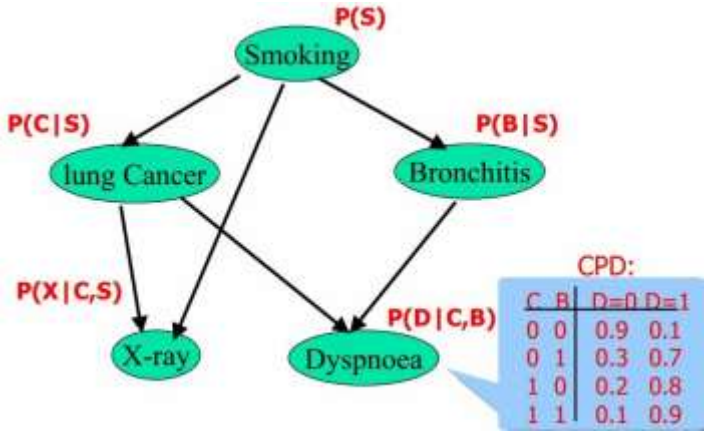
典型监督学习方法（分类+回归）



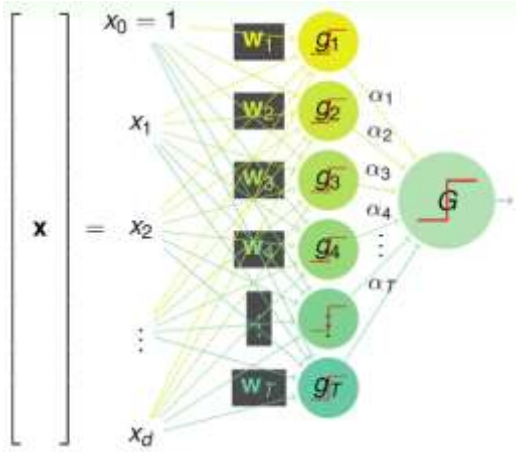
决策树（DT）



支持向量机（SVM）



贝叶斯网络（Bayesian）



神经网络（NN）

监督学习型决策示例（早期）

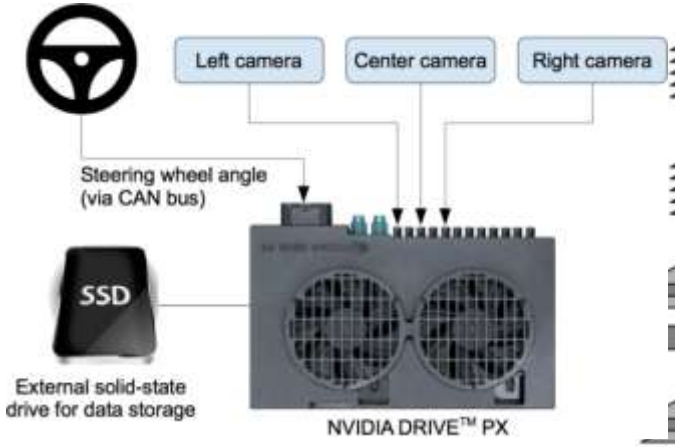
- **ALVINN (1990)**
  - Pomerlau：CMU
  - 输入 = 低分辨率图像
  - 全连接神经网络

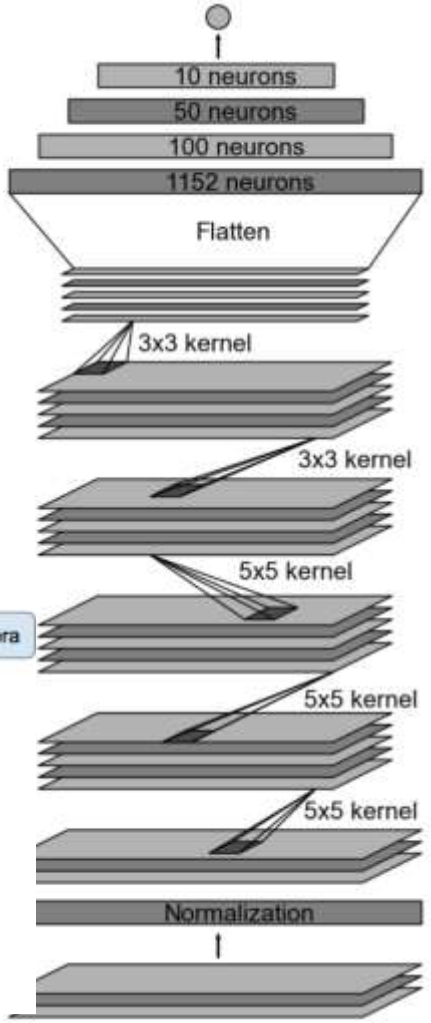
- **DAVE (2003)**
  - Muller 和 LeCun
  - 输入 = 高分辨率图像
  - 卷积神经网络

监督学习型决策示例（英伟达）

- **案例：英伟达**
  - **模型：CNN（卷积神经网络）**
  - **数据：前向摄像头图像→方向盘转角**

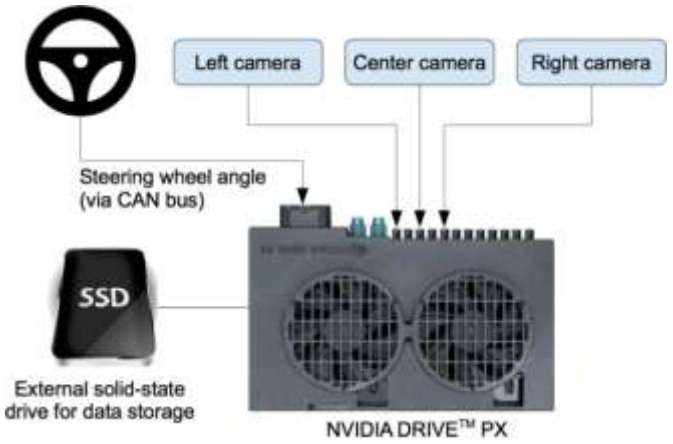
- **训练过程**
  - **训练数据：72小时的驾驶数据（10Hz）**
  - **测试结果：州际公路10英里的自动驾驶**

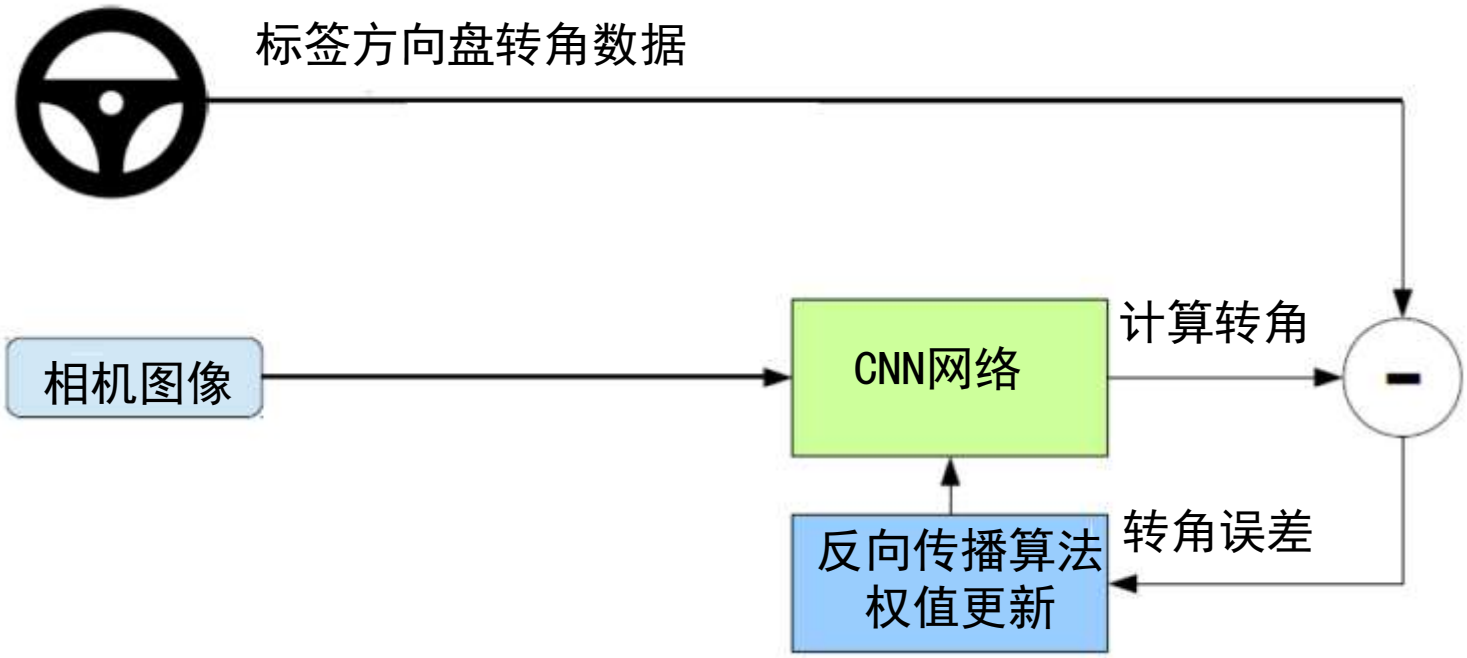




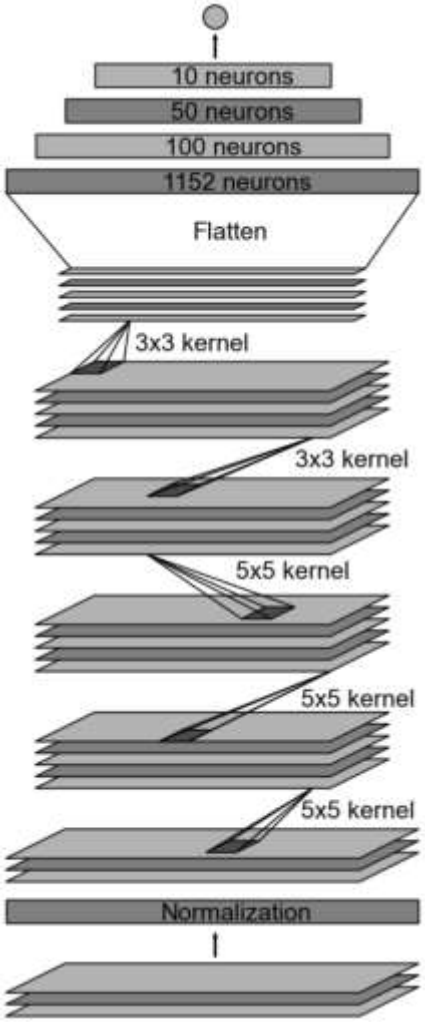
监督学习型决策示例（英伟达）

- **神经网络训练过程**
  - **训练数据：72小时的驾驶数据（10Hz）**
  - **测试结果：州际公路10英里的自动驾驶**





监督学习型决策示例（英伟达）



输出转角

全连接层

展平

卷积特征图，64@1X18

卷积特征图，64@3X20

卷积特征图，48@5X22

卷积特征图，36@14X47

卷积特征图，24@31X98

输入正则化，3@66X200

输入层：连续3张66X200图片（3@66X200）

PilotNet （约25万个权值）

监督学习型决策示例（英伟达）

- 神经网络关注区域

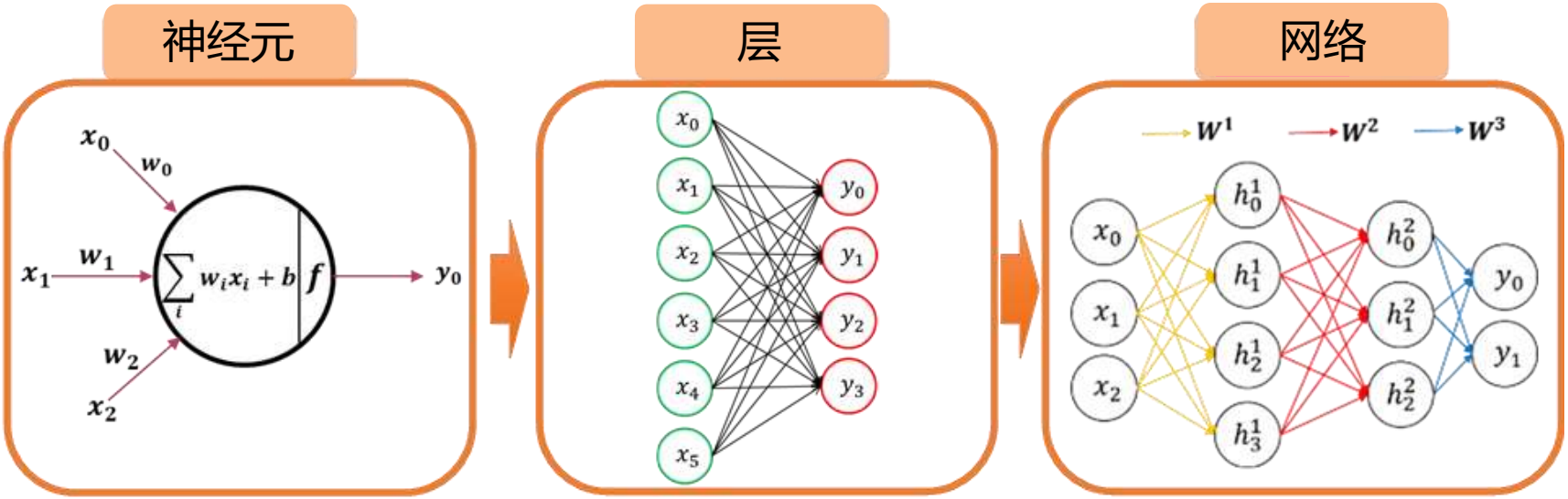
监督学习型决策示例（英伟达）

- **案例：英伟达（NVIDIA）**

监督学习：神经网络的结构

分层结构

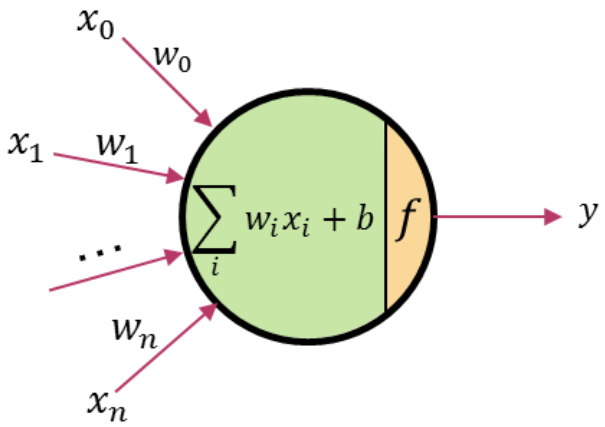
- 神经元（Neuron）：神经网络的基本计算单元
- 层（Layer）：神经元宽度方向的组合
- 网络（Network）：沿深度方向连接图层



监督学习：神经网络的结构

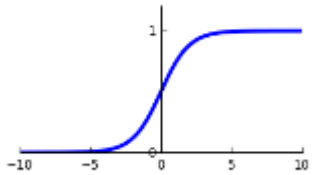
神经元

- （1）线性映射
  - 线性函数的叠加仍然是线性的
- （2）激活函数

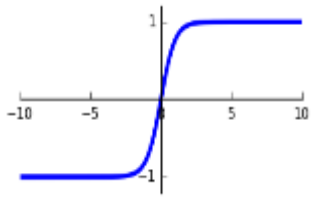


激活函数

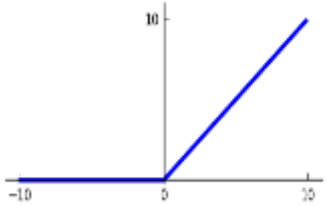
- Sigmoid $\sigma(x) = \dfrac{1}{1+\mathrm{e}^{-x}}$



- Tanh $tanh(x) = \dfrac{\mathrm{e}^{x}-\mathrm{e}^{-x}}{\mathrm{e}^{x}+\mathrm{e}^{-x}}$



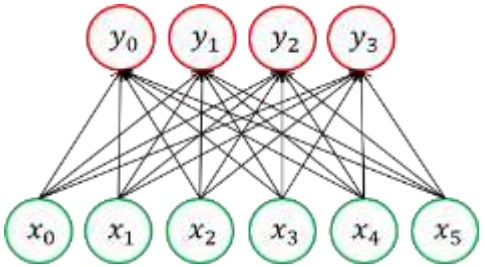
- ReLU $f(x) = \begin{cases} 0, x \le 0 \\ x, x > 0 \end{cases}$



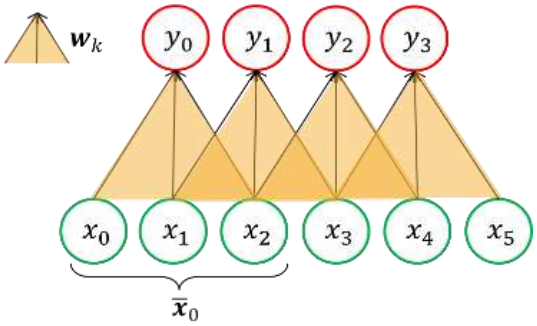
监督学习：神经网络的结构

层数

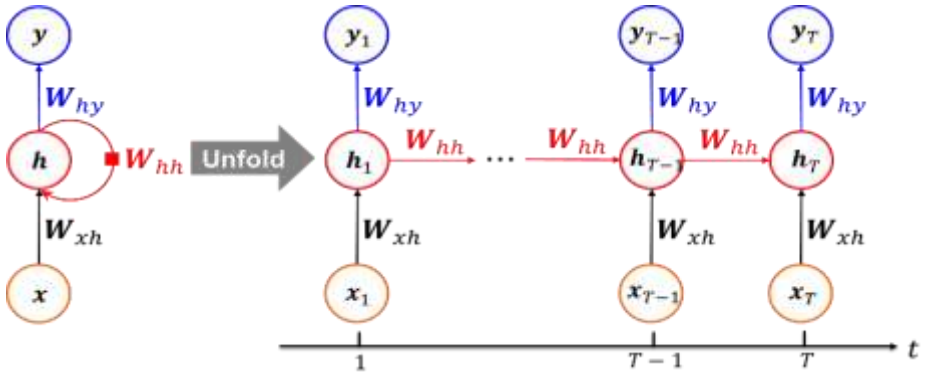
- 全连接（FC）层



- 卷积（COVN）层

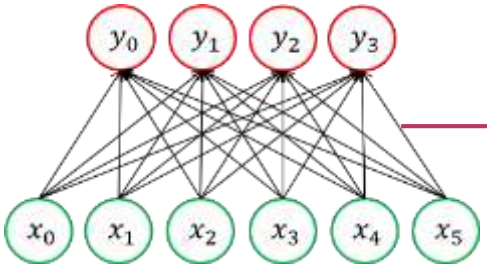


- 循环（REC）层



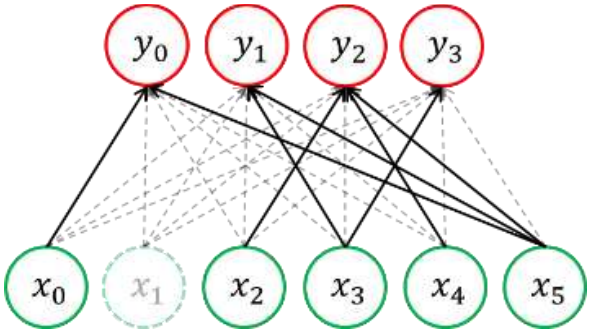
监督学习：神经网络的结构

全连接（FC）层



可训练的权重

- “Dropout”解决过度拟合问题（Hinton，2012年）
  - 随机移除一些神经元
  - 随机移除一些链接

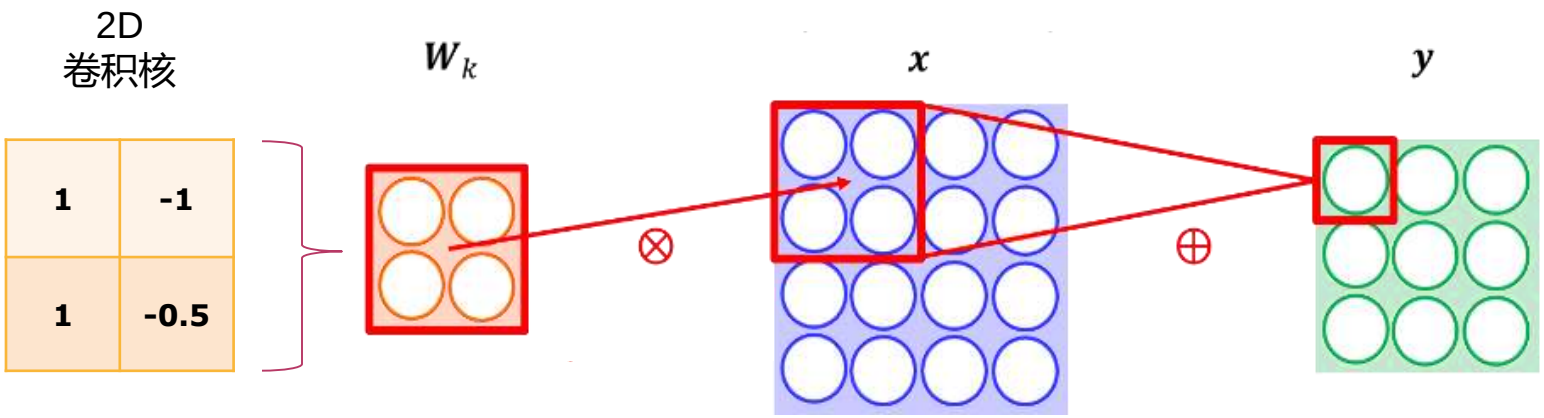


稀疏连接层

监督学习：神经网络的结构

卷积（CONV）层：共享一组共用权重的FC

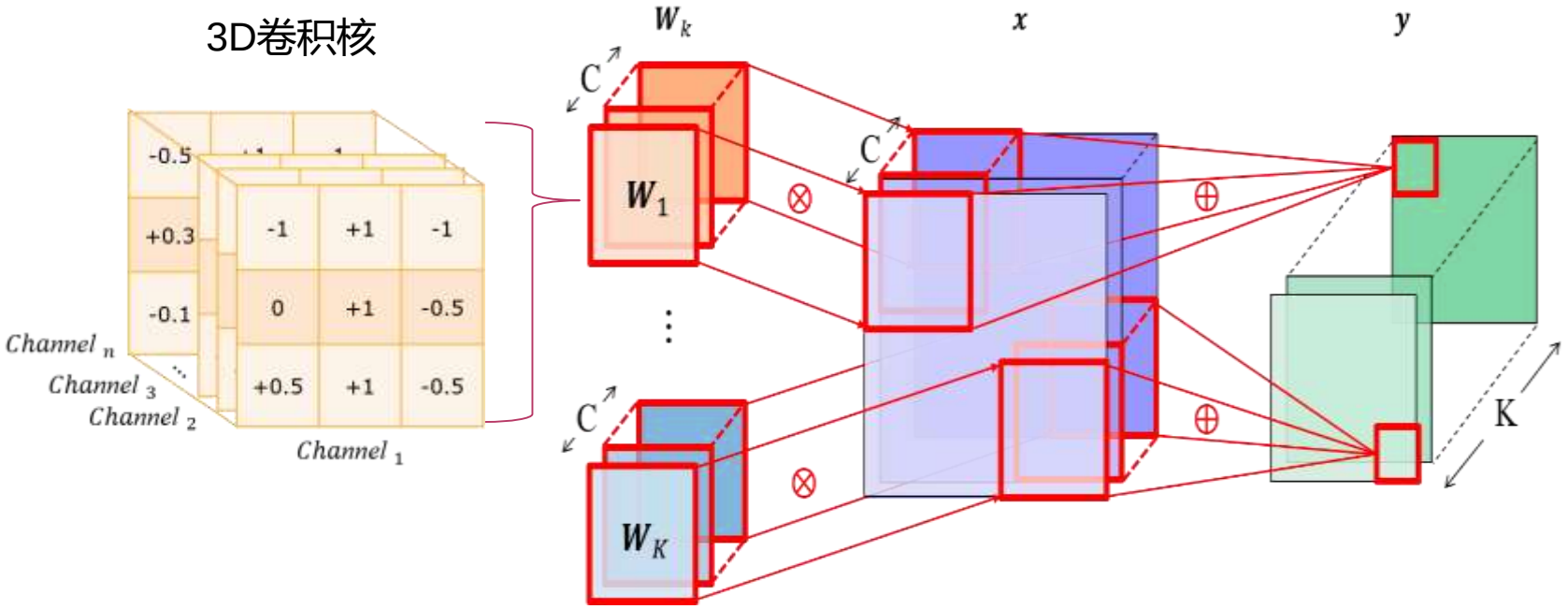
- CONV的卷积核= FC的可训练权重
- CONV的特征映射= FC的神经元层
- 卷积核被整个特征映射所共享
- 与FC相比，极大地减少了权重的数量



（a）单通道2D卷积核

卷积（CONV）层

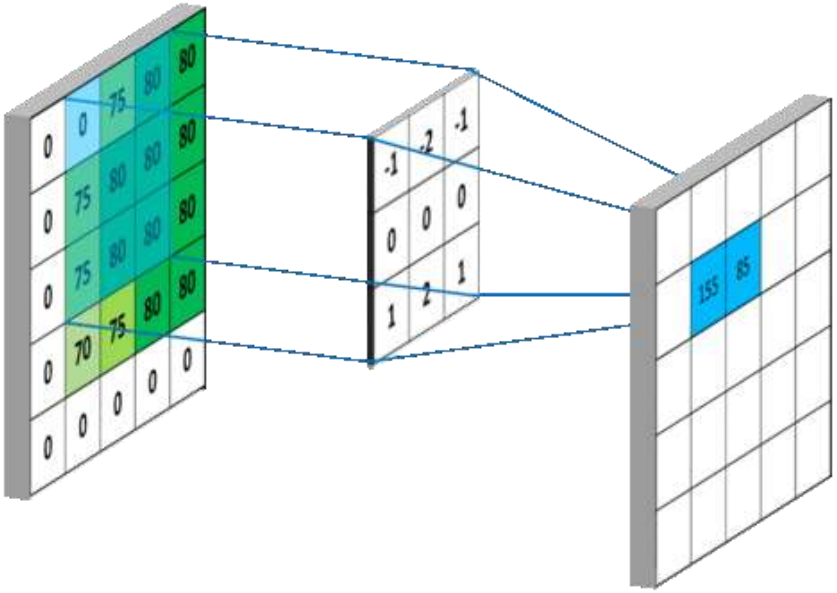
- 输出特征映射的通道（K） = 卷积核的数量
- 输入特征映射的通道（C） = 卷积核的层数



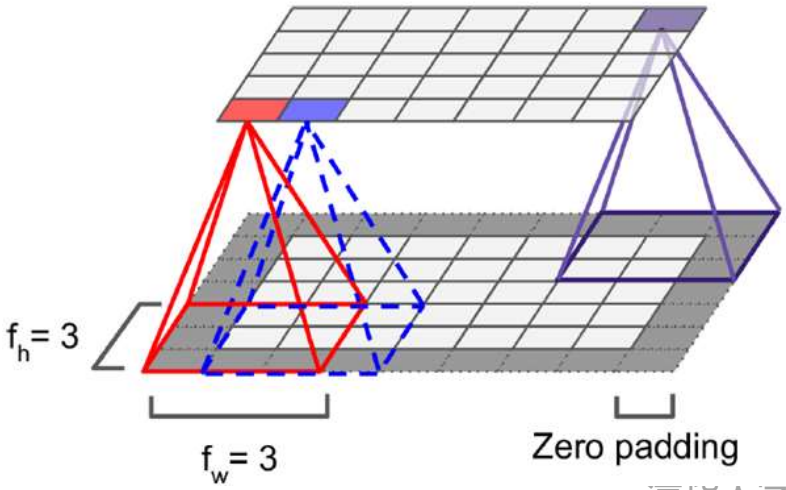
（b）多通道3D卷积核

监督学习：神经网络的结构

卷积层的关键参数



- 卷积核尺寸
- 卷积核数目
- 卷积核层数
- 滑动步幅
- 周边填充尺寸

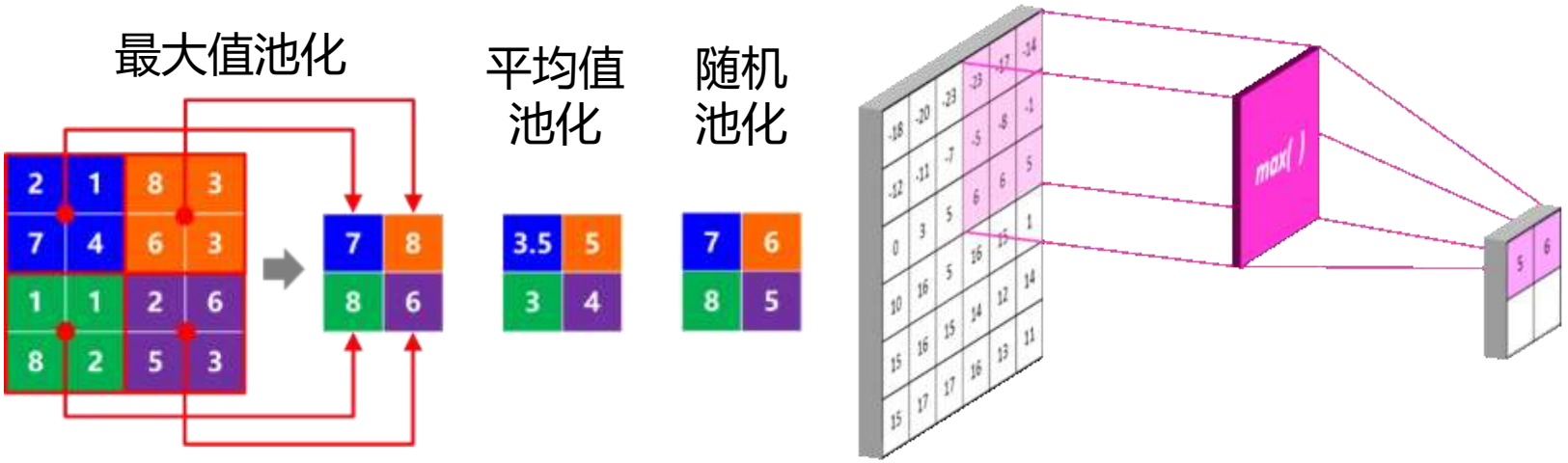


用零填充(Zero padding)输入特征图，以输出相同大小的特征图

监督学习：神经网络的结构

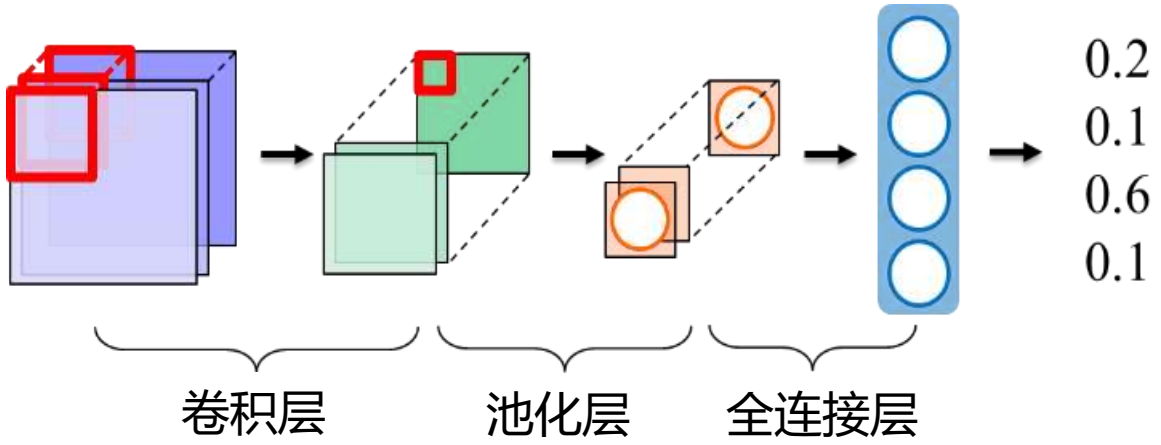
其他：池化层（Pooling）

- 邻居区域的统计数据，如：平均值、最大值、最小值
- 平移不变性
- 优点：（1）高层次特征提取；（2）降维；（3）避免过度拟合



监督学习：神经网络的结构总结

混合层网络



	#卷积层	内核大小	# 内核	# 内核层数	滑动步幅
LeNet	2	5	20,50	1,20	1
AlexNet	5	3,5,11	96~384	3~256	1,4
GoogLeNet	21	1,3,5,7	16~384	3~832	1,2
VGG-16	13	3	64~512	3~512	1
ResNet-152	151	1,3,7	64~2048	3~2048	1,2
MobileNet	27	1,3	32~1024	3~1024	1,2

监督学习：神经网络的训练

损失函数 $J(y^*, y)$ 的设计

- 均方误差：衡量模型输出和标签数据的距离

$$MSE(y^*, y) = \frac{1}{n}\sum_{i=1}^{n}(y_i^* - y_i)^2$$

- 交叉熵：测量真实分布y^*与模型预测分布y的相似性

$$H(y^*, y) = -\sum_{i=1}^{n} y_i^* \log(y_i)$$

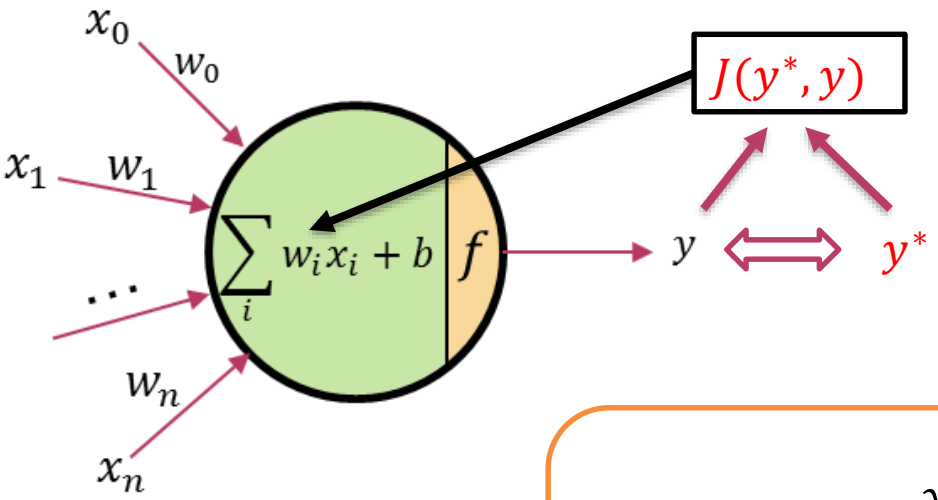
- Kullback–Leibler（KL）散度：测量两个分布之间的相似性

$$D_{KL}(y^*, y) = H(y^*) - H(y^*, y) = -\sum_{i=1}^{n} y_i^* \log\left(\frac{y_i}{y_i^*}\right)$$

监督学习：神经网络的训练

反向传播算法（单一神经元的梯度下降）

- 针对具有大量参数的神经网络
- 利用链式法则计算梯度

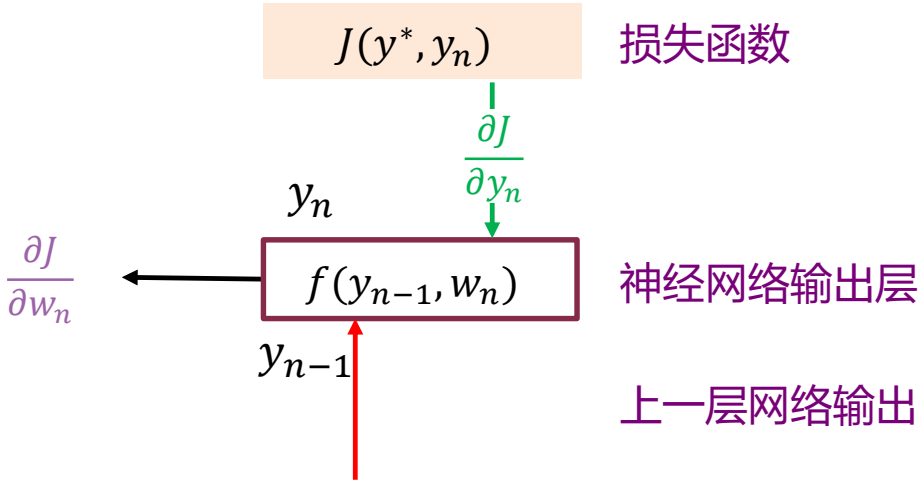


$$w \leftarrow w - \lambda \frac{\partial J(y^*, y)}{\partial w}$$

$$\frac{\partial J(y^*, y)}{\partial w} = \frac{\partial J}{\partial y} \frac{\partial y}{\partial w}$$

监督学习：神经网络的训练

反向传播算法（第n层：最后一层）



$$\frac{\partial J}{\partial y_n} = \frac{\partial J(y_n, y^*)}{\partial y_n}$$

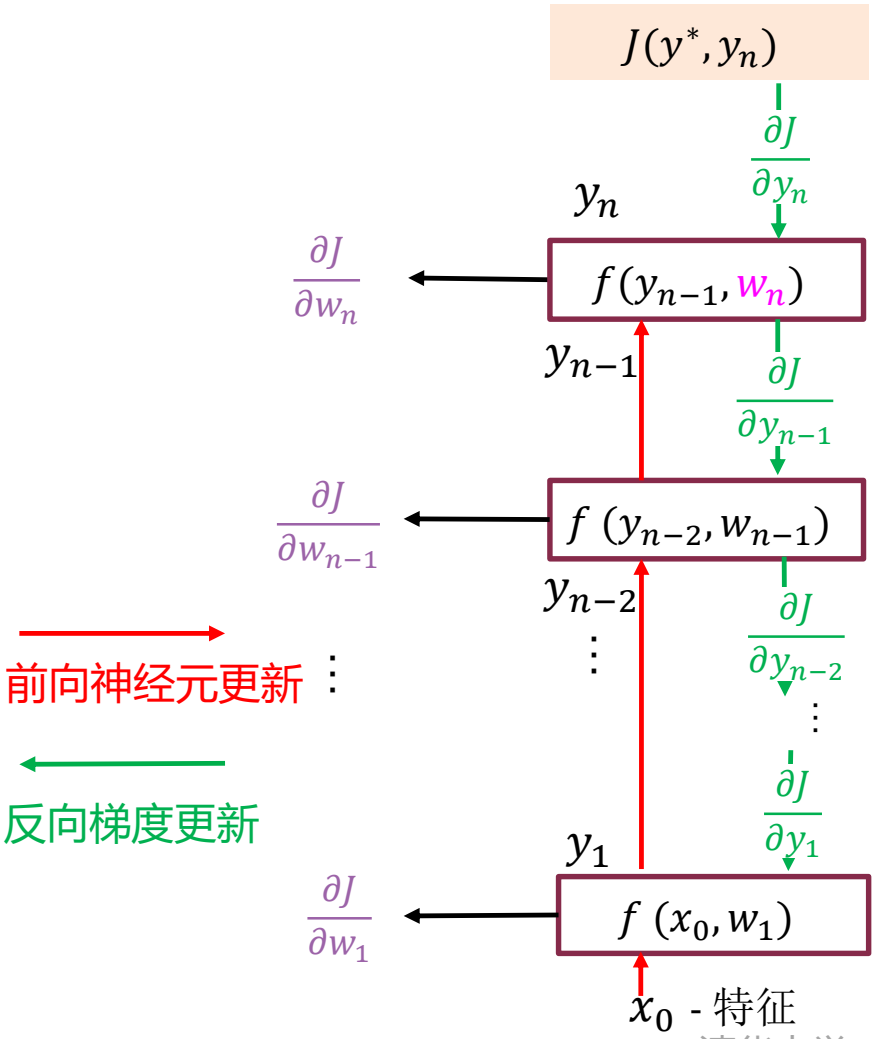
$$\frac{\partial J}{\partial w_n} = \frac{\partial J}{\partial y_n}\frac{\partial y_n}{\partial w_n}$$

$$y_n = f(y_{n-1}, w_n)$$

$$\frac{\partial J}{\partial w_n} = \frac{\partial J}{\partial y_n}\frac{\partial f(y_{n-1}, w_n)}{\partial w_n}$$

监督学习：神经网络的训练

反向传播算法

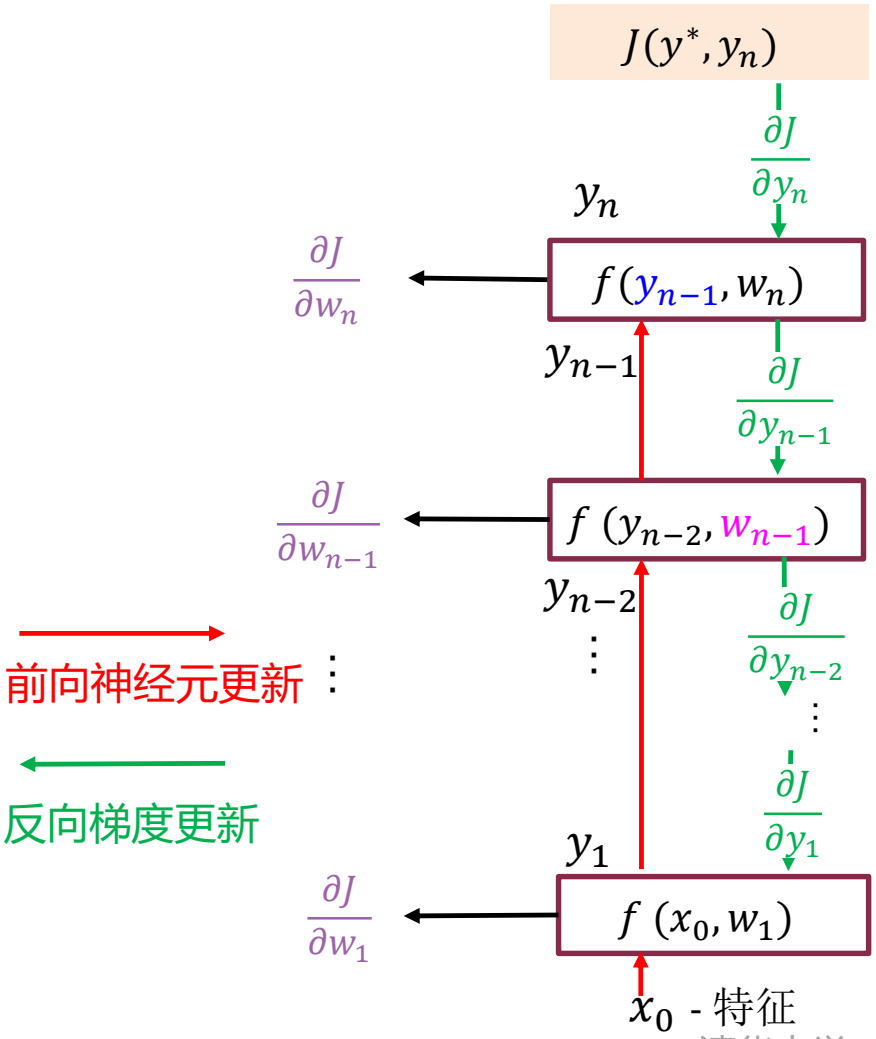


$$\frac{\partial J}{\partial y_n} = \frac{\partial J(y^*, y_n)}{\partial y_n}$$

$$\frac{\partial J}{\partial w_n} = \frac{\partial J}{\partial y_n}\frac{\partial y_n}{\partial w_n}$$

监督学习：神经网络的训练

反向传播算法



$$\frac{\partial J}{\partial y_n} = \frac{\partial J(y^*, y_n)}{\partial y_n}$$

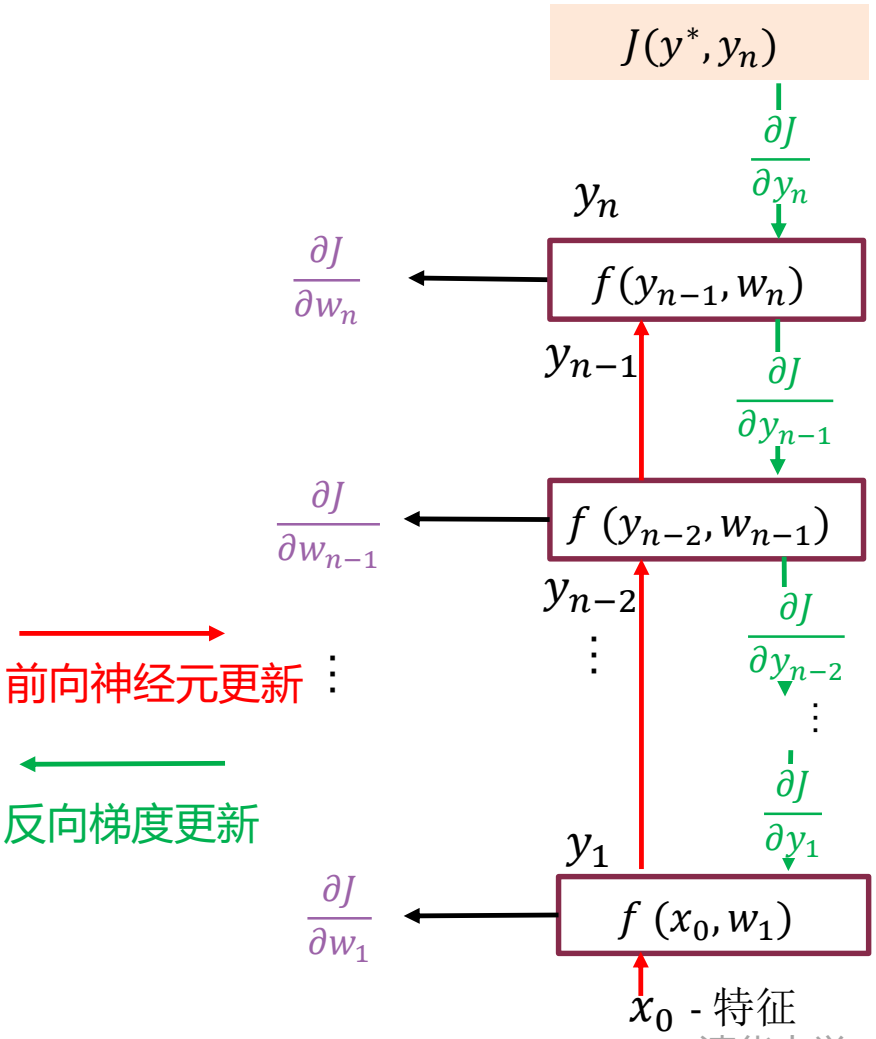
$$\frac{\partial J}{\partial w_n} = \frac{\partial J}{\partial y_n}\frac{\partial y_n}{\partial w_n}$$

$$\frac{\partial J}{\partial y_{n-1}} = \frac{\partial J}{\partial y_n}\frac{\partial y_n}{\partial y_{n-1}}$$

$$\frac{\partial J}{\partial w_{n-1}} = \frac{\partial J}{\partial y_{n-1}}\frac{\partial y_{n-1}}{\partial w_{n-1}}$$

监督学习：神经网络的训练

反向传播算法



$$\frac{\partial J}{\partial y_n} = \frac{\partial J(y^*, y_n)}{\partial y_n}$$

$$\frac{\partial J}{\partial w_n} = \frac{\partial J}{\partial y_n}\frac{\partial y_n}{\partial w_n}$$

$$\frac{\partial J}{\partial y_{n-1}} = \frac{\partial J}{\partial y_n}\frac{\partial y_n}{\partial y_{n-1}}$$

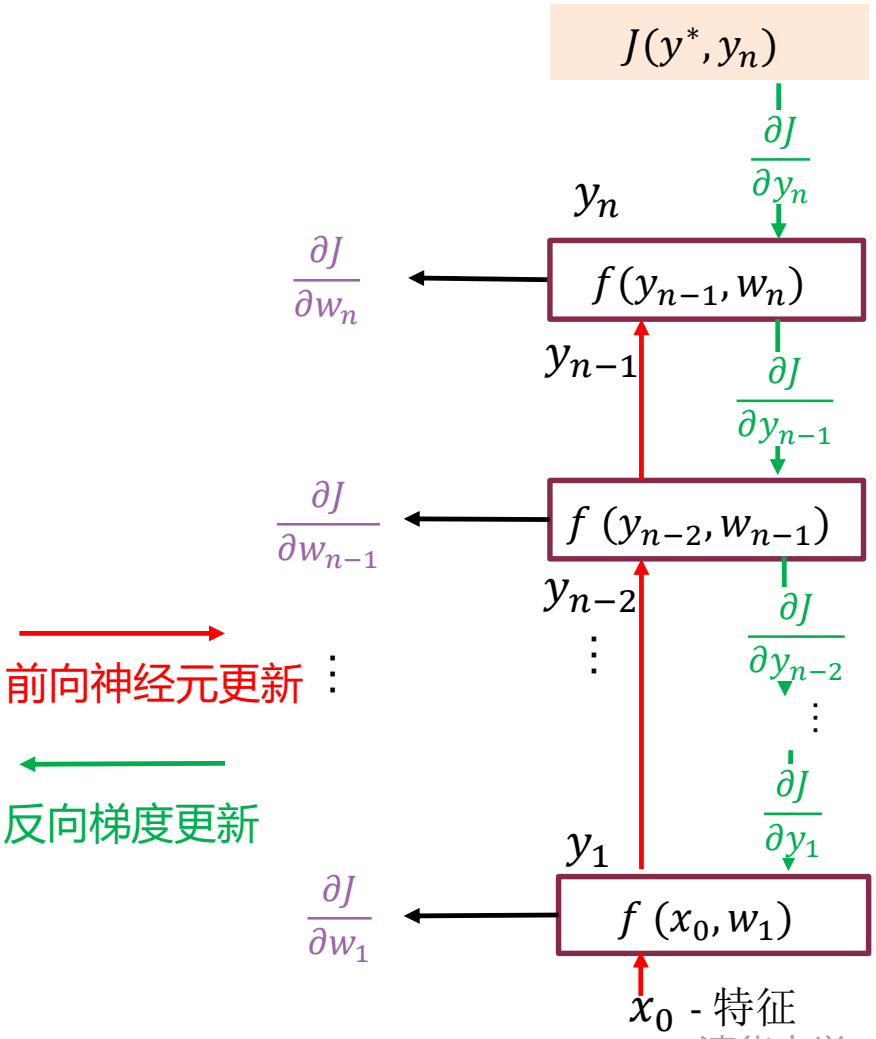
$$\frac{\partial J}{\partial w_{n-1}} = \frac{\partial J}{\partial y_{n-1}}\frac{\partial y_{n-1}}{\partial w_{n-1}}$$

$$\frac{\partial J}{\partial y_{n-2}} = \frac{\partial J}{\partial y_n}\frac{\partial y_n}{\partial y_{n-1}}\frac{\partial y_{n-1}}{\partial y_{n-2}}$$

$$\frac{\partial J}{\partial w_{n-2}} = \frac{\partial J}{\partial y_{n-2}}\frac{\partial y_{n-2}}{\partial w_{n-2}}$$

监督学习：神经网络的训练

反向传播算法



$$\frac{\partial J}{\partial y_n} = \frac{\partial J(y^*, y_n)}{\partial y_n}$$

$$\frac{\partial J}{\partial w_n} = \frac{\partial J}{\partial y_n}\frac{\partial y_n}{\partial w_n}$$

$$\frac{\partial J}{\partial y_{n-1}} = \frac{\partial J}{\partial y_n}\frac{\partial y_n}{\partial y_{n-1}}$$

$$\frac{\partial J}{\partial w_{n-1}} = \frac{\partial J}{\partial y_{n-1}}\frac{\partial f y_{n-1}}{\partial w_{n-1}}$$

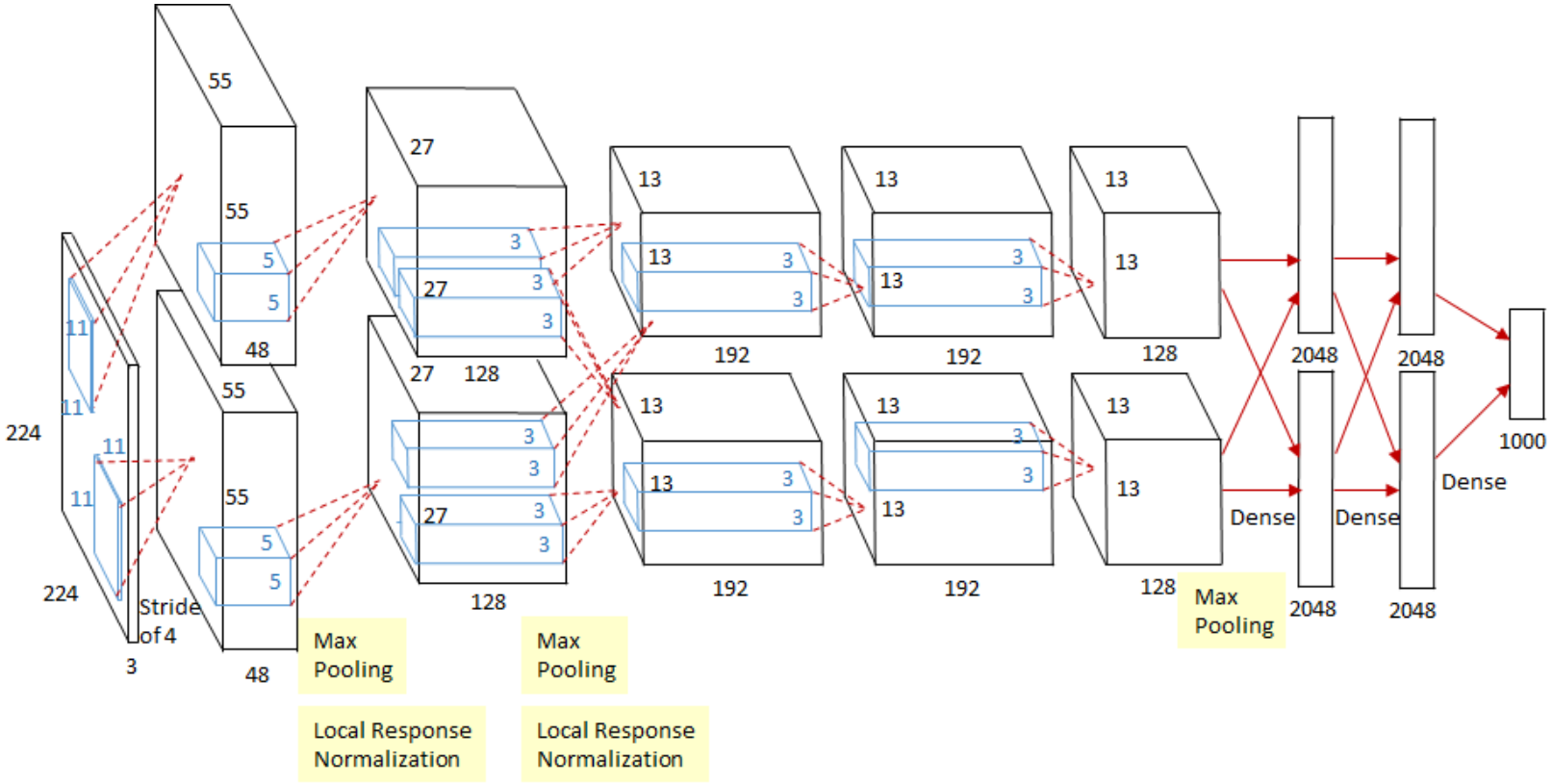
…… ……

$\frac{\partial J}{\partial w_1}$ → 损失函数

$\frac{\partial J}{\partial w_1}$ → 第一层网络权值

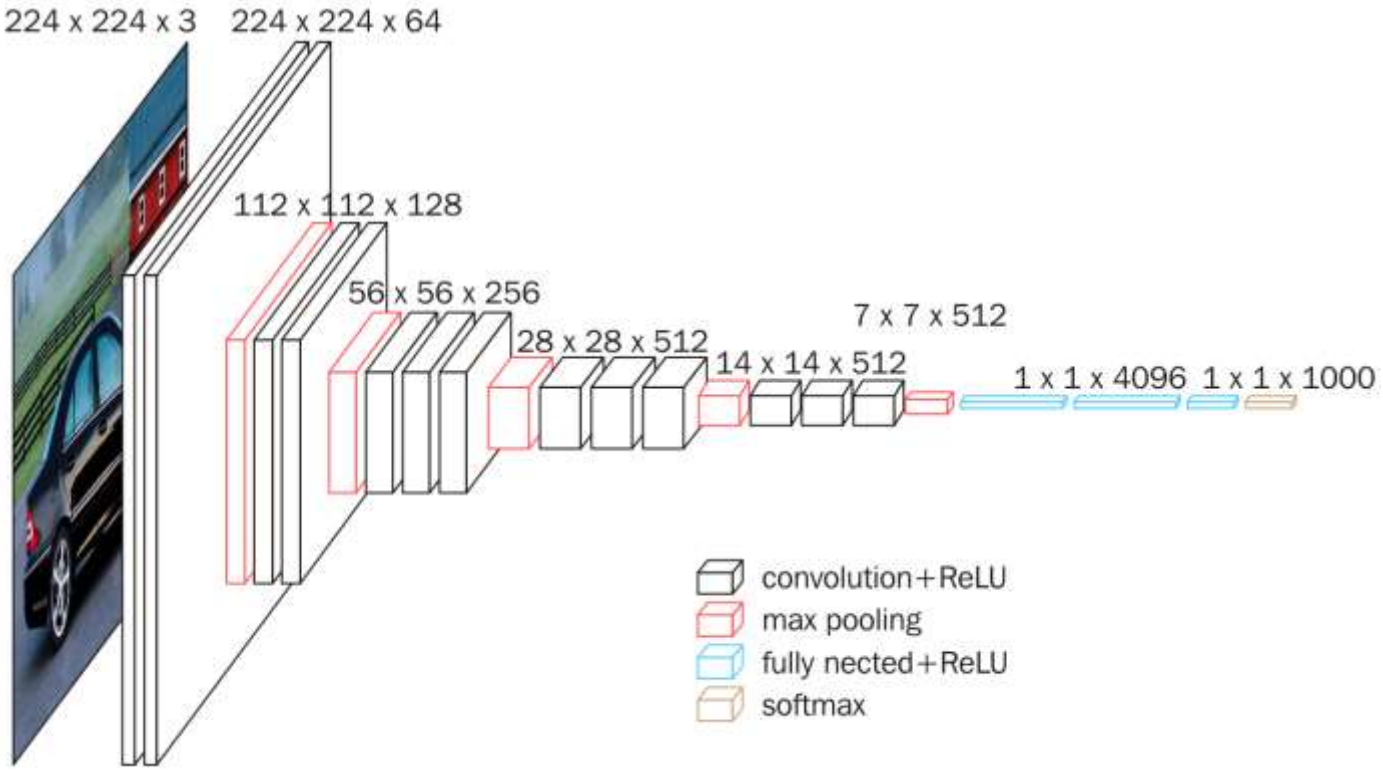
监督学习：典型神经网络示例

- **AlexNet (Hinton, 2012年)**
  - 首次使用 ReLU激活函数，8层
  - 用于GPU，快速计算的两个并行网络
  - ImageNet： top 5 错误率为15.4%



监督学习：典型神经网络示例

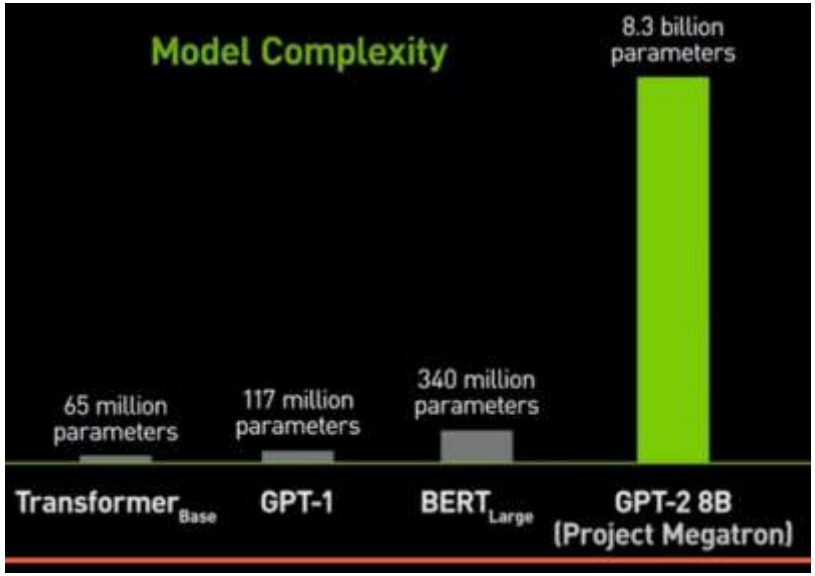
- **VGGNet（牛津视觉组，2014年）**
  - 小型内核，深层网络
  - 8 层 (AlexNet) → 16-19 层 (VGG16Net)
  - 仅3x3 CONV步幅1，pad 1和2x2 MAX POOL步幅2
  - 从AlexNet 的 top 5错误率为15.4% → top 5错误率为7.3%

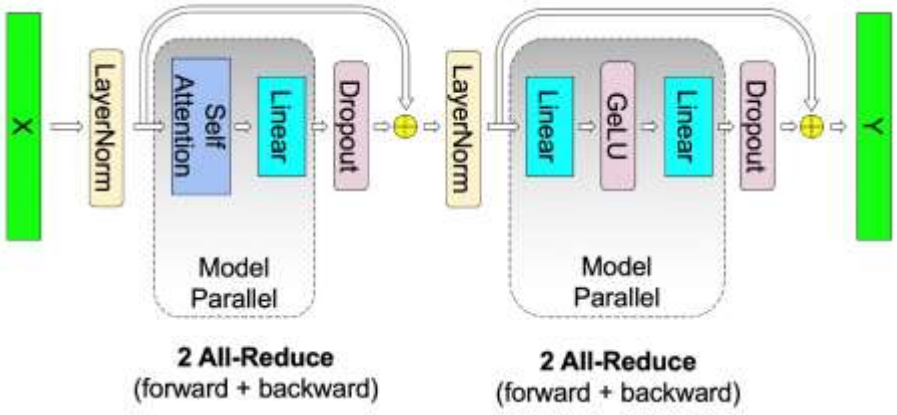


人类神经系统 VS 机器神经网络

英伟达 Megatron-LM 项目 (2019)

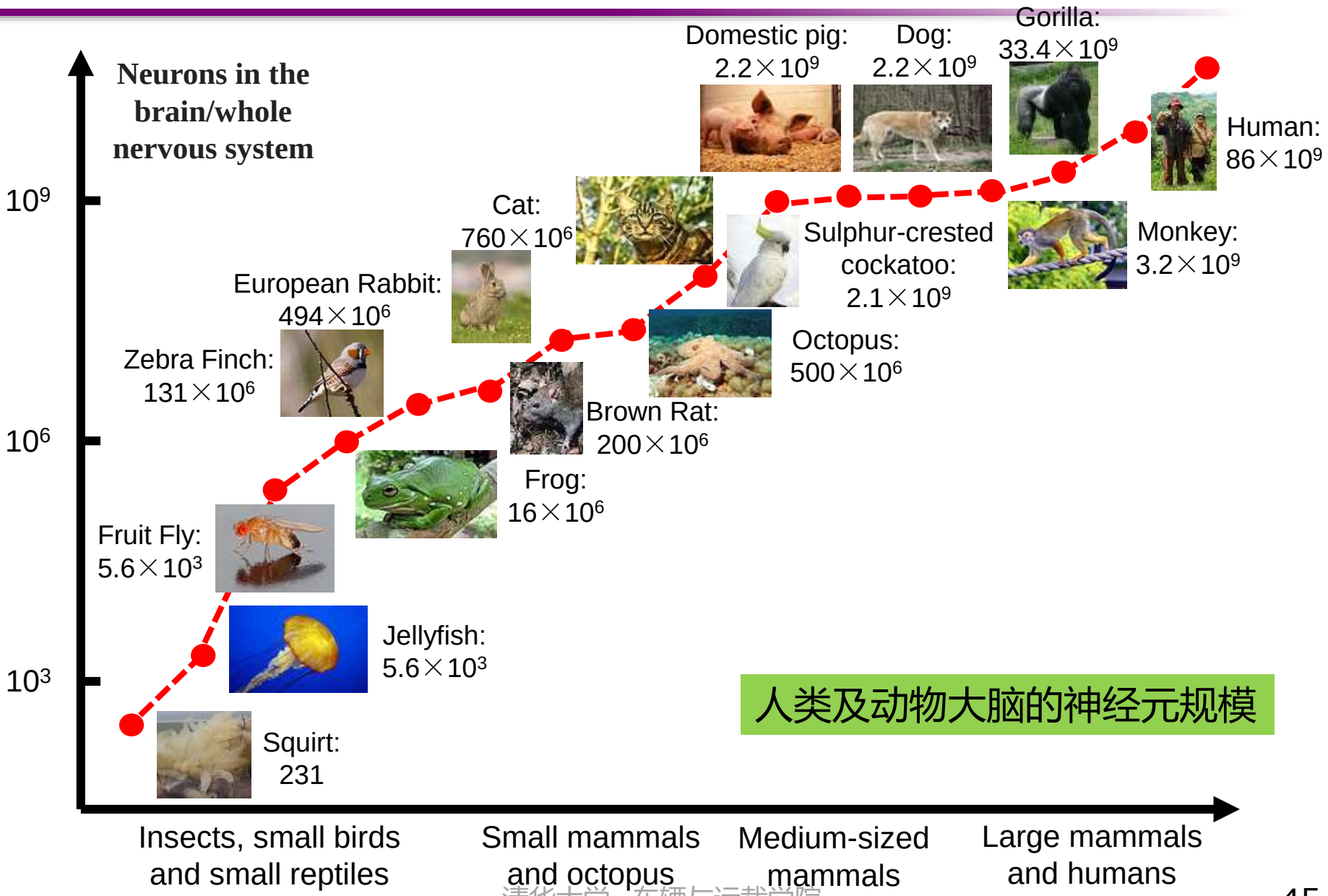
- BERT语言模型
- 训练2.8天，83亿参数
- 训练平台：32台DGX- 2h服务器（512 GPU）
- 前向推断速度2.2ms





Model Size	Number of layers	Hidden Size	Wikitext (Perplexity ↓)	Lambada (Accuracy ↑)
345 M	24	1024	24.21	55.04
775 M	36	1280	20.44	58.86
2.5 B	54	1920	17.83	63.30
8.3 B	72	3072	17.41	63.11

人类神经系统 VS 机器神经网络



人类及动物大脑的神经元规模

单选题 1分

设置

若只以图灵测试为标准，你认为以深度学习为代表的AI可以完美近似人类吗?

A 可以：人是机器，AI也是机器

B 不可以：人类不是上帝

C 不可能：人类设计的AI不能完美接近人类本身

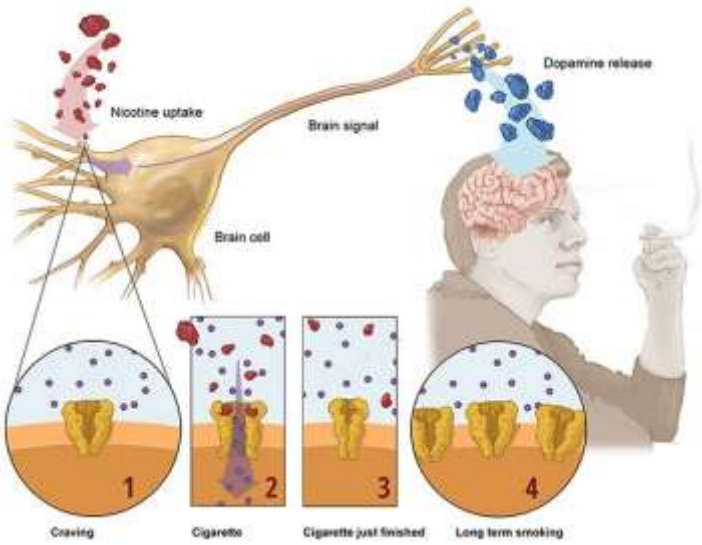
D 说不清楚：问题实在过于复杂

提交

集中式决策：强化学习型

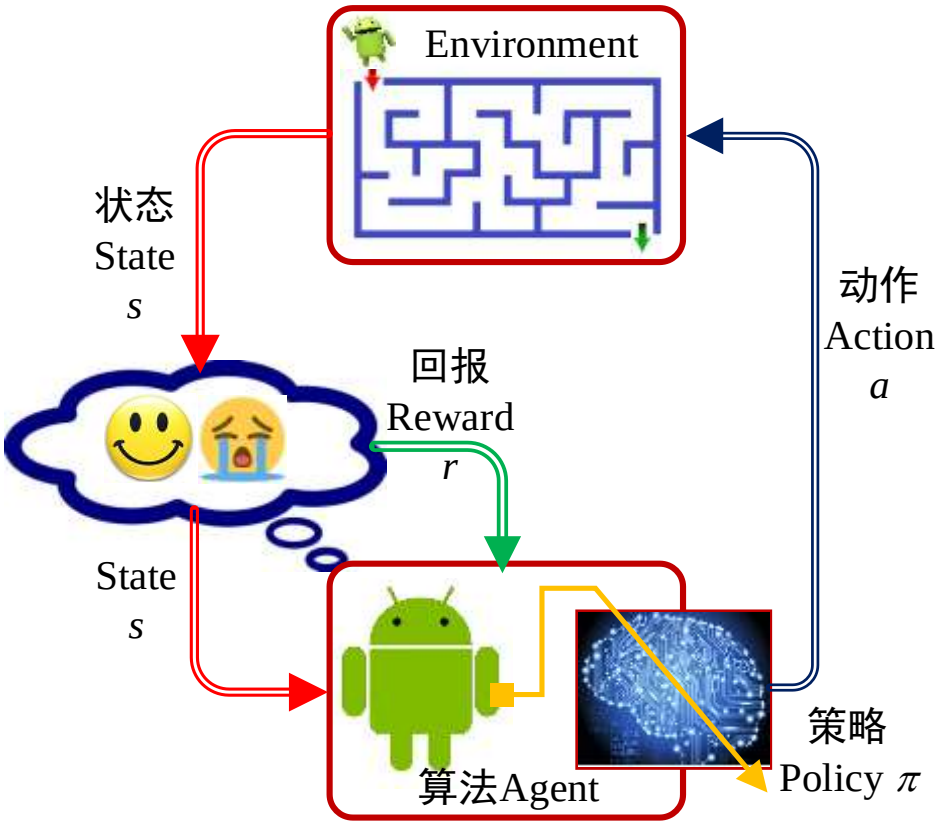
强化学习简介

- **强化学习是一种生物启发算法**
  - 模仿动物的学习过程
  - 重复受奖赏回报行为，减少受惩罚回报行为



强化学习算法本质

- 求解**最优策略 π**，**最大化（最小化）长期价值$v^{\pi}(s)$或$q^{\pi}(s,a)$**
- **服从环境Environment的动态特性**



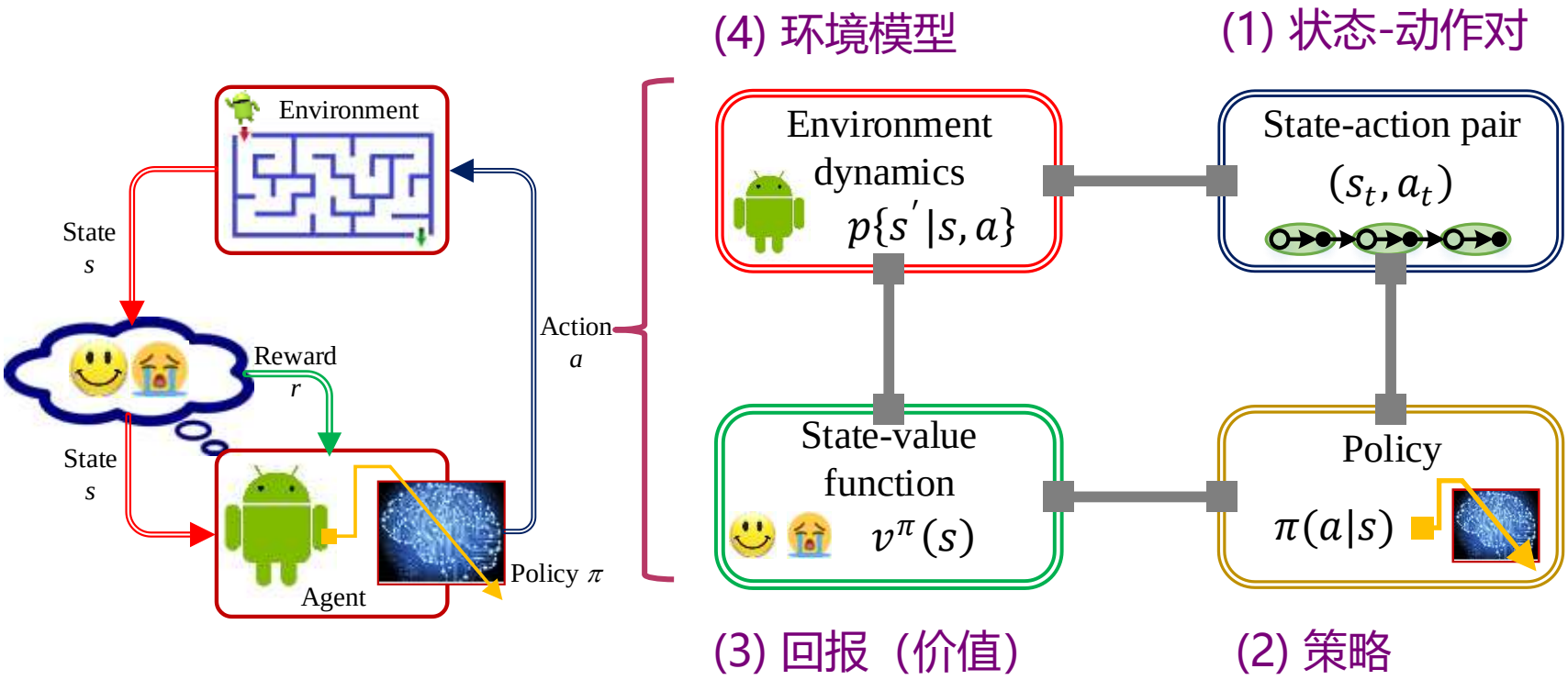
$$\max_{\pi} v^{\pi}(s), \forall s \in \mathcal{S}$$

Subj. to

$$p\{s'|s,a\}=\mathcal{P}_{ss'}^{a}$$

$\pi^*(a|s)$?

强化学习的四元素构架


(4) 环境模型
(1) 状态-动作对
Environment
State
s
Reward
r
State
s
Action
a
Agent
Policy π
Environment
dynamics
$p\{s'|s,a\}$
State-action pair
(s_t, a_t)
State-value
function
$v^{\pi}(s)$
Policy
$\pi(a|s)$
(3) 回报（价值）
(2) 策略

多选题 1分

设置

如下《射雕英雄传》的武林高手，那一些属于强化学习机制的武功学习？

A 郭靖：师学多门，勤学肯练

B 黄蓉：天资聪颖，一点就通

C 黄药师：造诣非凡，无师自通

D 洪七公：丐帮嫡传，变化万千

E 周伯通：一心二用，左右互搏

提交

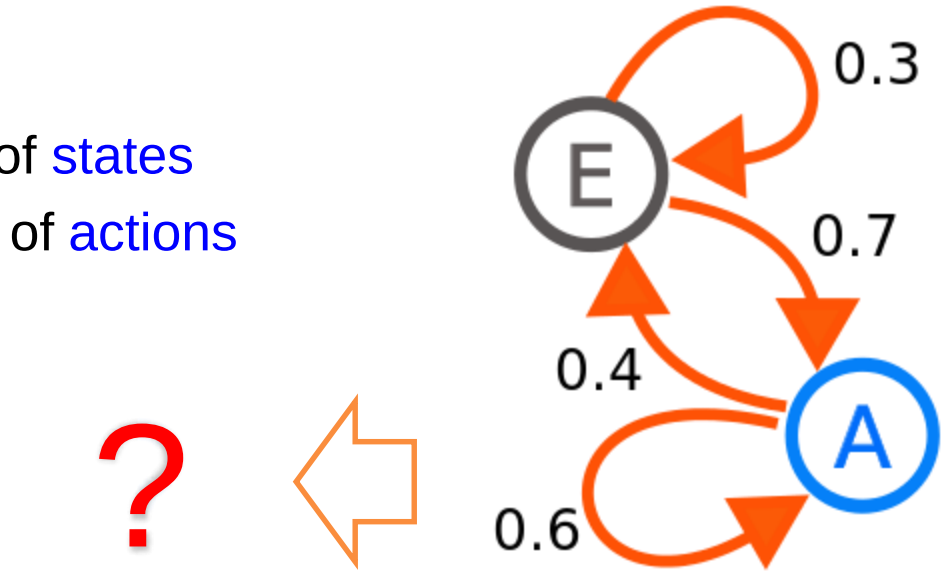
环境模型

Markov chain（马尔科夫链）：

- 转移概率函数/Transition probability in environment model

$$\mathcal{P}_{ss'}^{a} \triangleq p\{s'|s,a\}$$
$$s \in \mathcal{S}, a \in \mathcal{A}$$

- t：current time/时间
- s：state/状态
- a：action/动作
- $\mathcal{S}$：a finite set of states
- $\mathcal{A}$：a finite set of actions



Reward

瞬时奖励信号

- A function of triple (s, a, s')

$$r_t \overset{\text{def}}{=} r_{ss'}^{a} = r(s_t, a_t, s_{t+1})$$

- 定义了当前时刻策略的优劣性

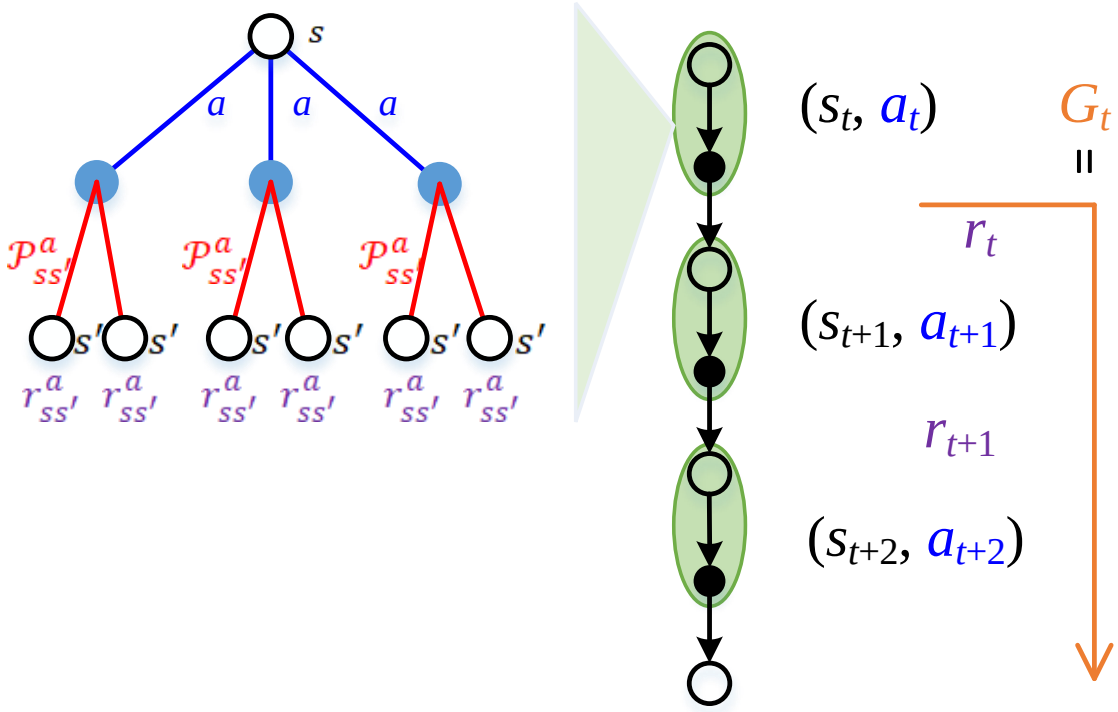
累积折扣回报

$$G_t = \sum_{i=0}^{+\infty} \gamma^i r_{t+i}$$

- γ : 折扣因子, $0 < \gamma < 1$
- 接近 0 → "myopic" evaluation
- 接近 1 → "far-sighted" evaluation

Reward

- 状态、动作和奖励的关系



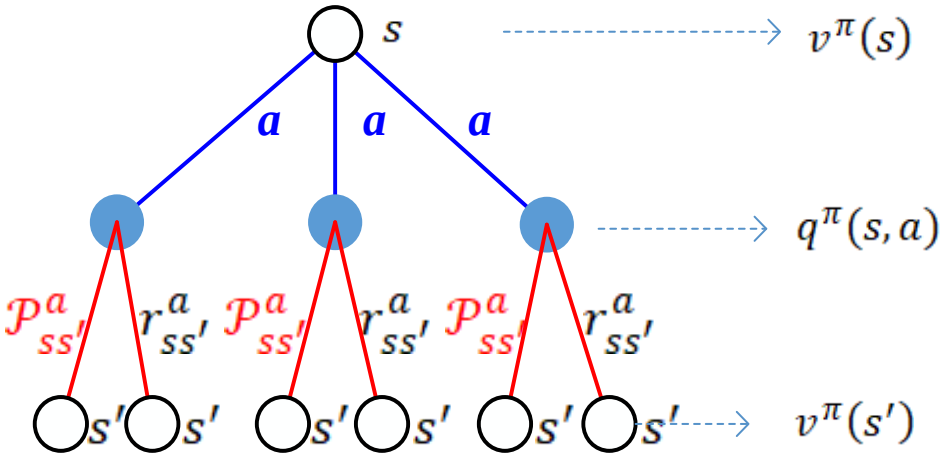
- 累积折扣回报 G_t 的计算公式

$$G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots = \sum_{i=0}^{+\infty} \gamma^i r_{t+i}$$

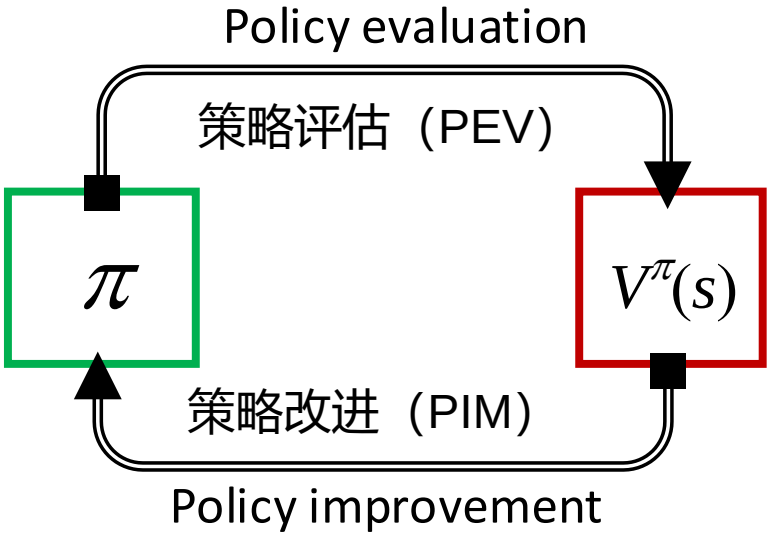
值函数 \ Value function

- 状态值函数 $v^{\pi}(s) \overset{\text{def}}{=} \mathbb{E}_{\pi}\{G_t|s\} = \mathbb{E}_{\pi}\left\{\sum_{i=0}^{+\infty}\gamma^i r_{t+i}\,|s_t = s\right\}$

- 动作值函数 $q^{\pi}(s,a) \overset{\text{def}}{=} \mathbb{E}_{\pi}\{G_t|s,a\} = \mathbb{E}_{\pi}\left\{\sum_{i=0}^{+\infty}\gamma^i r_{t+i}\,|s_t = s, a_t = a\right\}$



强化学习的基本迭代方法



强化学习算法可以采用策略评估和策略改进的交替迭代，直到收敛至最优策略。

策略评估：计算当前策略对应的价值
策略改进：根据当前价值改进策略

策略改进方式

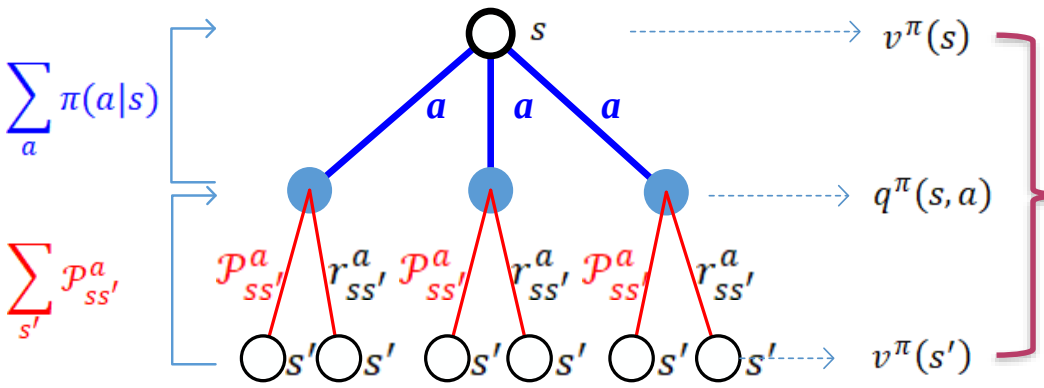
- 基于动作值函数， $q^*(s,a)$

$$a^* = \arg\max_a q^*(s,a)$$

- 基于状态值函数， $v^*(s')$

$$a^* = \arg\max_a \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a \left(r_{ss'}^a + \gamma v^*(s') \right)$$

模型已知



自洽条件

$$q^\pi(s,a) = \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a \left(r_{ss'}^a + \gamma v^\pi(s') \right)$$

强化学习算法示例—蒙特卡洛算法

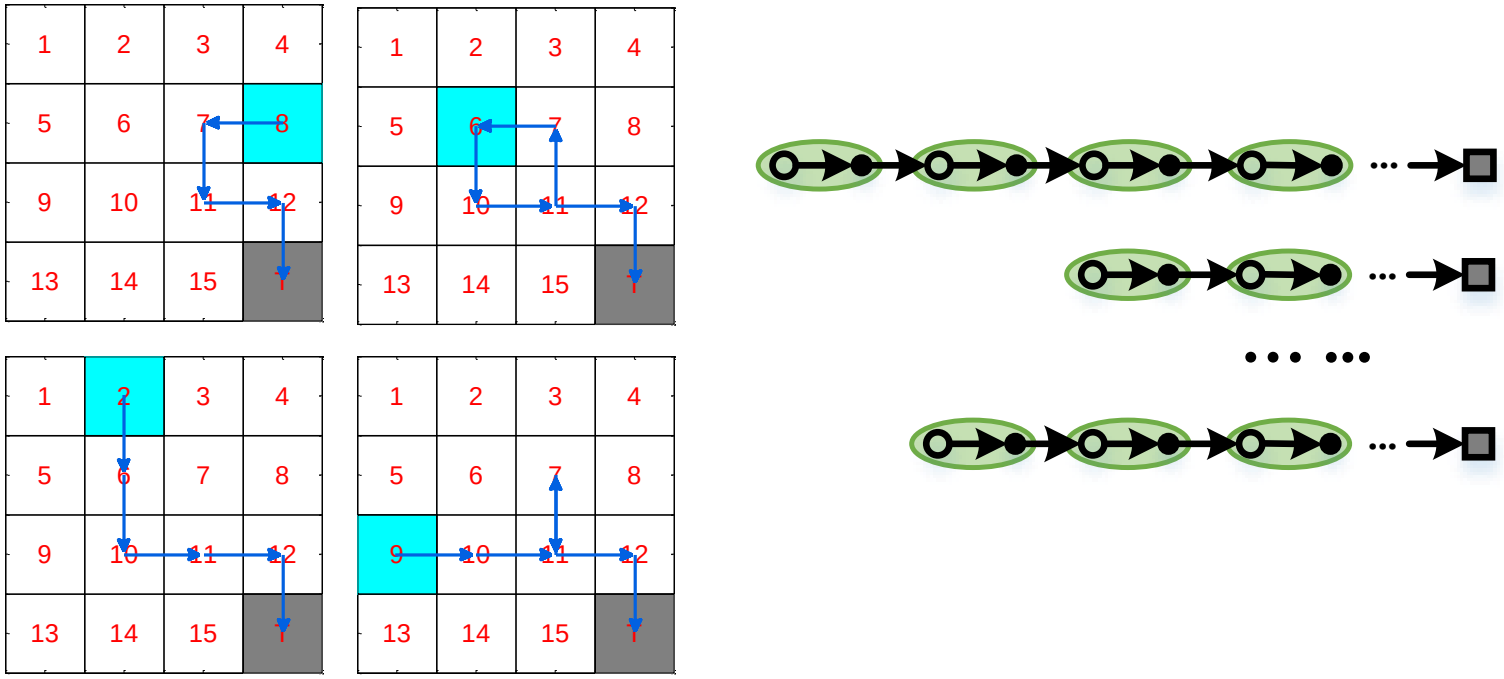
蒙特卡洛（Monte Carlo，MC） 算法

- 适用于周期性任务（每次任务时长有限）
- 根据周期任务数据进行学习
- 无需环境模型
- 价值函数简单: 累积回报的平均值

强化学习算法示例—蒙特卡洛方法

MC 策略评估

- 利用当前策略与环境进行交互，产生N组的样本数据



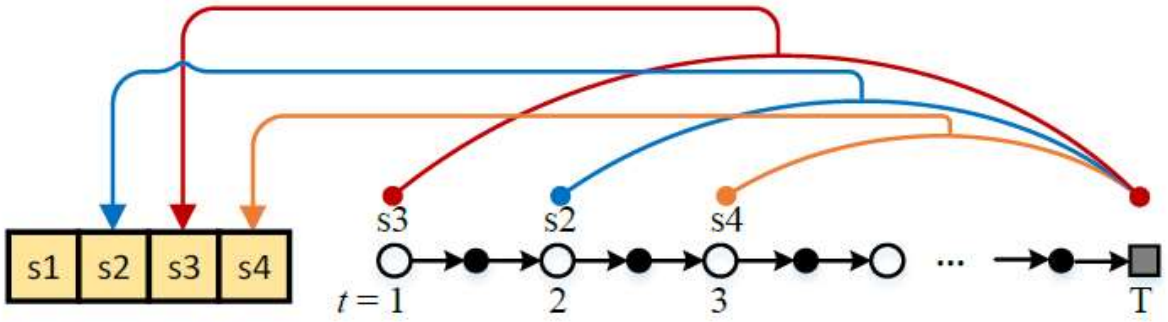
强化学习算法示例—蒙特卡洛方法

MC 策略评估

- 计算状态值函数 V^{π}
- 从同一初始状态出发，计算所有折扣回报的平均值

$$V^{\pi}(s) = \text{Avg}\{G_{t:T} | s_t = s\}$$

$$G_t = \sum_{i=t}^{T-1} \gamma^{i-t} r_i$$



状态-价值表

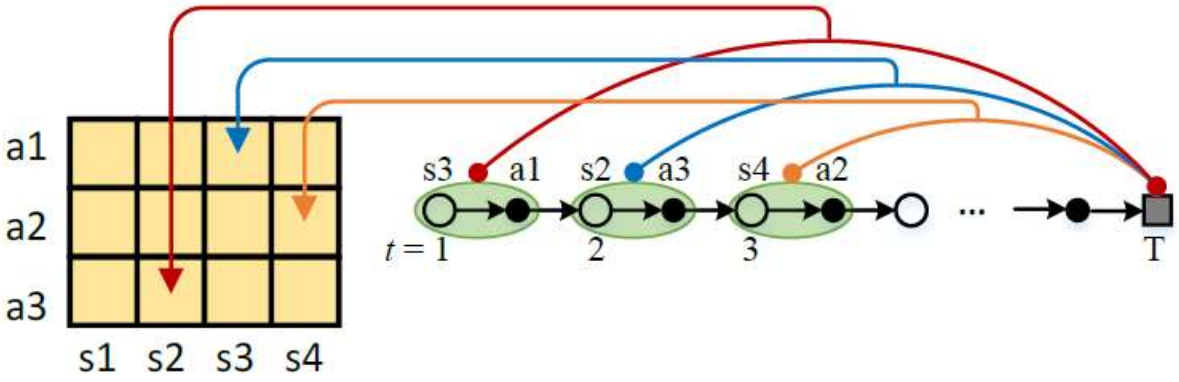
强化学习算法示例—蒙特卡洛算法

MC 策略评估

- 计算动作值函数 Q^{π}
- 从同一初始状态-动作对出发，计算所有折扣回报的平均值

$$Q^{\pi}(s,a) = \text{Avg}\{G_{t:T} | s_t = s, a_t = a\}$$

$$G_t = \sum_{i=t}^{T-1} \gamma^{i-t} r_i$$



动作-价值表

强化学习算法示例—蒙特卡洛方法

MC 策略改进

- 目标: 寻找更优策略 $\bar{\pi}$

$$\pi \leq \bar{\pi}$$

- 模型未知：基于$Q^{\pi}(s,a)$, 贪心（Greedy）估算 $\boldsymbol{\pi^{g}(a|s)}$

$$a^{*} = \arg\max_{a} Q^{\pi}(s,a)$$

$$\pi^{g}(a|s) = \begin{cases} 1, & if\ a = a^{*} \\ 0, & if\ a \neq a^{*} \end{cases}$$

- 模型已知：基于$V^{\pi}(s)$, 贪心（Greedy）估算 $\boldsymbol{\pi^{g}(a|s)}$

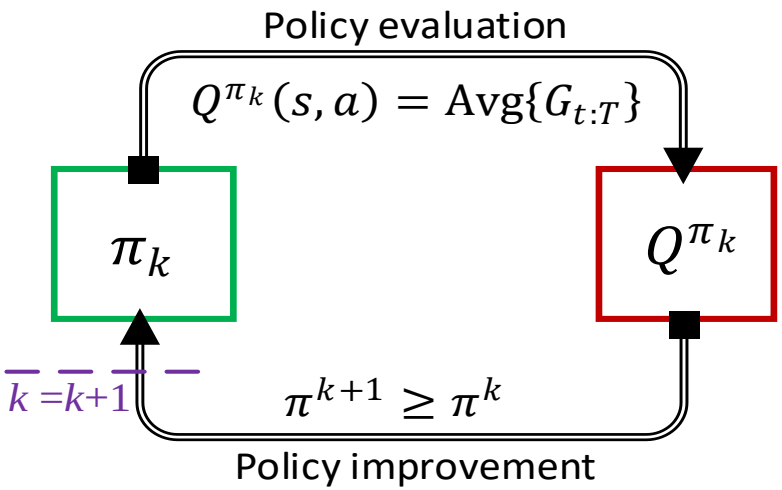
$$a^{*} = \arg\max_{a} \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^{a} \left(r_{ss'}^{a} + \gamma V^{\pi}(s') \right)$$

$$\pi^{g}(a|s) = \begin{cases} 1, & if\ a = a^{*} \\ 0, & if\ a \neq a^{*} \end{cases}$$

强化学习算法示例—蒙特卡洛方法

MC 算法迭代

- 交替执行 PEV 和 PIM 两步，直到算法收敛

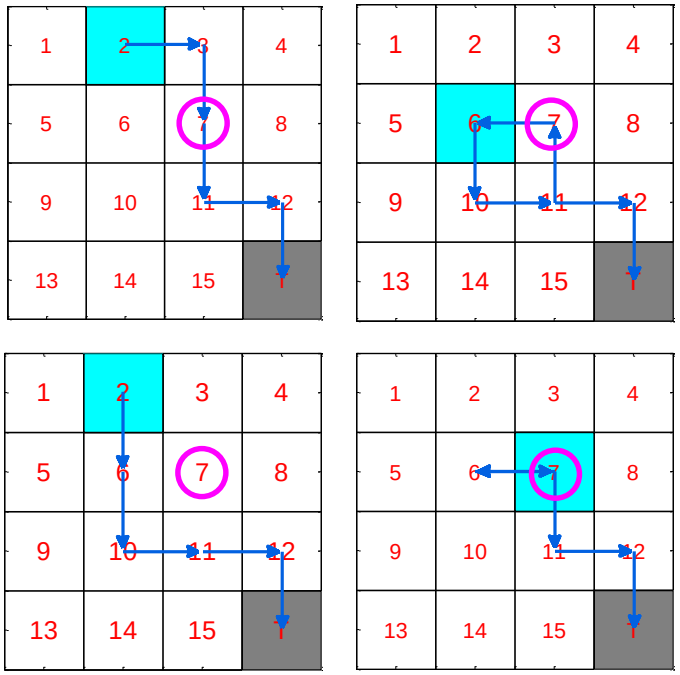


$$\pi^0 \xrightarrow{\text{PEV}} Q^{\pi_0} \xrightarrow{\text{PIM}} \pi^1 \xrightarrow{\text{PEV}} Q^{\pi_1} \xrightarrow{\text{PIM}} \cdots \xrightarrow{\text{PIM}} \pi^* \xrightarrow{\text{PEV}} q^*$$

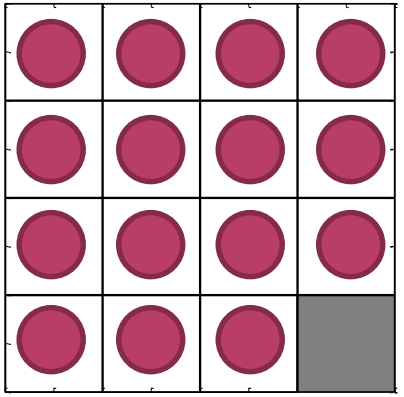
- 每步迭代
  - 值函数更新，更加接近最优值函数
  - 策略更新，更加接近最优策略

思考题

- **示例: 请计算当$s = 7$时的状态值函数 $V^{\pi}(s = 7)$**
  - 若给定策略$\pi(s, a)$ →通过探索环境获取 4 份数据
  - 天蓝色代表起点；灰色代表终点



?



每走一步的奖励

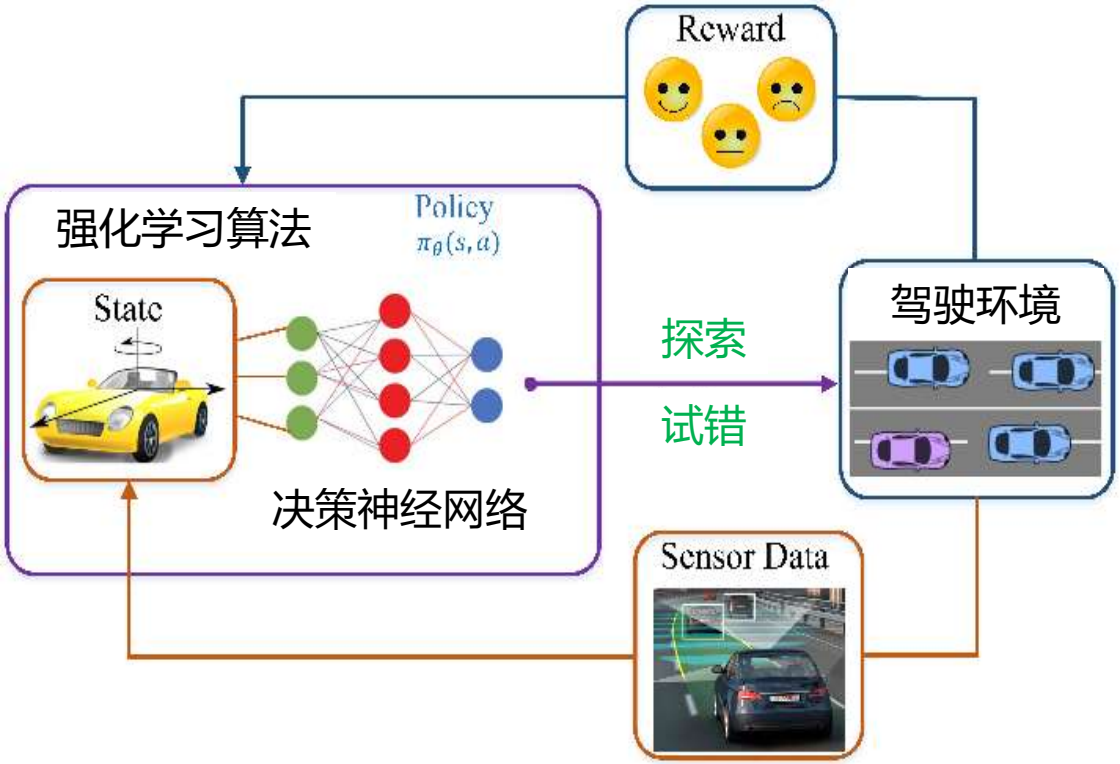
$$r(s, a, s') = \begin{cases} -1 & \text{if } s' \neq \text{T} \\ +9\text{,} & \text{if } s' = \text{T} \end{cases}$$

$\gamma = 0.9$ 折扣因子

强化学习型决策框架

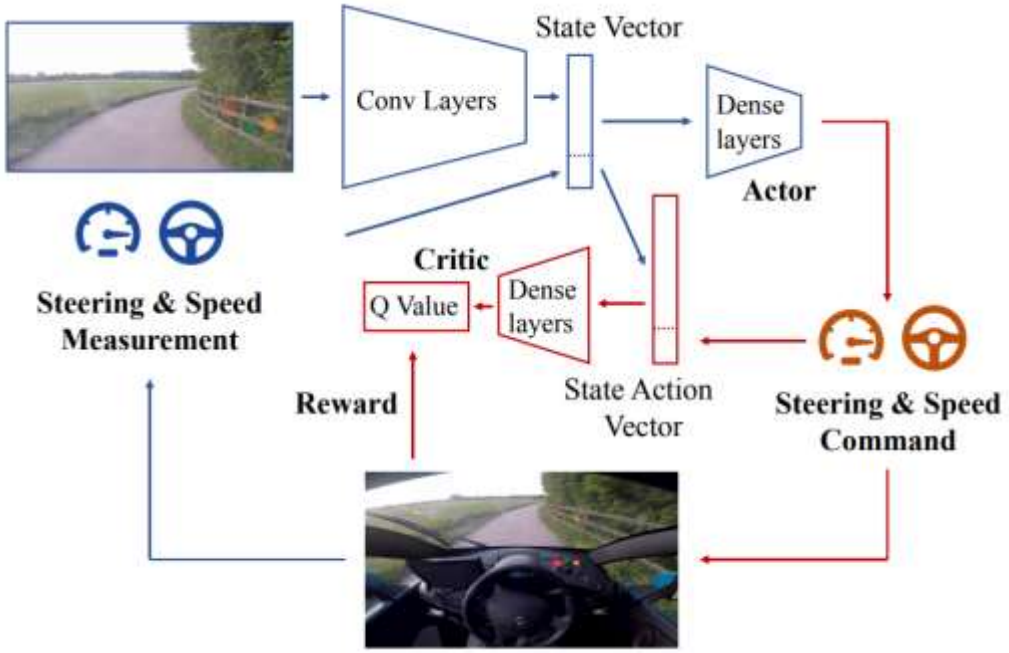
强化学习型集中式自主决策

- 人工智能领域的研究热点
- 不依赖大量的带标签训练数据
- 探索试错式自我进化



强化学习型决策示例

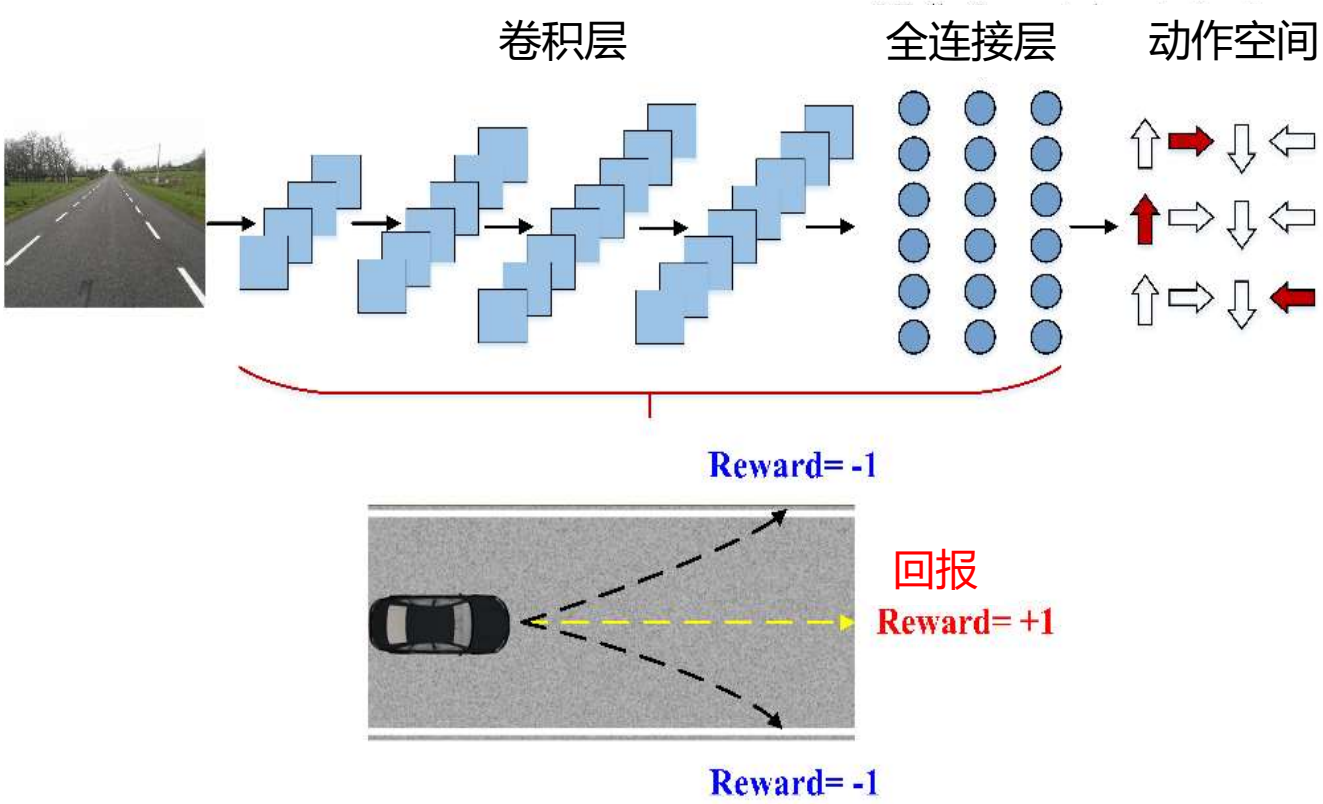
- **Wayve的决策规划**
  - 任务：车道行驶
  - 算法：DDPG（深度确定性策略梯度）
  - 模型：CNN（卷积神经网络）
- **训练及测试**
  - 收敛时长：约1天



强化学习型决策示例

Wayve的决策规划模型

- DDPG 神经网络训练

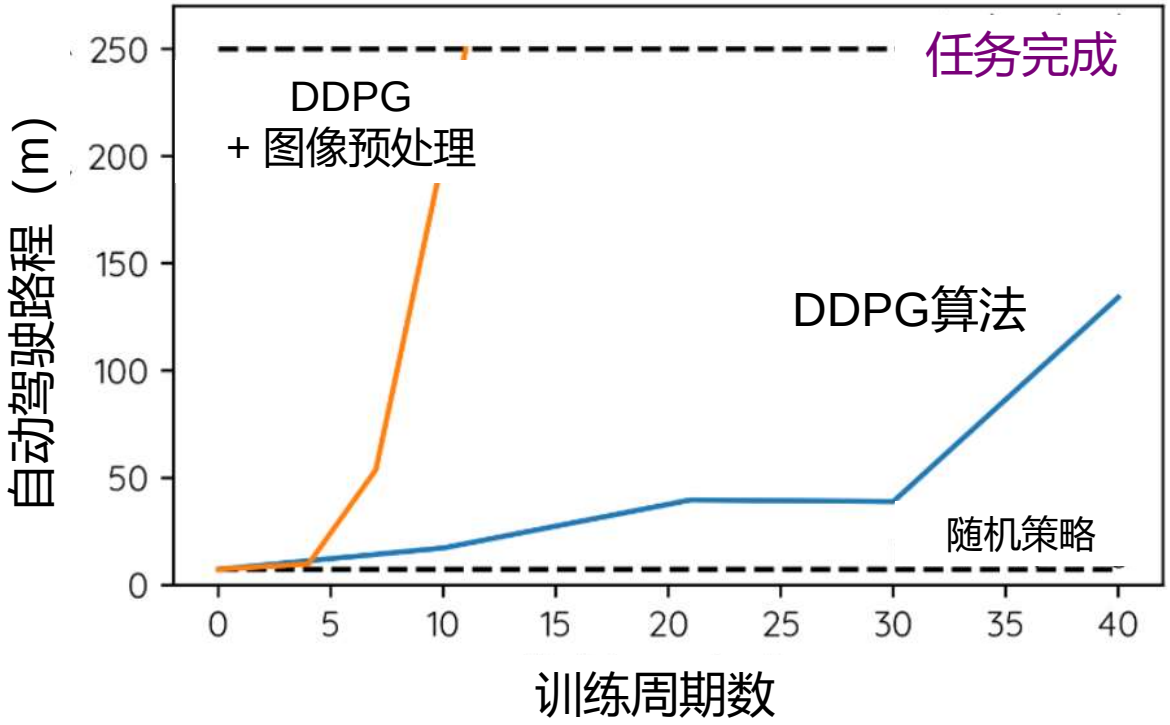


强化学习型决策示例

强化学习型决策示例

训练结果

- 随机策略：不能完成车道行驶任务
- DDPG：逐步提升控制策略
- DDPG+图像预处理：提升训练速度



本课小结

监督学习型决策

- 需要大量带标签的驾驶员数据
- 利用深度神经网络进行策略训练
- 所需的标签数据量太大，是制约训练效果提升的难题

强化学习型决策

- 不需要带标签的驾驶员数据
- 利用探索试错原理寻找策略
- 回报函数的设计是关键，安全性仍是制约上路的瓶颈问题

决策规划技术小结

决策规划技术小结

决策规划技术的总结

分解式决策 VS **集中式决策**

- 易用，便于工程师实践
- 任务分解，便于团队协作
- 节省存储和计算资源
- 算法可解释性好

- 类人机制，框架结构简洁
- 具有较好的可扩展性
- 环境感知信息不损失
- 几乎不需要人工标记数据

现有技术的不足

- （1）几乎不能利用车辆模型和驾驶员经验，移植能力差
- （2）驾驶数据样本不均匀，难以训练（长尾效应严重）
- （3）决策能力改进依赖于探索试错，行车安全性不能保障

谢谢大家！