PPO x Family

开讲啦

主办

承办

协办

支持

课程目标

- 介绍深度强化学习和PPO算法
- PPO算法的原理分析
- PPO算法的实践应用

课程大纲

第一课

开启决策AI探索之旅

第二课

解构复杂动作空间

第三课

表征多模态观察空间

第四课

解密稀疏奖励空间

第五课

探索时序建模

第六课

统筹多智能体

第七课

挖掘黑科技

第八课

突破智能体
终极界限

您将获得

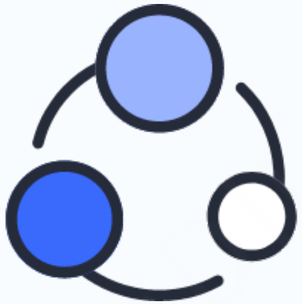
从零到一，掌握设计决策AI的万能算法 PPO

盘清算法理论、理顺代码逻辑，玩转实践应用

CCF官方认证

考核机制

课后小测验



实践巩固题

期末大作业

Interaction

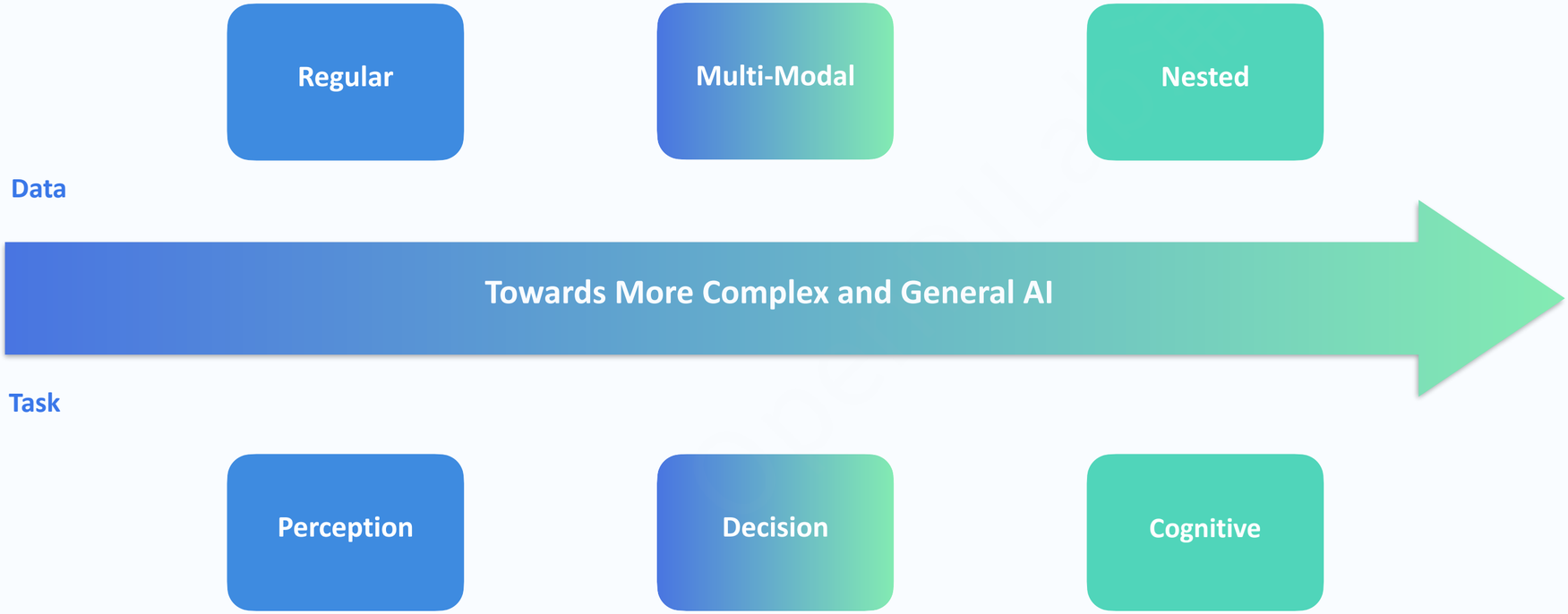
互动交流

小助手微信号：OpenDILab

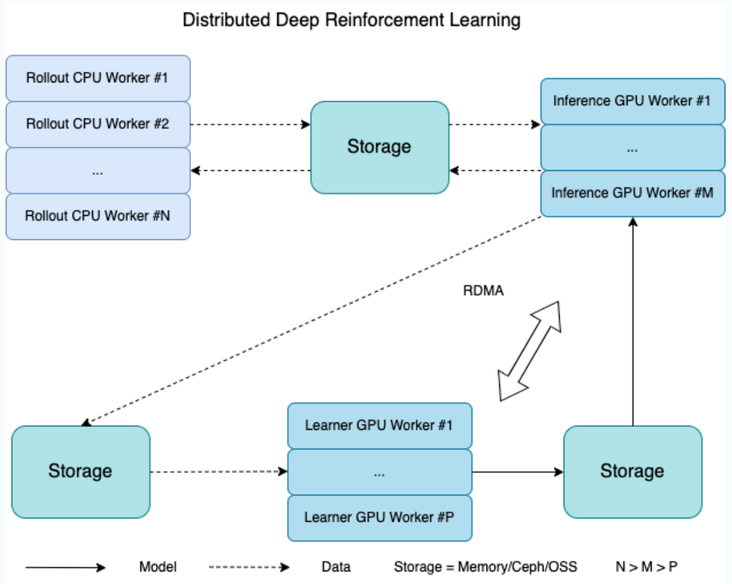
GitHub：https://github.com/opendilab/PPOxFamily

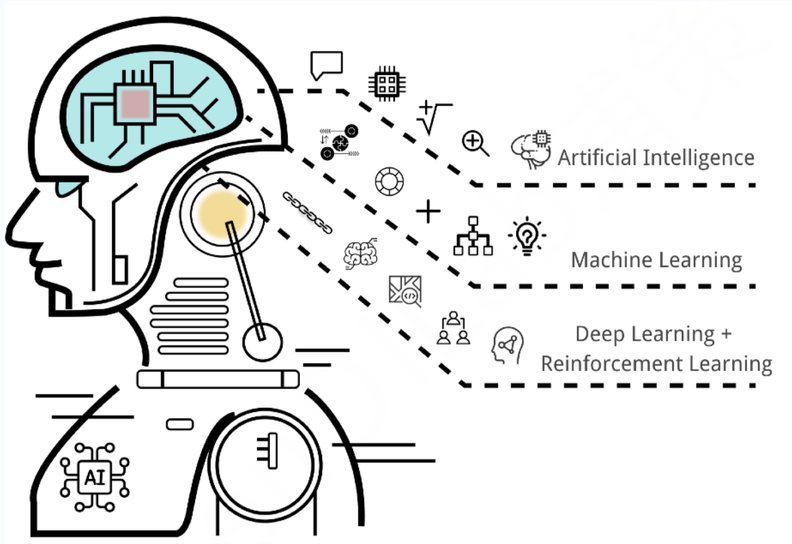
Slack：https://join.slack.com/t/opendilab/shared_invite/zt-v9tmv4fp-nUBAQEH1_Kuyu_q4plBssQ

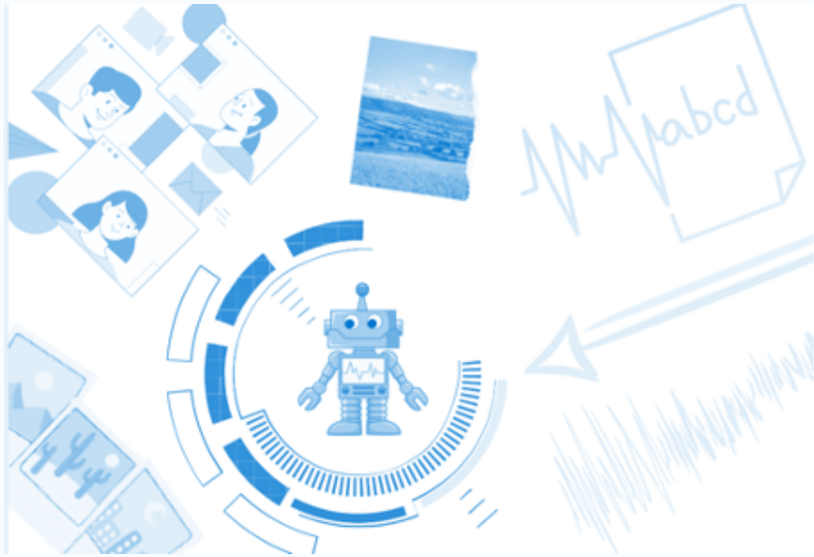
从感知AI到决策AI


Regular
Multi-Modal
Nested
Data
Towards More Complex and General AI
Task
Perception
Decision
Cognitive

人工智能







算力

算法

感知型AI　　决策型AI

数据

感知

语言/语音/图像

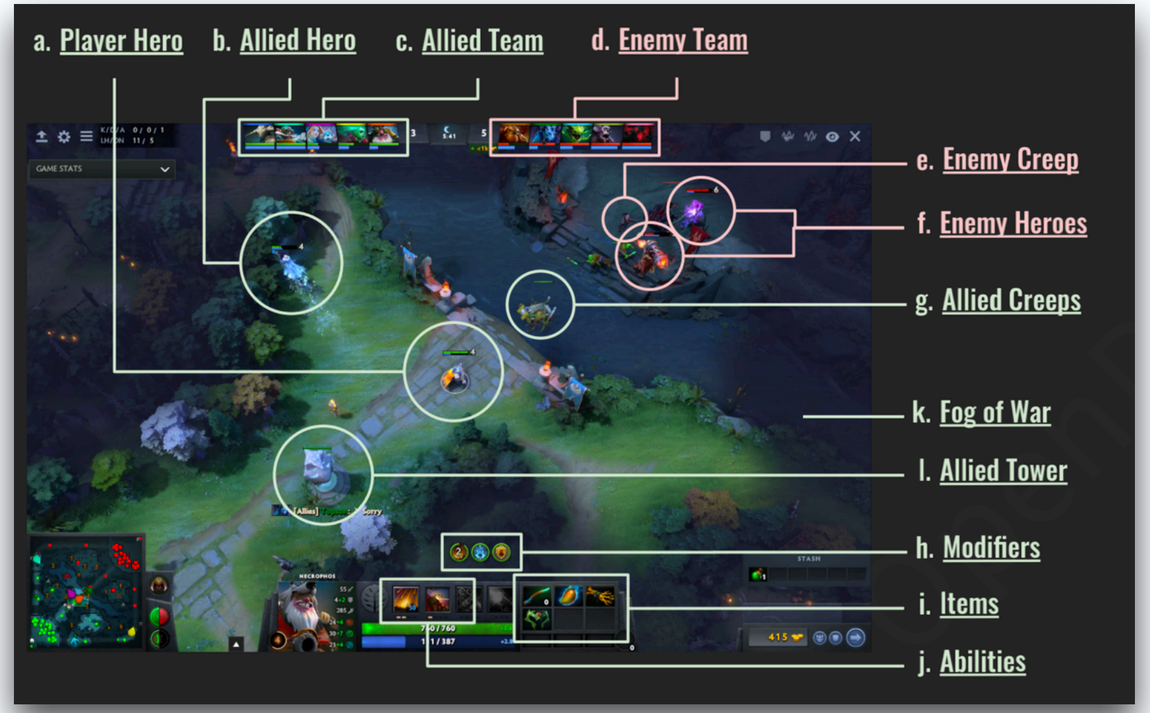
https://openai.com/dall-e-2/

https://github.com/opendilab/DI-star

规划/推理

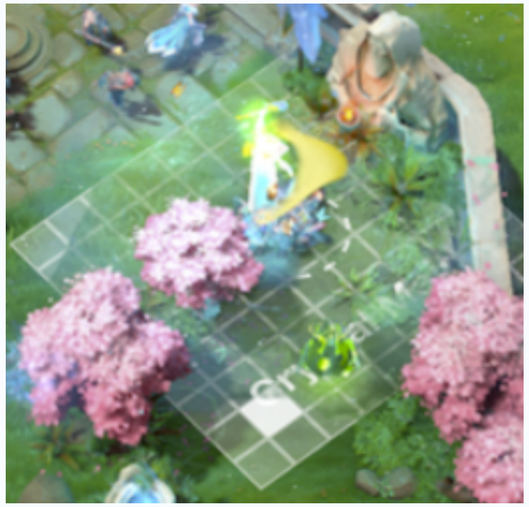
决策

感知+决策



感知： 多种模态信息的提取和融合

决策： 选择目标单位

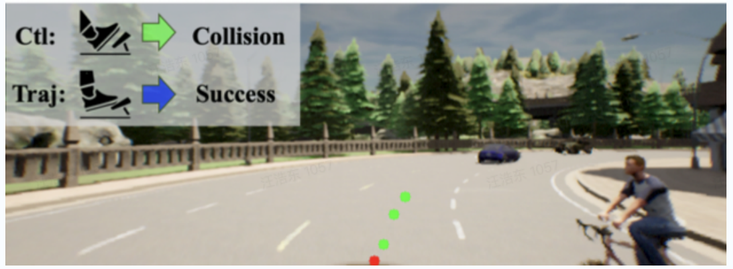


决策： 选择目标位置

感知+决策

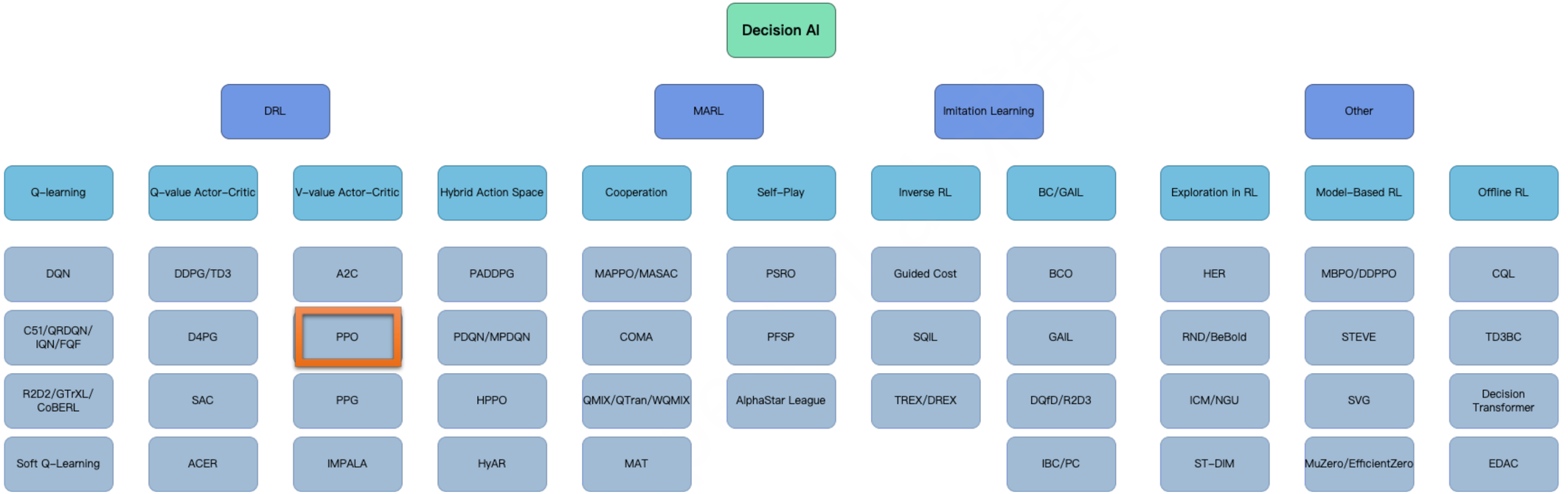
感知：交通标志、路障和车道线的检测

决策：控制车距



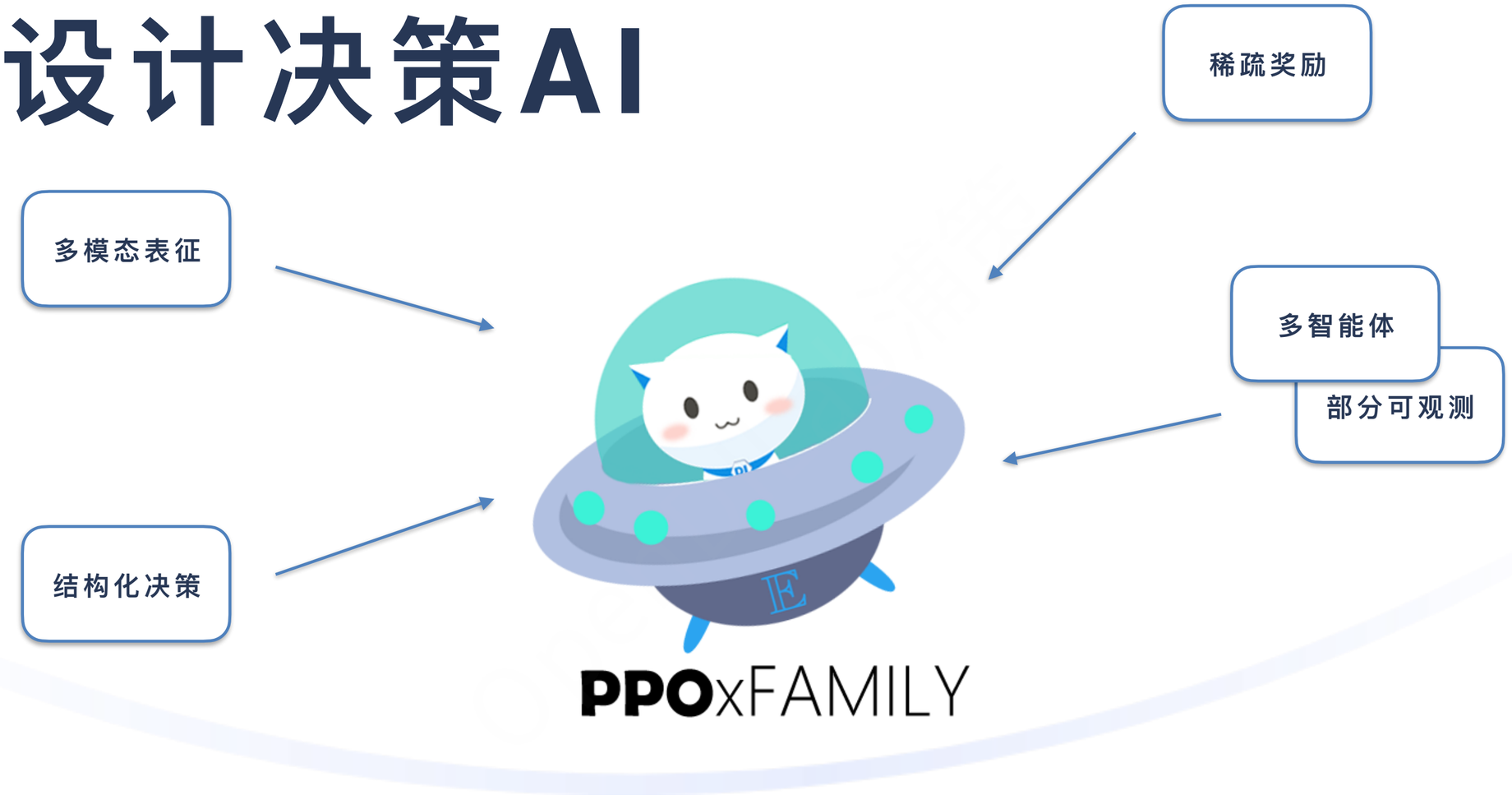
决策：车辆变道

如何设计决策AI



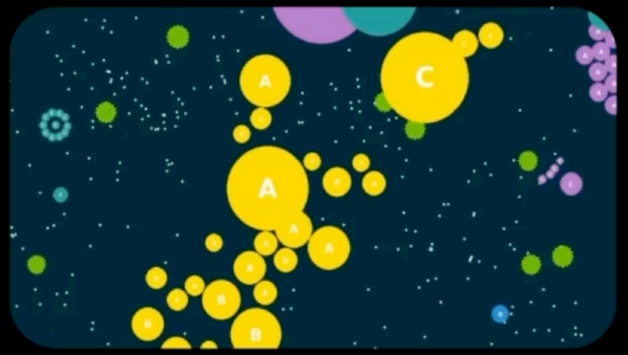
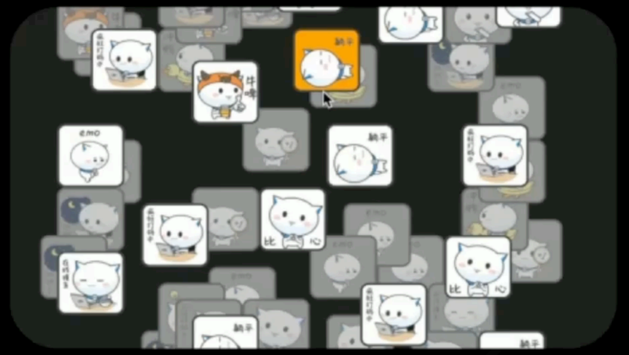
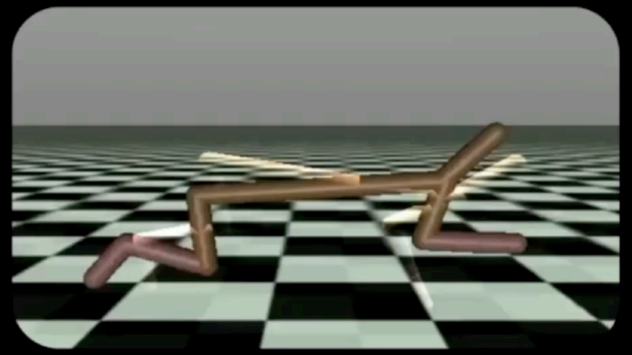
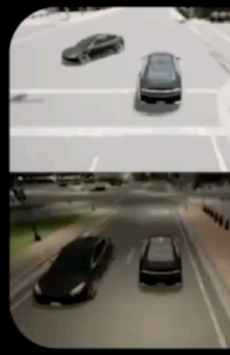
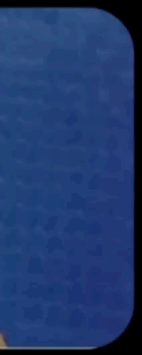
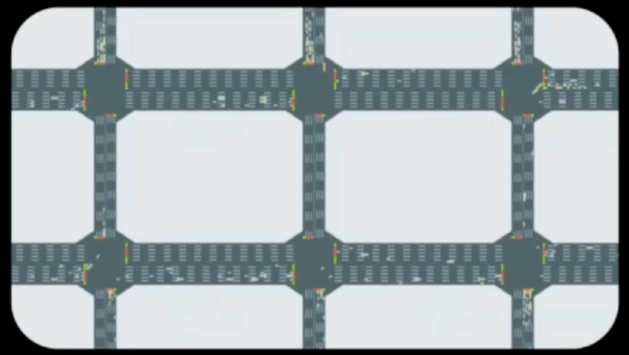
PPO强化学习算法，开启决策AI的万能钥匙

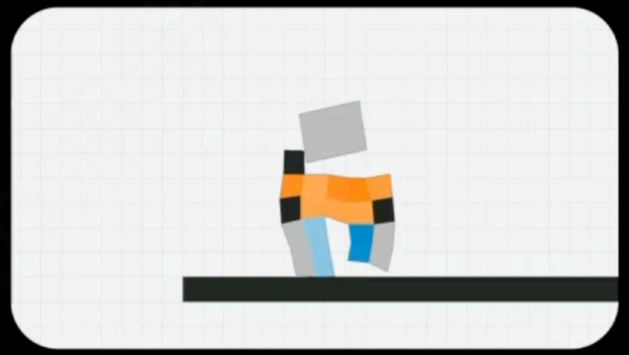
如何设计决策AI



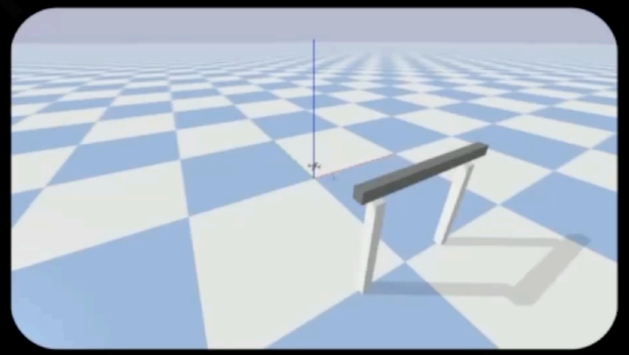
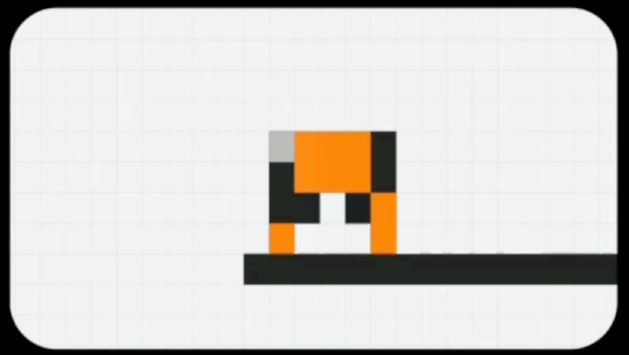
集中一点，登峰造极

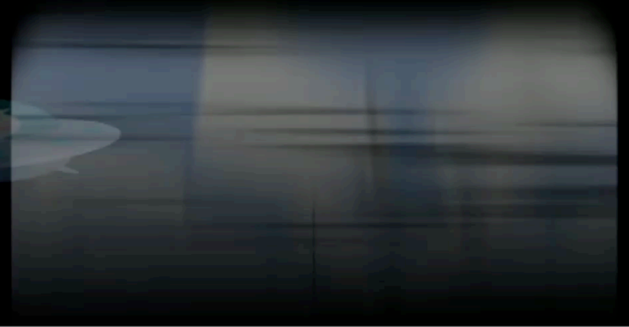
深入理解一种算法 PPO ，灵活应对各种决策场景

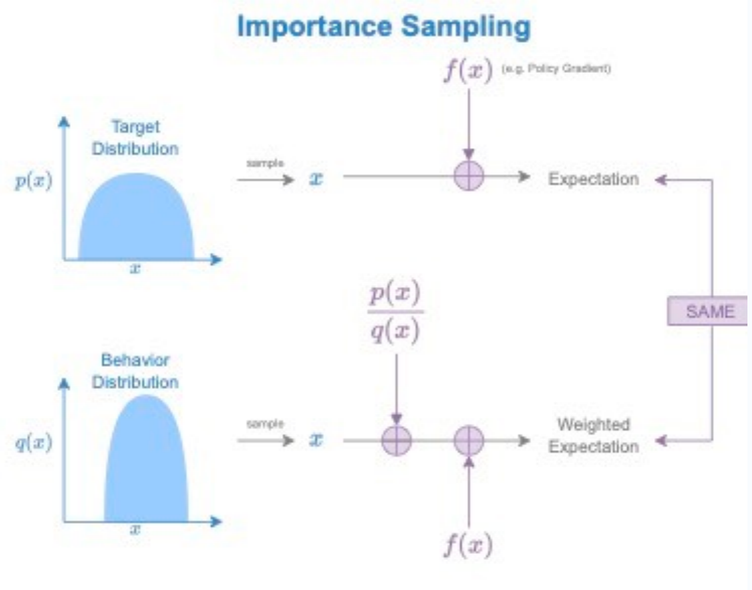





SCORE 2 0
BALL_OWNED 1
high_pass




PPO x Family




```python
def ppo_policy_error(data: namedtuple,
                     clip_ratio: float = 0.2,
                     dual_clip: Optional[float] = None) -> Tuple[namedtuple, namedtuple]:
    logit_new, logit_old, action, adv, weight = data
    if weight is None:
        weight = torch.ones_like(adv)
    dist_new = torch.distributions.categorical.Categorical(logits=logit_new)
    dist_old = torch.distributions.categorical.Categorical(logits=logit_old)
    logp_new = dist_new.log_prob(action)
    logp_old = dist_old.log_prob(action)
    dist_new_entropy = dist_new.entropy()
    if dist_new_entropy.shape != weight.shape:
        dist_new_entropy = dist_new.entropy().mean(dim=1)
    entropy_loss = (dist_new_entropy * weight).mean()
    # policy_loss
    ratio = torch.exp(logp_new - logp_old)
    if ratio.shape != adv.shape:
        ratio = ratio.mean(dim=1)
    surr1 = ratio * adv
    surr2 = ratio.clamp(1 - clip_ratio, 1 + clip_ratio) * adv
    if dual_clip is not None:
        clip1 = torch.min(surr1, surr2)
        clip2 = torch.max(clip1, dual_clip * adv)
        # only use dual_clip when adv < 0
        policy_loss = -(torch.where(adv < 0, clip2, clip1) * weight).mean()
    else:
        policy_loss = (-torch.min(surr1, surr2) * weight).mean()
    with torch.no_grad():
        approx_kl = (logp_old - logp_new).mean().item()
        clipped = ratio.gt(1 + clip_ratio) | ratio.lt(1 - clip_ratio)
        clipfrac = torch.as_tensor(clipped).float().mean().item()
    return ppo_policy_loss(policy_loss, entropy_loss), ppo_info(approx_kl, clipfrac)
```

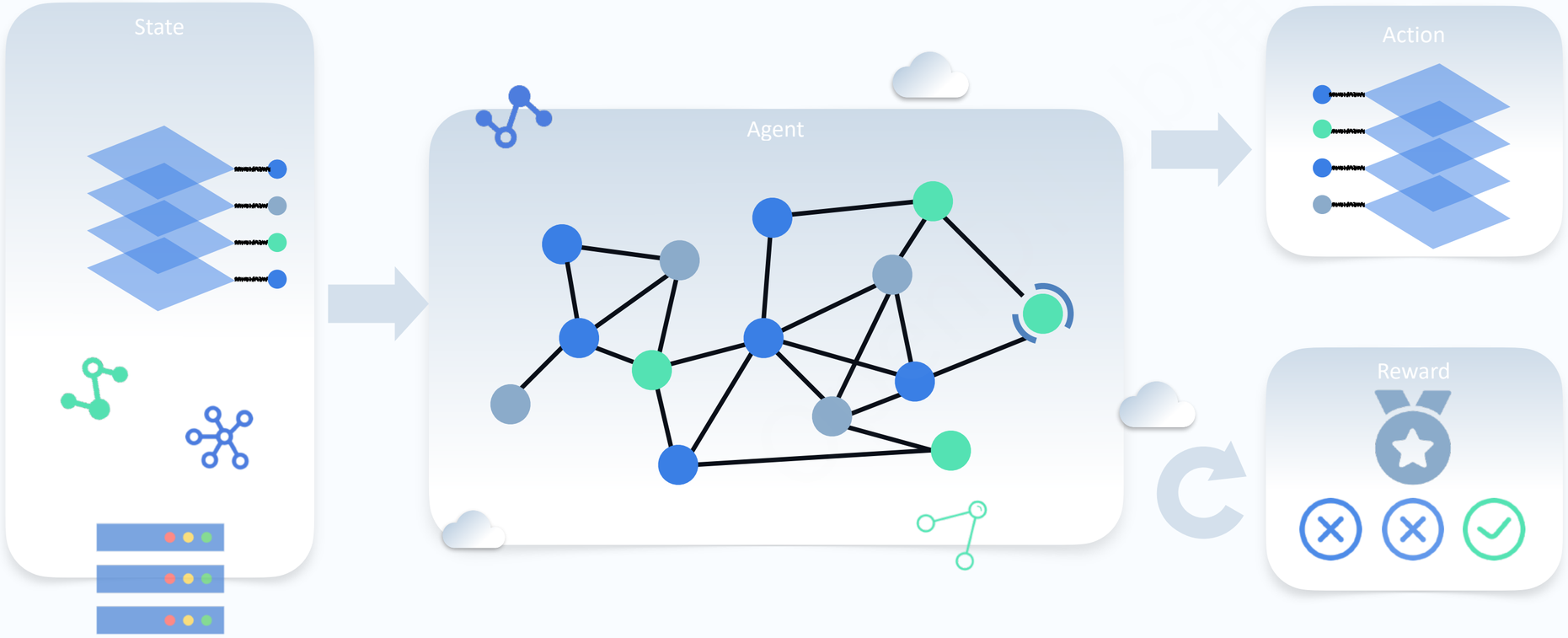
盘清算法理论

理顺代码逻辑

玩转应用实践

Why Deep Reinforcement Learning

为什么选择深度强化学习

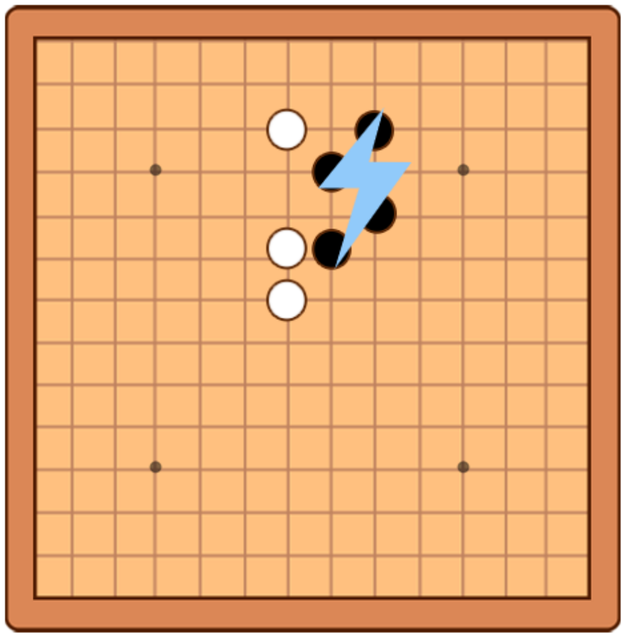


动机

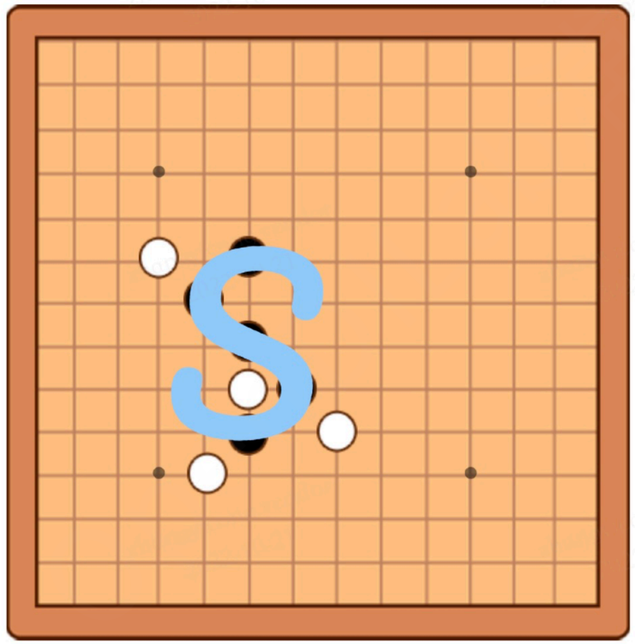
- 搜索最优解的不同方式
- 用强化学习来优化
- 用深度学习来表征
- 决策问题的形式化定义

搜索最优解的不同方式

从模仿中学习



五子棋必胜阵法一：闪电阵法



五子棋必胜阵法二：蛇形阵法

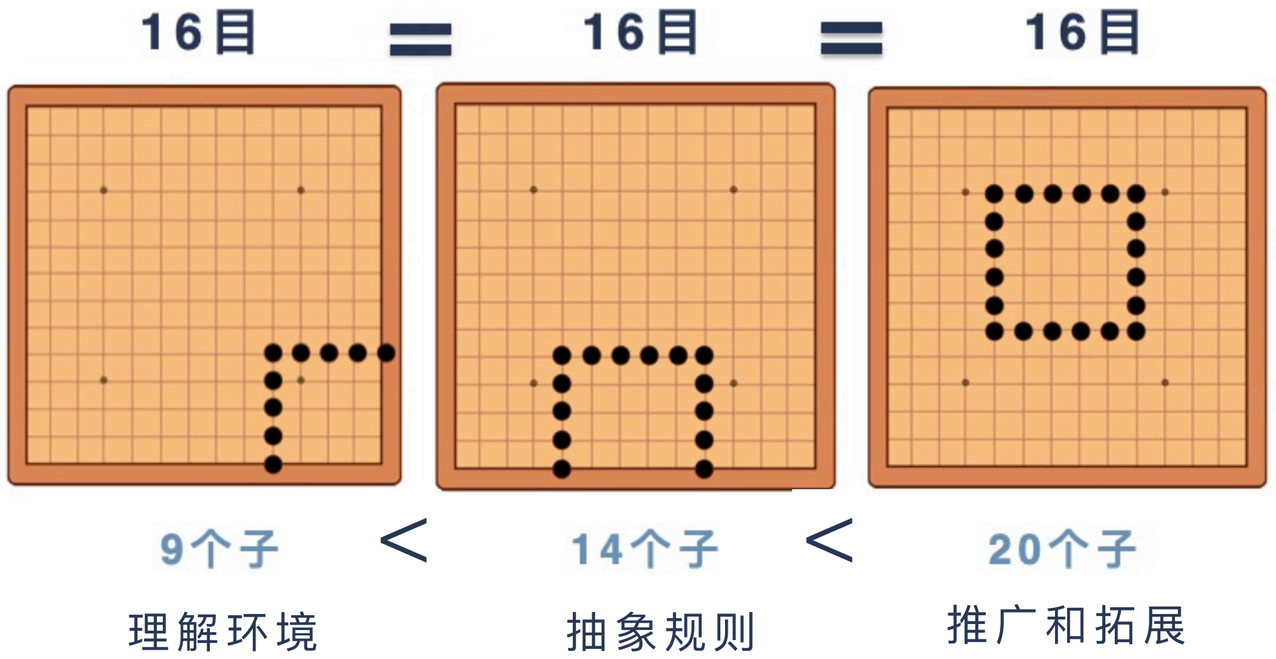
通过**棋谱**来学棋

- 优势：简洁，直观
- 劣势：数据要求高，可迁移性差

从试错中学习

通过**对弈**来学棋

- 优势：可以不断提升与强化
- 劣势：过程复杂，效率和稳定性有待提升

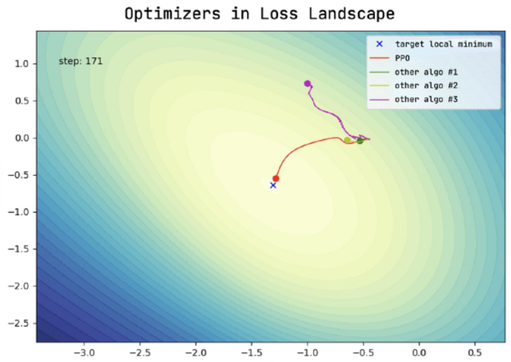


从试错中学习

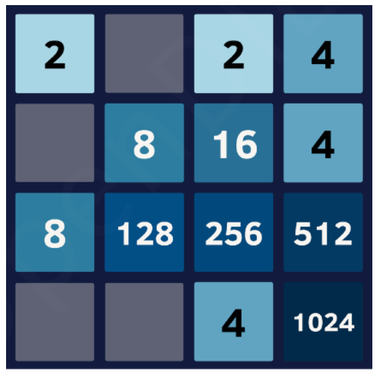
问题

通过对弈来学棋

- 优势：可以不断提升与强化
- 劣势：过程复杂，效率和稳定性有待提升



搜索效率

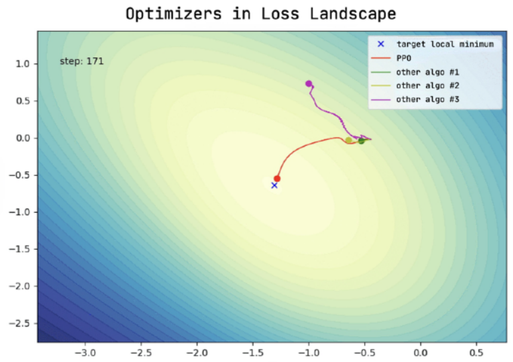


随机性

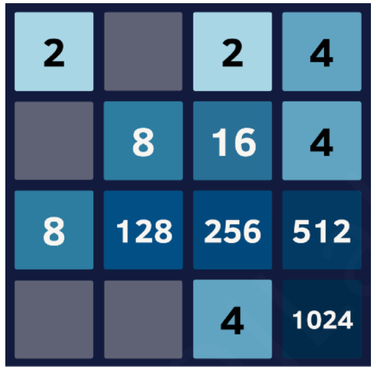
非完全信息

从试错中学习

问题



搜索效率



随机性

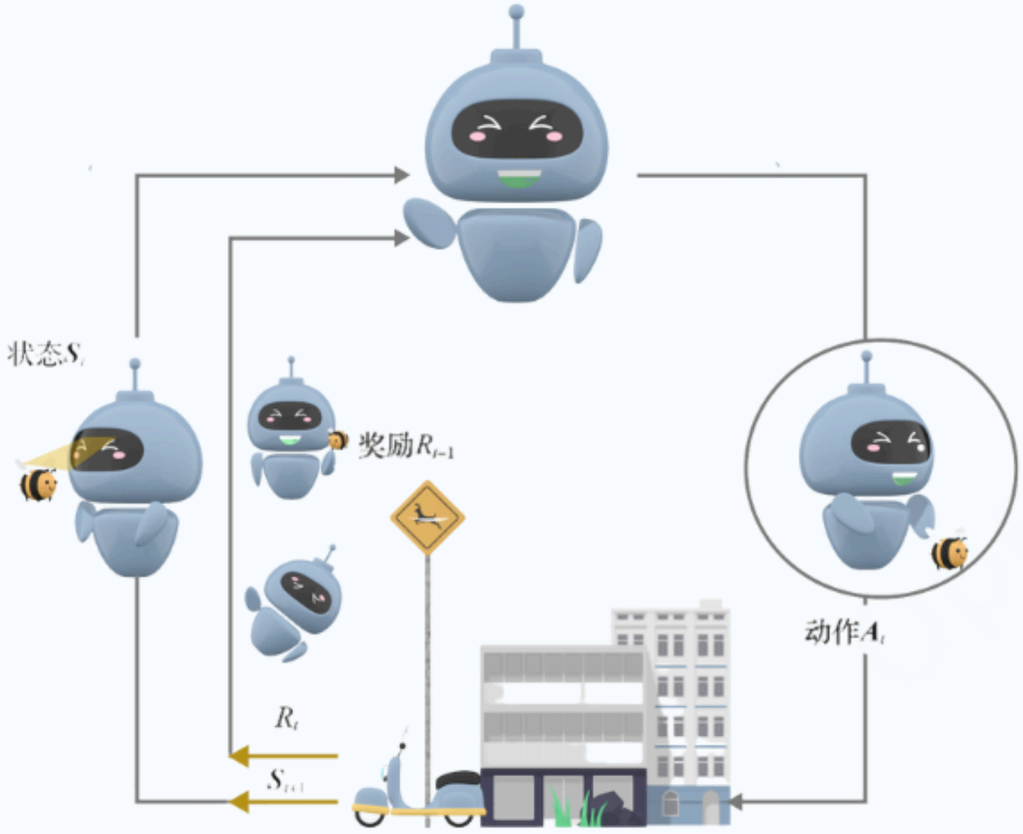
非完全信息

更强大，更通用，更稳定的

搜索最优解的方法

——深度强化学习

强化学习的基本设定



环境是对决策问题的**抽象和标准化**

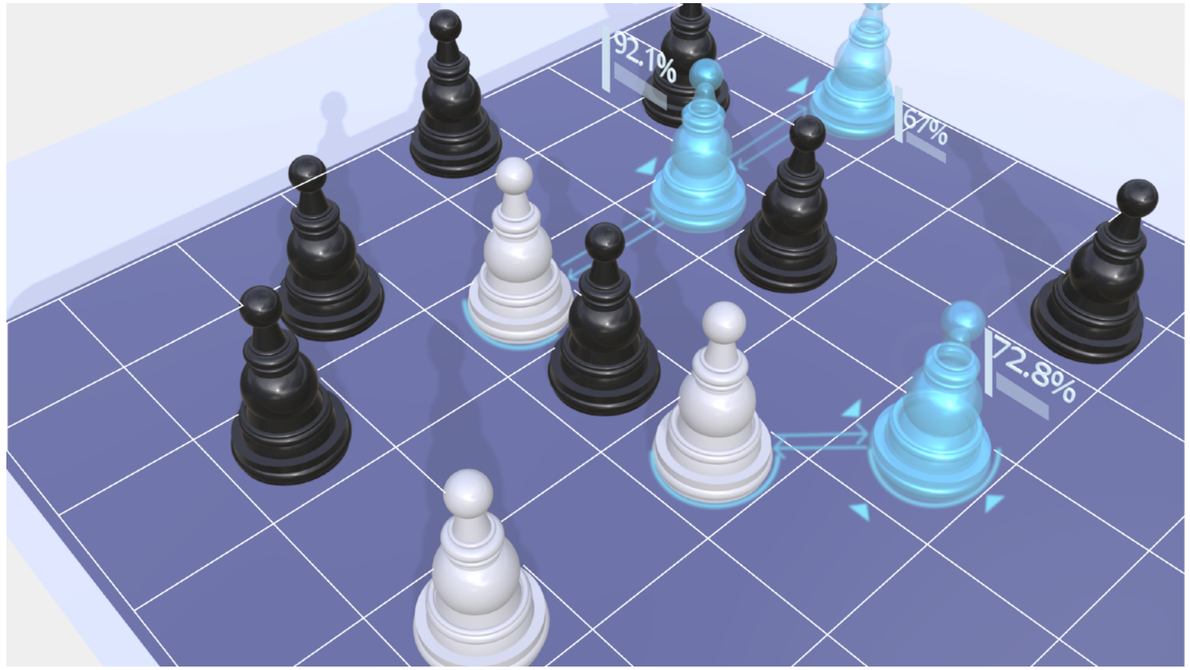
智能体的目标是找到该环境上的**最优解**

智能体和环境通过交互来进行**在线学习**

不断强化自身，演化出无限可能

强化学习的特点

环境未知性：非完全信息下的决策，例如斗地主中不知道对方的牌面
环境不确定性：可表现为状态和奖励的各种不确定性，例如随机生成的游戏道具



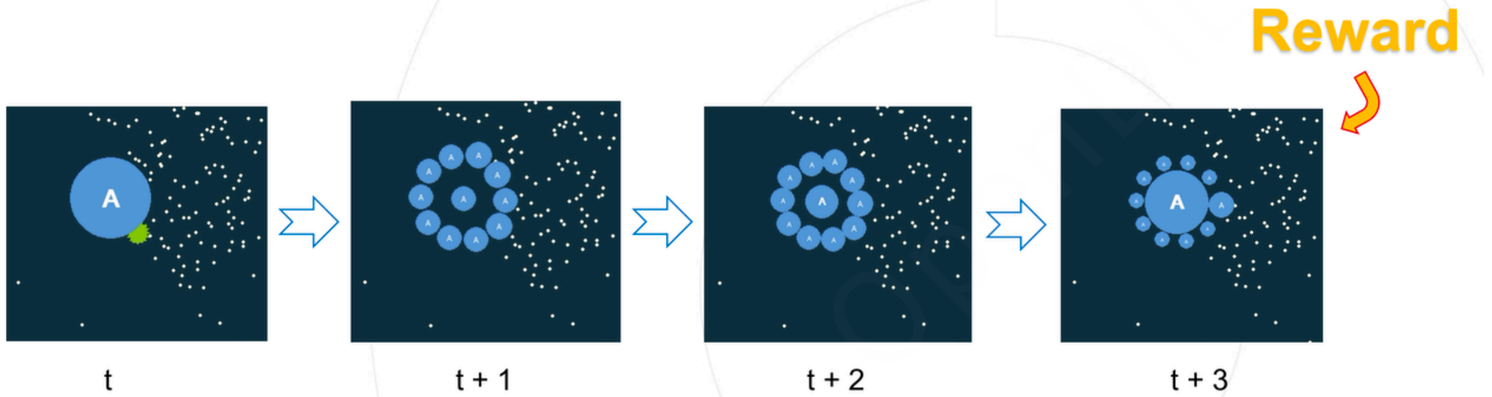
对比**传统搜索方法**，
可以建模环境的**未知性和不确定性**
自主学到更抽象的搜索策略

强化学习的特点

对比**监督学习**（从固定的标签中学习）

需要从**延迟性的，**

间接的奖励中学习。



延迟性：当前决策行为的奖励可能时间上滞后很久才能获得，例如，前人栽树，后人乘凉

间接：相对于直接通过标签来监督，奖励信息是间接的

注：延迟性奖励只出现在部分决策问题中，主要对应那些需要一连串长序列决策行为的问题

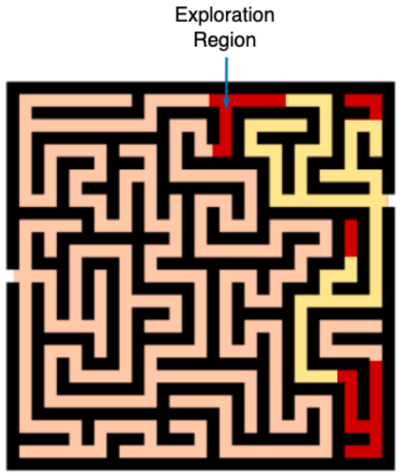
GoBigger: https://github.com/opendilab/GoBigger

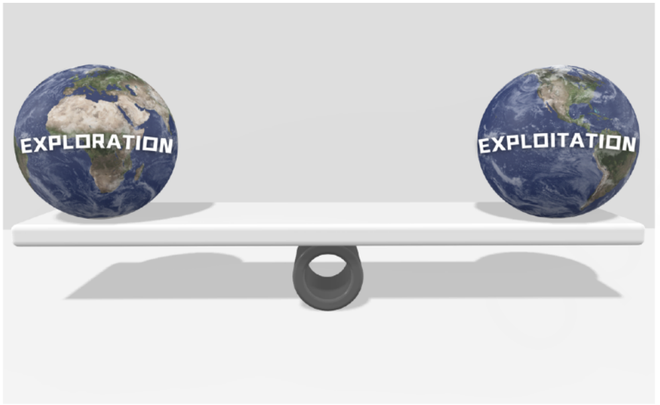
强化学习的特点

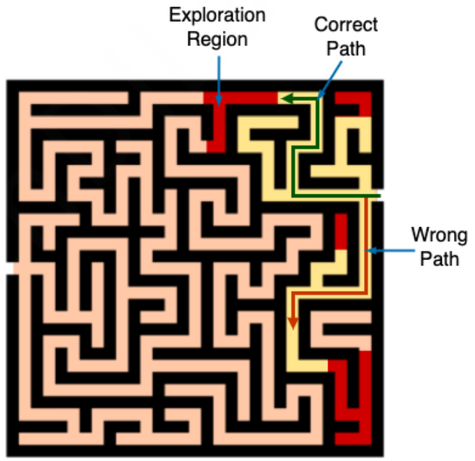
对比**离线学习**，

需要**平衡探索和利用**，

需要从**非独立同分布**的数据中学习。







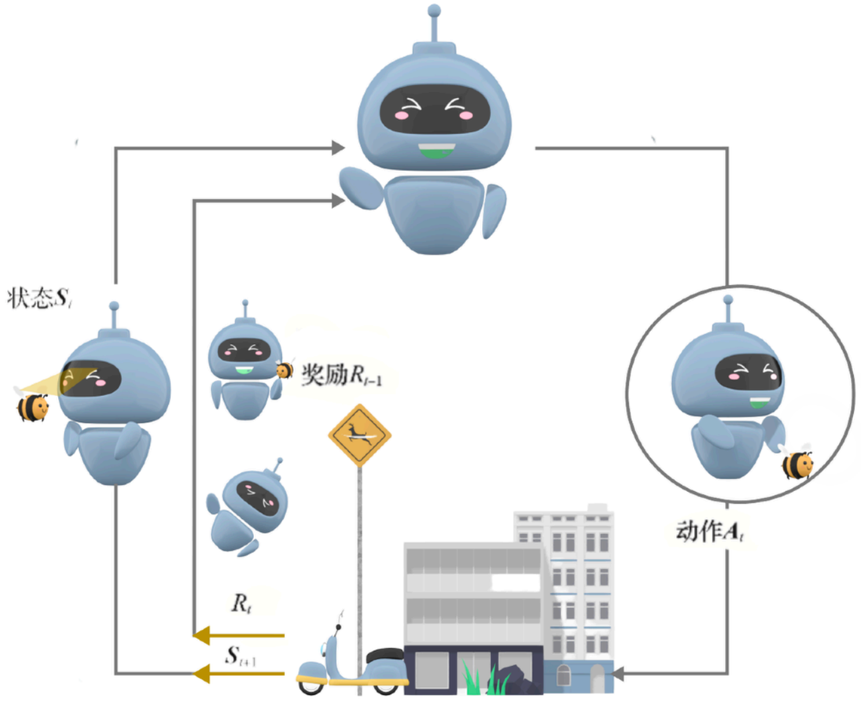
探索（Exploration）：找到新的有价值的数据和信息，例如探索迷宫中的未知部分

利用（Exploitation）：根据已有数据挖掘出知识内涵，例如对迷宫中已知部分进行分析

非独立同分布数据问题定义：https://inria.github.io/scikit-learn-mooc/python_scripts/cross_validation_time.html

强化学习的特点

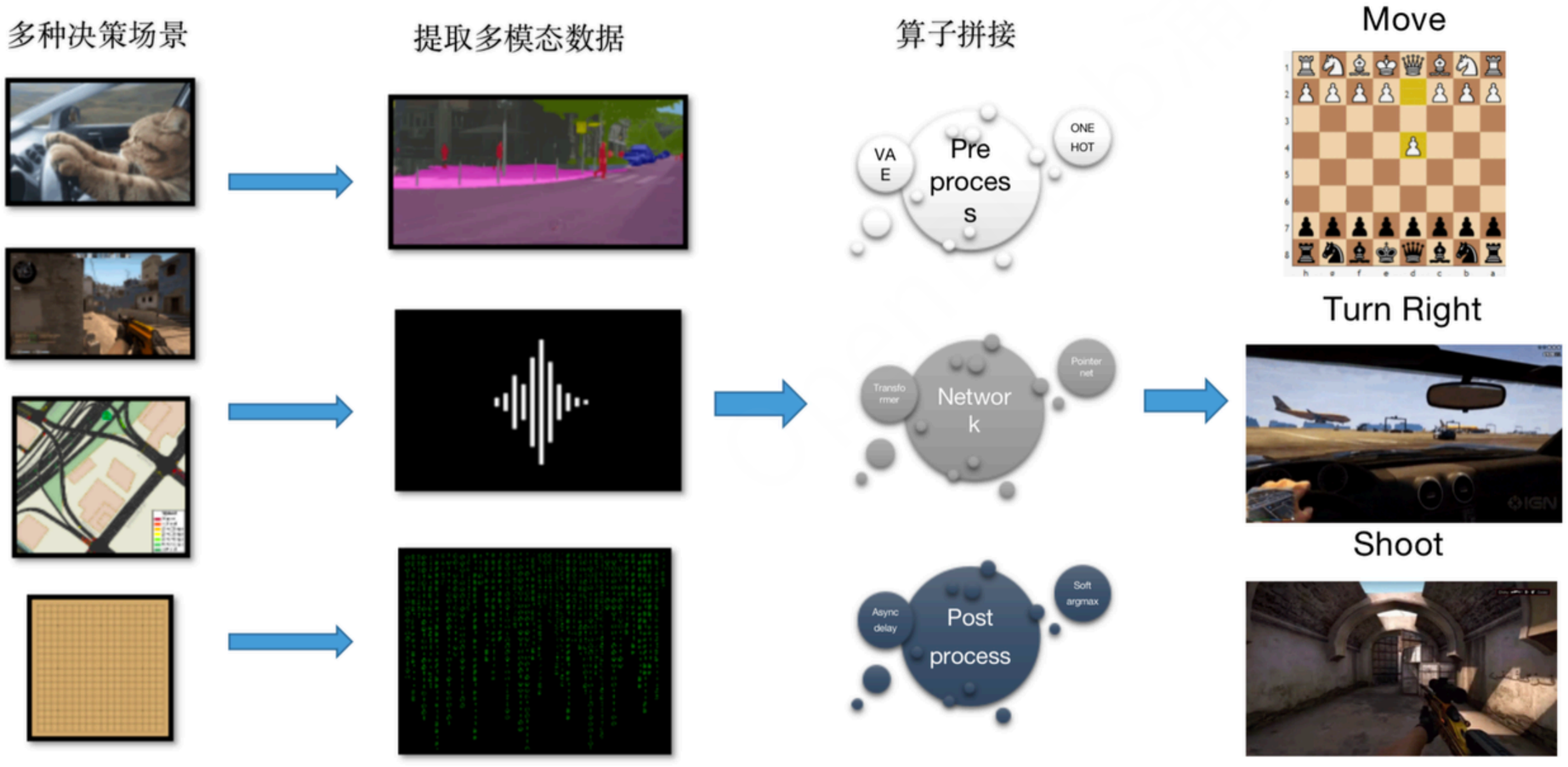
- 建模环境
- 从奖励中学习
- 平衡利用和探索



为什么要结合深度学习

强大的
非线性
表征能力

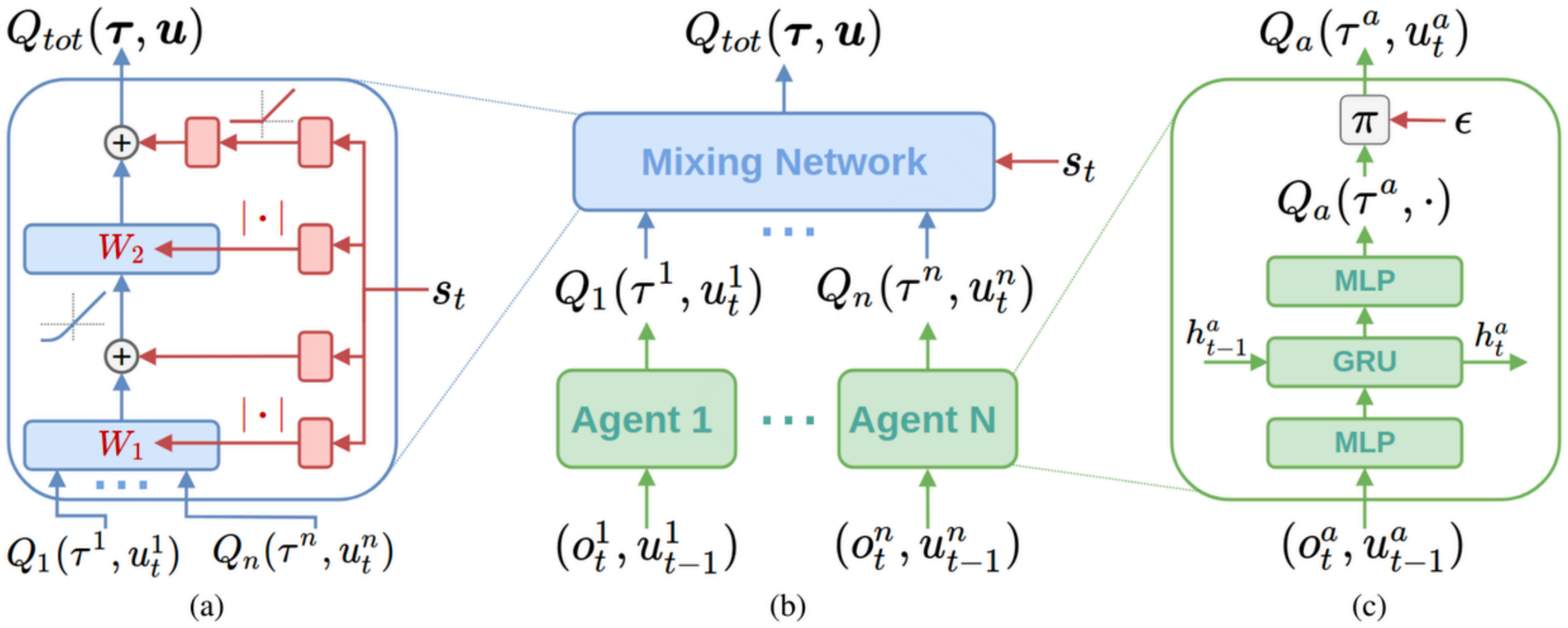
处理多种决策场景的不同输入和输出（即多模态观察空间和混合动作空间）



为什么要结合深度学习

强大的
非线性
表征能力

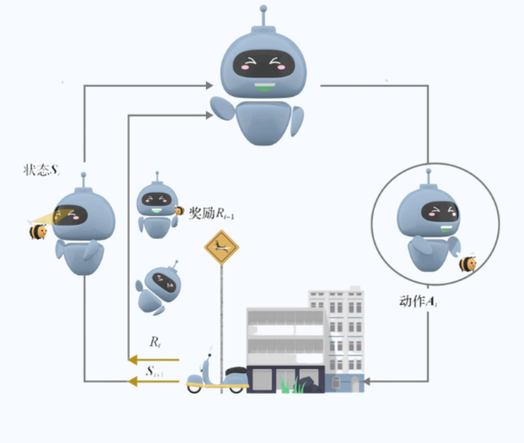
建模强化学习算法中独有的一些算法概念（以多智能体协作为例）



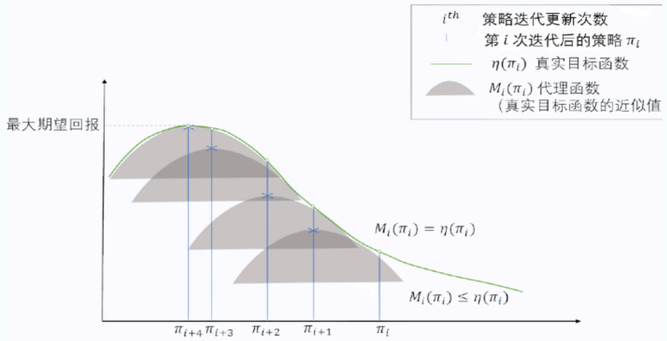
QMIX: https://arxiv.org/pdf/1803.11485.pdf

形式化定义RL

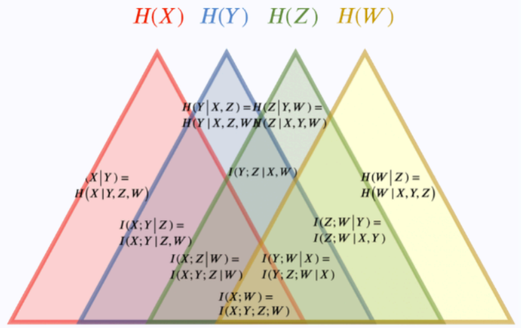
决策问题 → 马尔科夫决策过程



RL问题环境

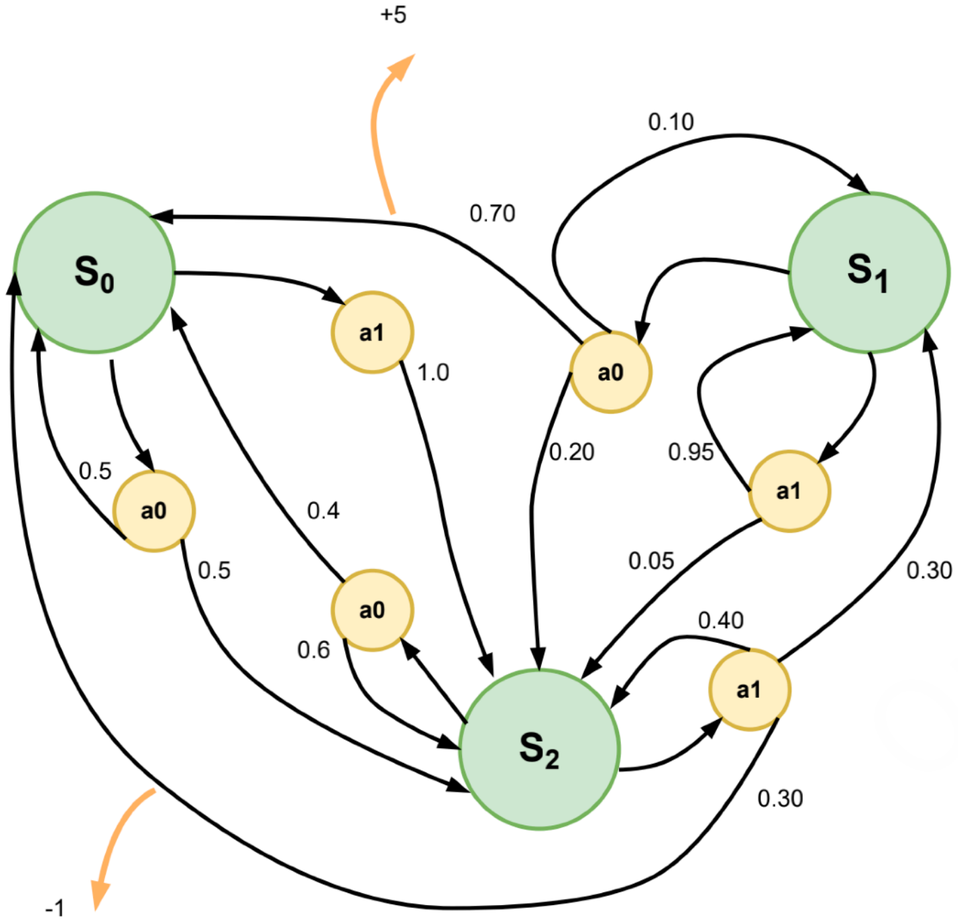


RL优化目标



马尔科夫性

强化学习的问题定义



描述环境的五元组 $< S, A, P, R, \rho_0 >$

- **状态集合** S
- **动作集合** A
- **在状态之间转移的规则**，即转移概率 $P(s_{t+1} \mid s_t, a_t)$，其中 $s_t \in S, a_t \in A$
- 状态转移后，**获得”即时奖励“的规则** $r_t = R(s_t, a_t, s_{t+1})$
- **初始状态分布** ρ_0

注：t 表示一个 episode 内的时间步（timestep）

强化学习的优化目标

- 智能体策略 π，如果是确定性策略则为函数$a_t = \pi(s_t)$，如果是随机性策略则为条件概率分布 $\pi(a_t|s_t)$，如果将表示策略的参数记为 θ，策略通常又写作 π_θ

- 轨迹 τ，策略和环境进行交互，产生的状态动作序列（按照交互的时间先后顺序），假设有T步决策（T由环境本身决定），则轨迹 $\tau = (s_0, a_0, s_1, a_1, \dots)$

强化学习的优化目标

- 回报 G_t，一般情况下，强化学习希望能够最大化整个交互周期内能够获得的总奖励。并且，我们使用折扣因子 γ 来平衡短期和长期的奖励，于是，我们得到回报的定义为

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

那么强化学习就可以定义为如下的优化问题，

寻找最大化期望回报的策略 $argmax_{\pi_\theta} \mathbb{E}_{\tau \sim \pi_\theta}[G_t(\tau)]$。

Why focus on PPO

为什么聚焦于PPO

目的

深入浅出策略梯度

策略梯度各种改进的集大成之作：PPO

$$L_\theta = \min(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)}\hat{A}^{\theta_k}(s_t,a_t), \text{clip}(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)}, 1-\epsilon, 1+\epsilon)\hat{A}^{\theta_k}(s_t,a_t))$$

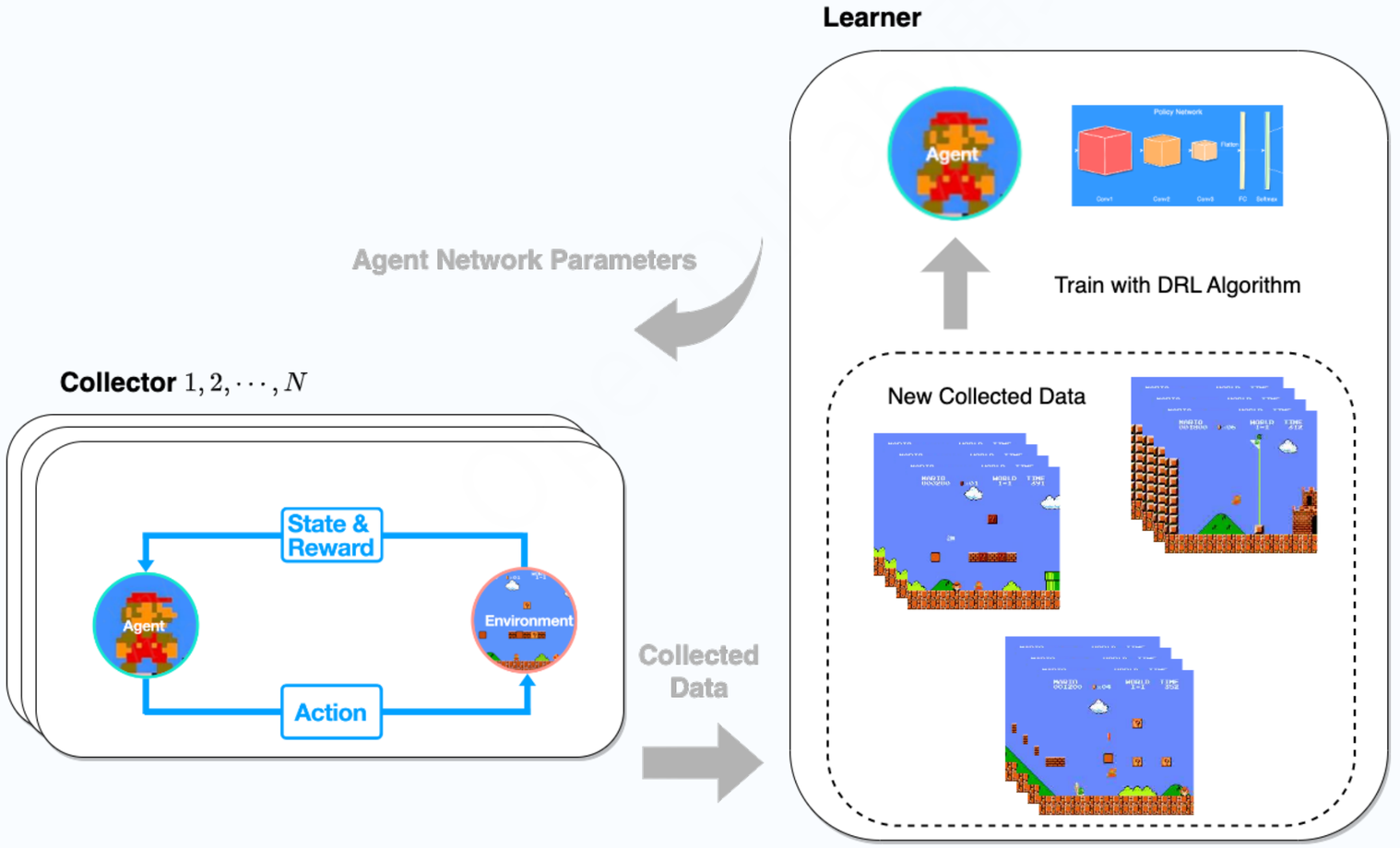
深入浅出策略梯度

- 收集数据
- 设计目标
- 优化策略

https://www.bilibili.com/video/BV1e841157Um/?spm_id_from=333.999.0.0

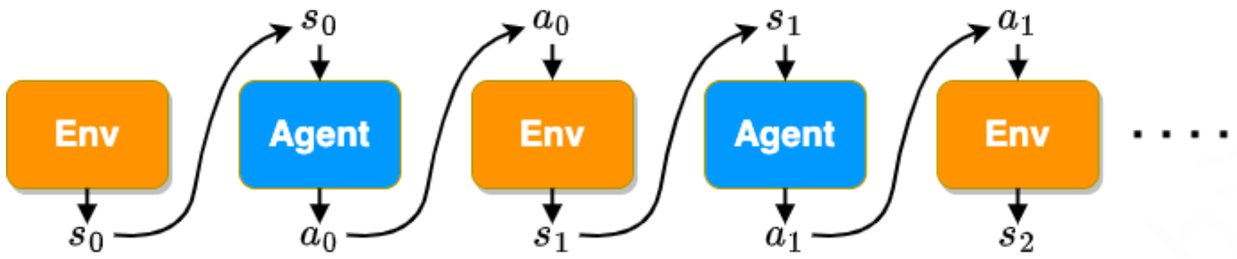
在线收集数据

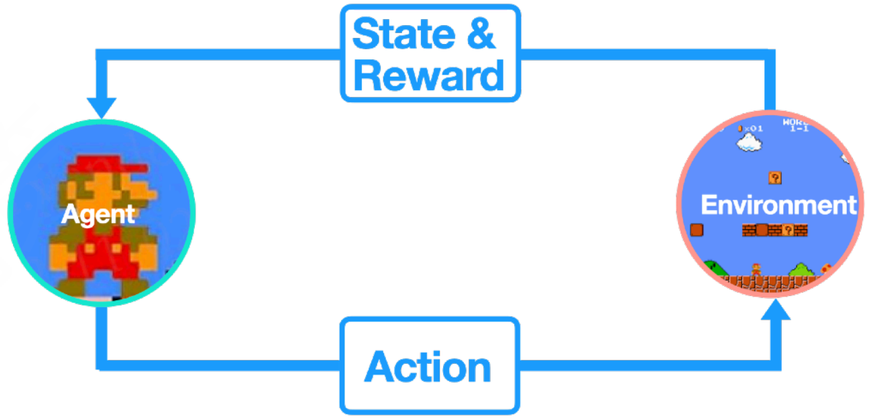
数据收集器（Collector） → 学习器（Learner）

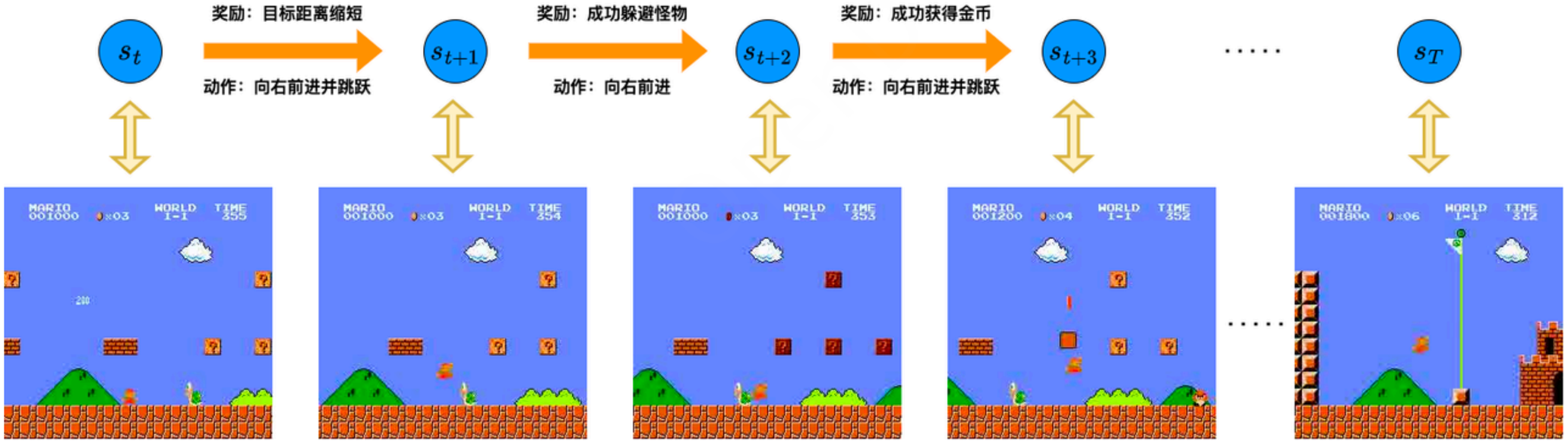


数据收集

- 智能体和环境交互产生轨迹



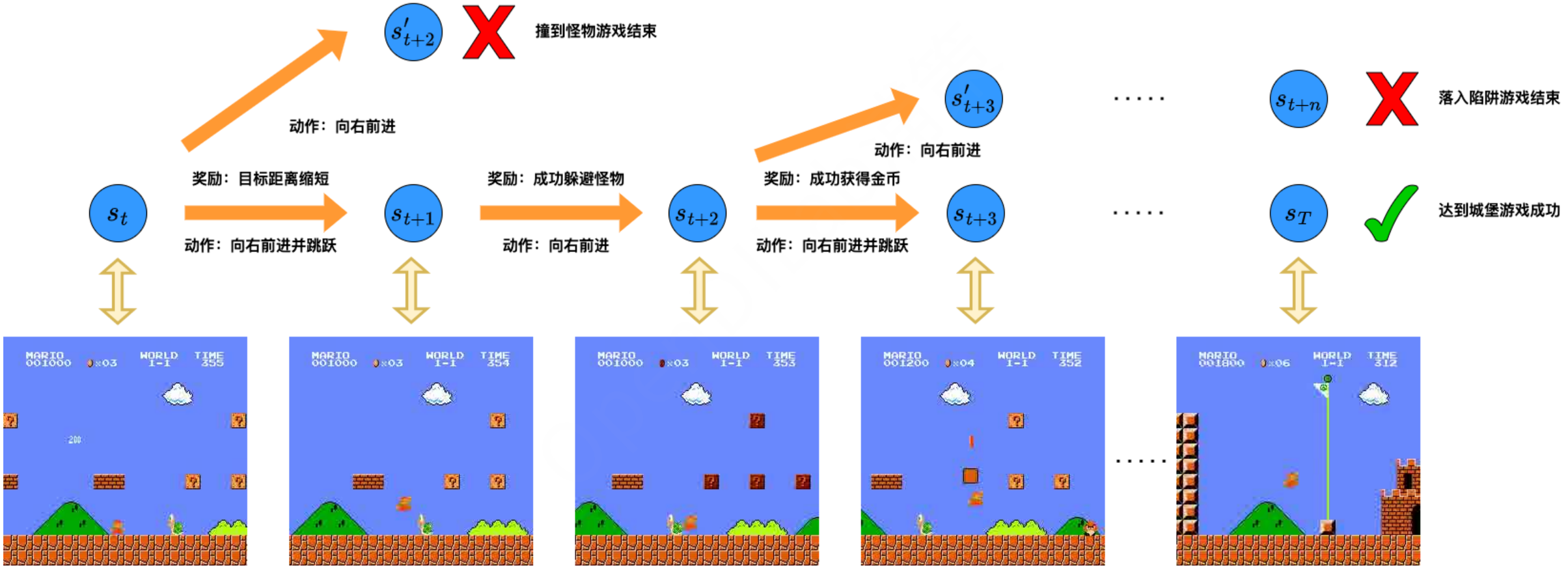




https://di-engine-docs.readthedocs.io/zh_CN/latest/13_envs/gym_super_mario_bros_zh.html

数据收集+探索

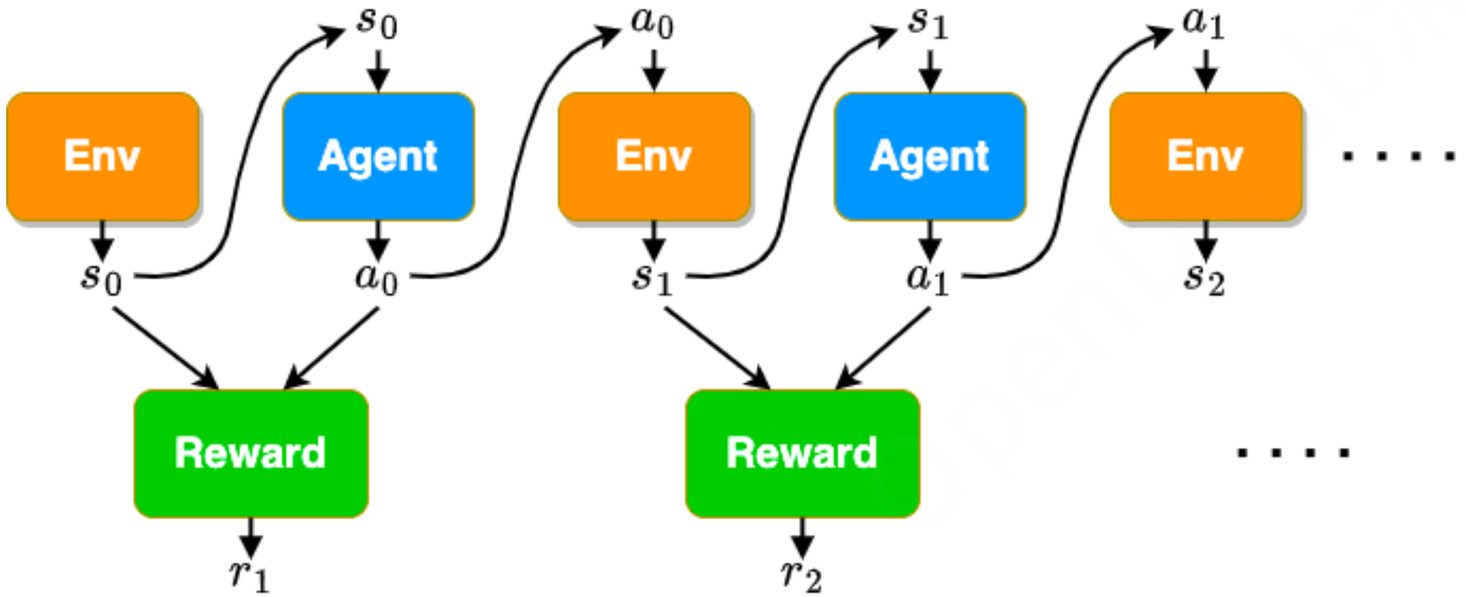
- 探索：尝试当前策略之外的新事物
- 不断探索到新的有价值的信息，才能不断强化智能体



https://github.com/opendilab/awesome-exploration-rl

设计目标

$$P_\theta(\tau) = \rho(s_0)P_\theta(a_0|s_0)P(s_1|s_0,a_0)P_\theta(a_1|s_1)P(s_2|s_1,a_1)\cdots$$



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

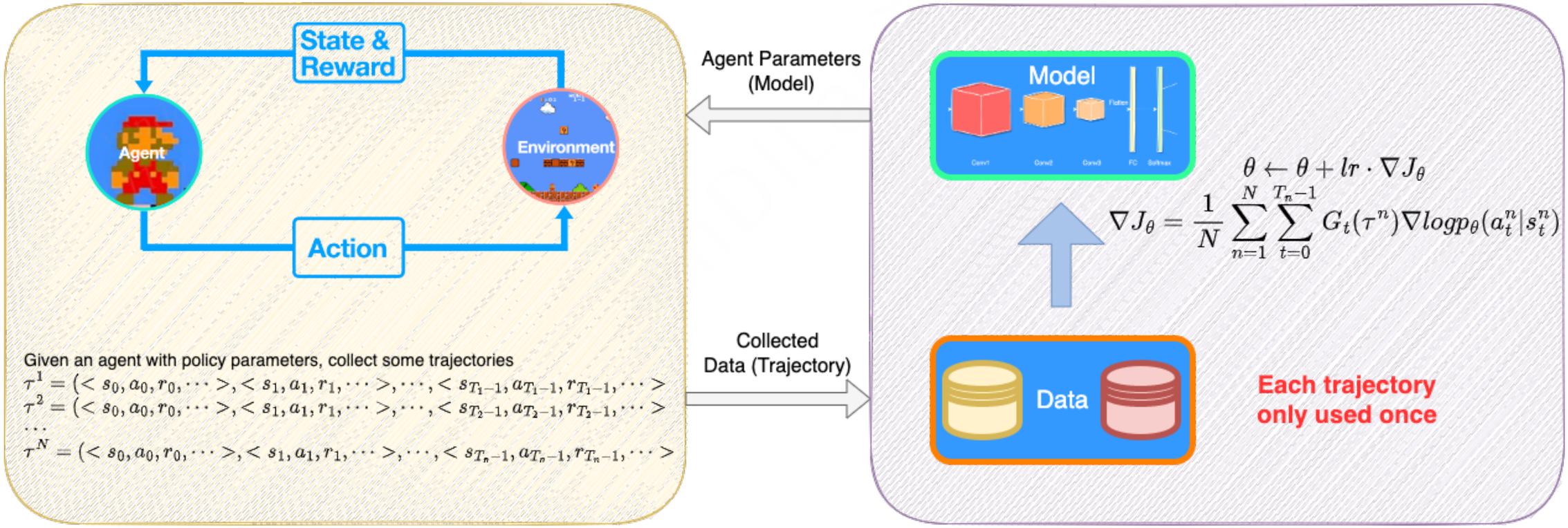
轨迹生成的角度

- 环境的规则 $P(s_{t+1} \mid s_t, a_t)$
- 智能体的行为 $P_\theta(a_t \mid s_t)$
- 初始状态分布 $\rho(s_0)$

注：$P_\theta(\tau)$ 和前文中的 $\tau \sim \pi_\theta$ 都是指，轨迹是根据参数化的策略生成的

优化策略

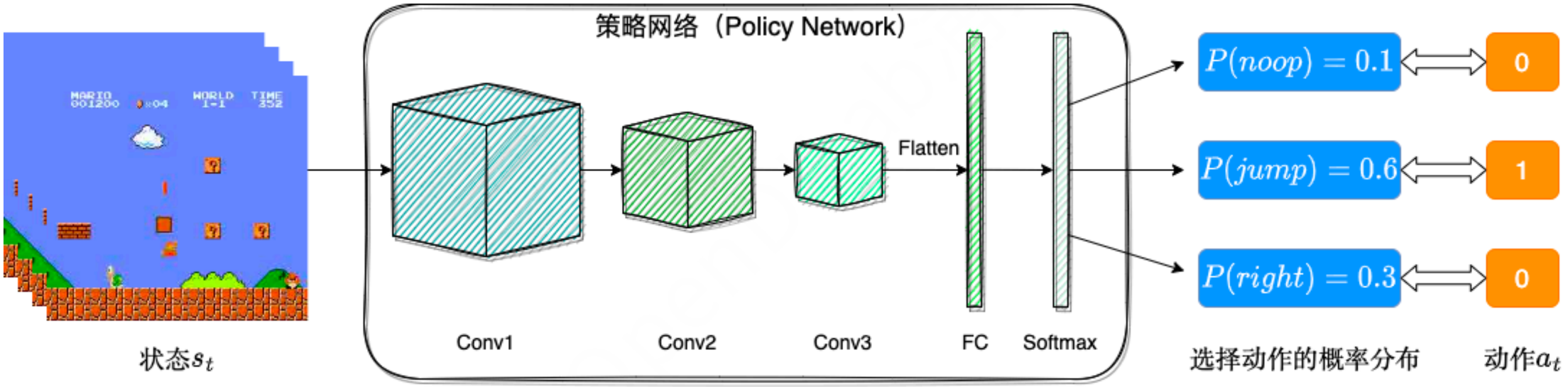
策略梯度定理：增大策略选择高回报值动作的概率，减小策略选择低回报值动作的概率 -> 多步MDP



https://github.com/opendilab/PPOxFamily/tree/main/chapter1_overview/chapter1_supp_pg.pdf

优化策略

策略网络（Policy Network）

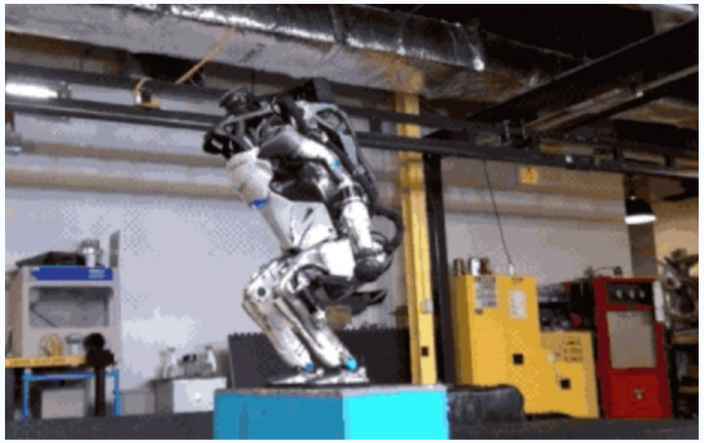


$$\nabla J_\theta = \frac{1}{N}\sum_{n=1}^{N}\sum_{t=0}^{T_n-1} G_t(\tau)\nabla log p_\theta(a_t \mid s_t)$$

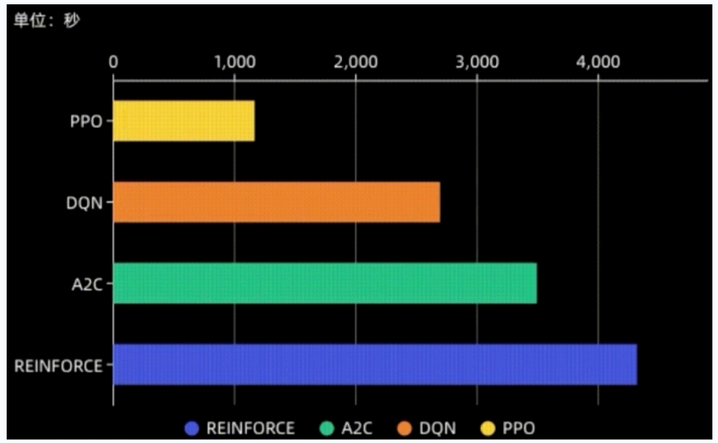
+ 随机梯度下降（SGD）

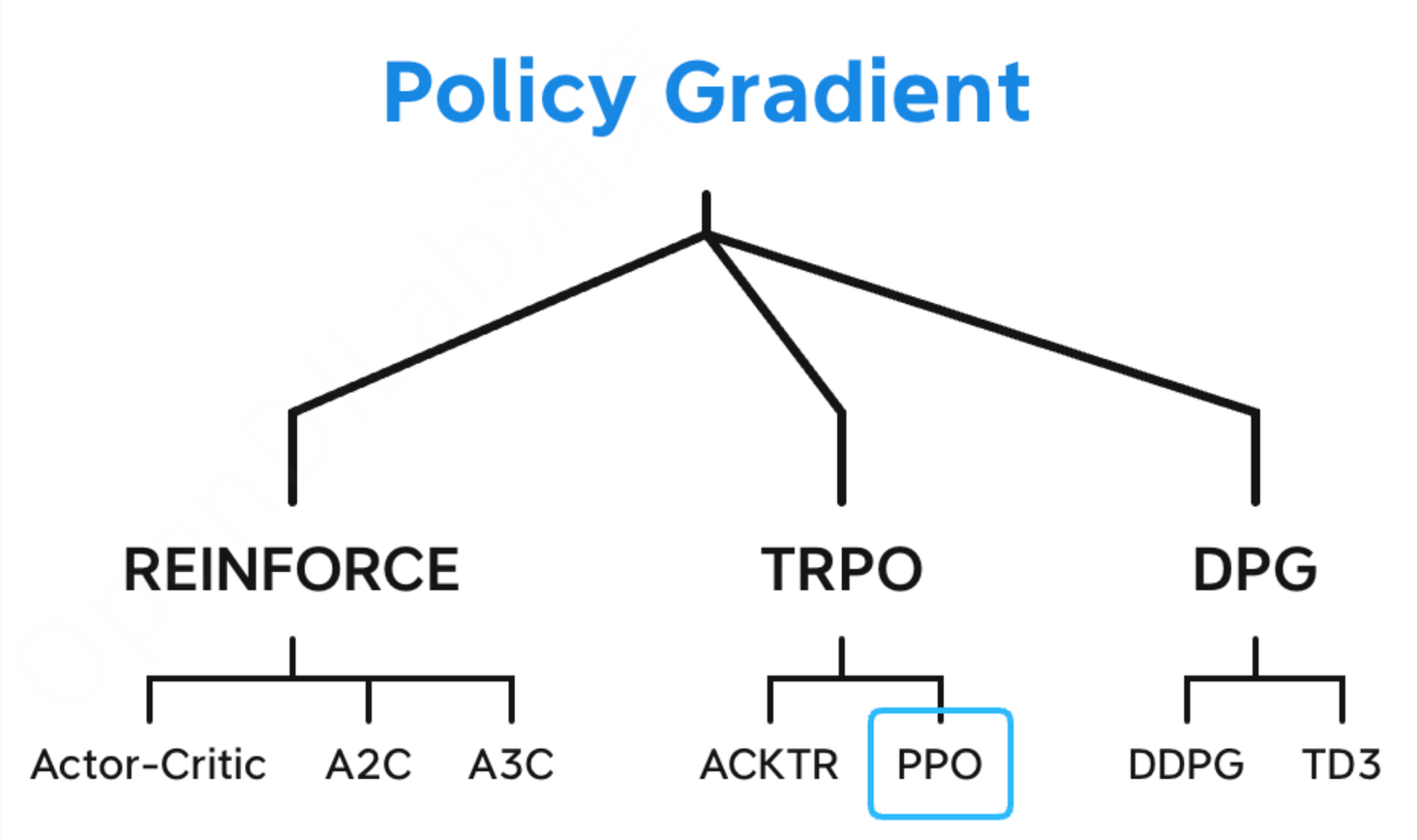
策略梯度的发展史

稳定性



效率





Actor(演员)-Critic(评论家)

- 问题：REINFORCE 策略梯度方法使用 episode return，梯度方差大，数据利用效率低
- 解决思路：结合 Value-Based RL 方法的思路，训练一个价值函数网络来估计这个值

演员 $\pi_\theta(a|s)$

采取动作使评论家满意的策略

评论家 $Q_\Phi(s,a)$

学会准确估计演员策略所采取动作价值的值函数

$$\nabla J_\theta = \frac{1}{N}\sum_{n=1}^{N}\sum_{t=0}^{T_n-1} Q_\phi(s_t,a_t)\nabla log p_\theta(a_t|s_t) \quad Q_\phi(s_t,a_t) = E_{\tau\sim\pi}\left[\sum_{l=0}^{\infty}\gamma^l r^{t+l}\right]$$

https://zhuanlan.zhihu.com/p/21262246

A2C

- 问题：REINFORCE 方法梯度高方差，Actor-Critic 存在梯度偏差
- 解决思路：结合二者的优势，并减去一个基线函数来标准化，从而在保持无偏梯度估计的同时减小方差

$$\nabla J_\theta = \frac{1}{N}\sum_{n=1}^{N}\sum_{t=0}^{T_n-1} G_t(\tau)\,\nabla log p_\theta(a_t \mid s_t)$$

+ no bias　　- higher variance

$$\nabla J_\theta = \frac{1}{N}\sum_{n=1}^{N}\sum_{t=0}^{T_n-1} Q_\phi(s_t, a_t)\,\nabla log p_\theta(a_t \mid s_t)$$

- not unbiased　　+ lower variance

$$\nabla J_\theta = \frac{1}{N}\sum_{n=1}^{N}\sum_{t=0}^{T_n-1} (G_t(\tau) - V_\phi(s_t))\,\nabla log p_\theta(a_t \mid s_t)$$

+ no bias　　+ lower variance

可以用 Advantage 函数来表示

https://github.com/opendilab/PPOxFamily/tree/main/chapter1_overview/chapter1_supp_a2c.pdf

Advantage Actor-Critic (A2C): https://arxiv.org/pdf/1602.01783.pdf

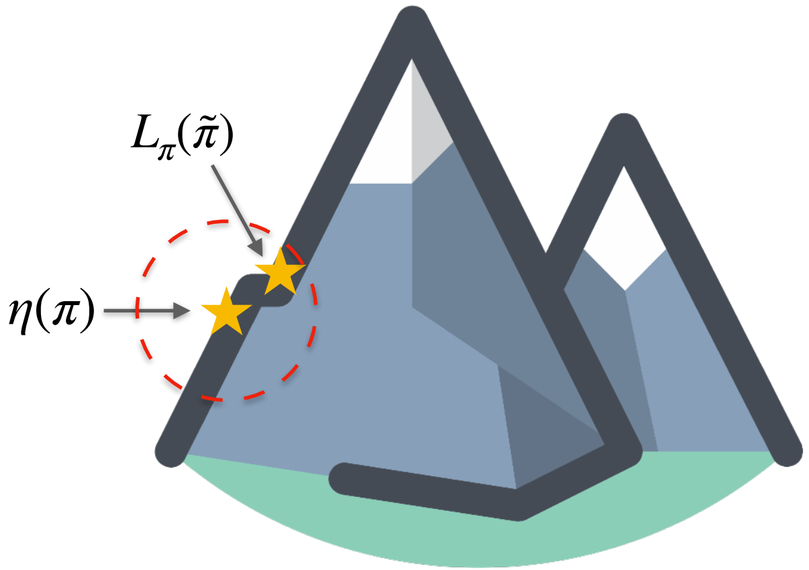
TRPO：信赖域

问题：如何让策略持久稳定提升

解决思路：可靠的方向，合适的步长

$$\eta(\tilde{\pi}) = \eta(\pi) + \sum_s \rho_{\tilde{\pi}}(s) \sum_a \tilde{\pi}(a \mid s) A_\pi(s,a) \quad \text{（新旧策略之间的关系）}$$

$$\eta(\tilde{\pi}) \approx L_\pi(\tilde{\pi}) = \eta(\pi) + \sum_s \rho_\pi(s) \sum_a \tilde{\pi}(a \mid s) A_\pi(s,a) \quad \text{（替代函数）}$$



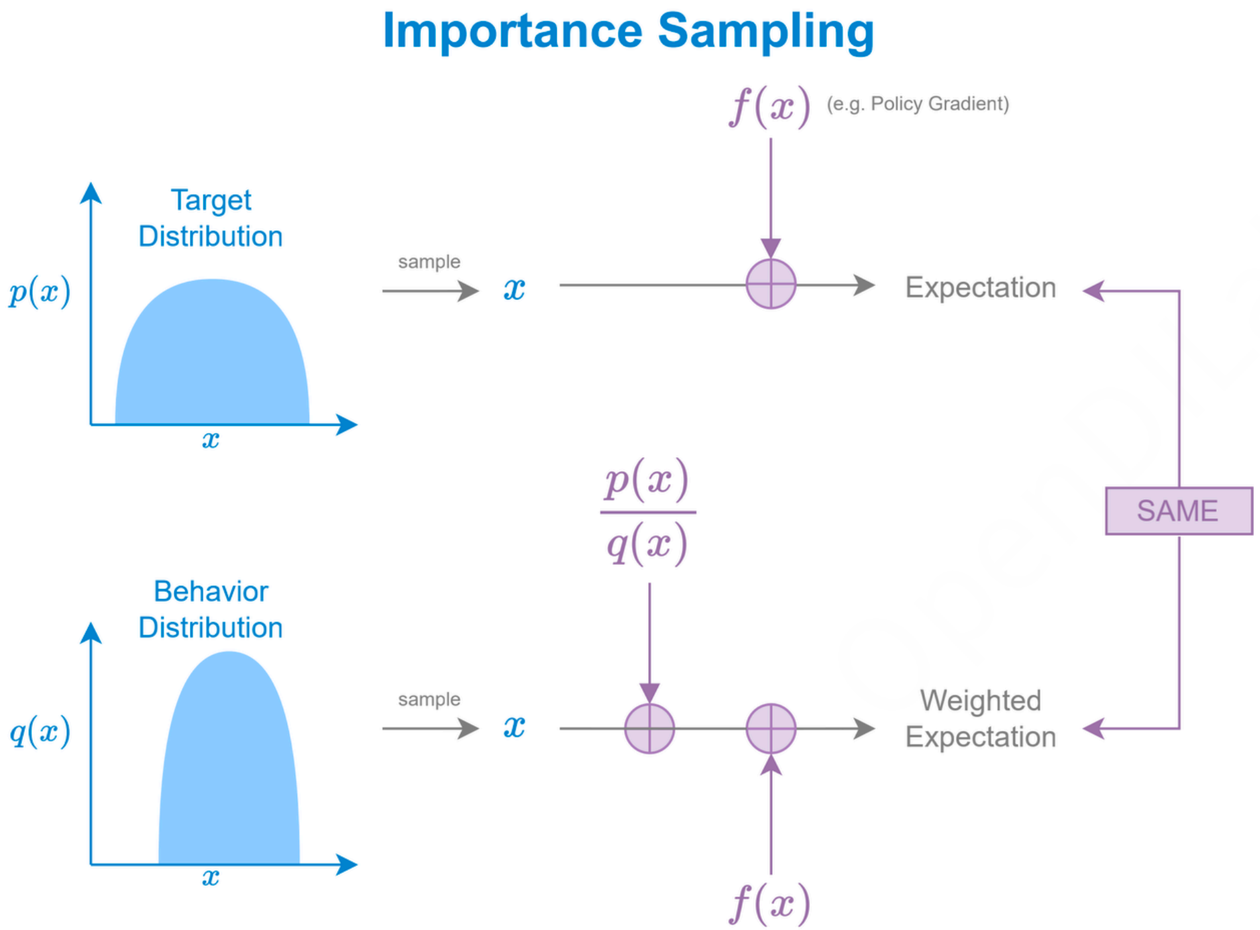
$$\eta(\tilde{\pi}) \geq L_\pi(\tilde{\pi}) - \frac{4\epsilon\gamma}{(1-\gamma)^2}\alpha \quad \text{（新策略和替代函数之间的定量关系）}$$

$$\text{where } \alpha = \max_s D_{\mathrm{KL}}(\pi(\cdot \mid s) \| \tilde{\pi}(\cdot \mid s)), \epsilon = \max_{s,a} \left| A_\pi(s,a) \right|$$

https://github.com/opendilab/PPOxFamily/tree/main/chapter1_overview/chapter1_supp_trpo.pdf

Trust Region Policy Optimization (TRPO): https://arxiv.org/pdf/1502.05477.pdf

TRPO：重要性采样



$$\eta(\tilde{\pi}) \approx L_{\pi}(\tilde{\pi}) = \eta(\pi) + \sum_{s} \rho_{\pi}(s) \sum_{a \sim \tilde{\pi}} \tilde{\pi}(a \mid s) A_{\pi}(s, a)$$

新策略的期望　　旧策略的优势函数

$$\eta(\tilde{\pi}) \approx L_{\pi}(\tilde{\pi}) = \eta(\pi) + \sum_{s} \rho_{\pi}(s) \sum_{a \sim \pi} IS \cdot \pi(a \mid s) A_{\pi}(s, a)$$

通过 IS 来进行矫正　　$IS = \frac{\tilde{\pi}(a \mid s)}{\pi(a \mid s)}$

https://github.com/opendilab/PPOxFamily/tree/main/chapter1_overview/chapter1_supp_trpo.pdf

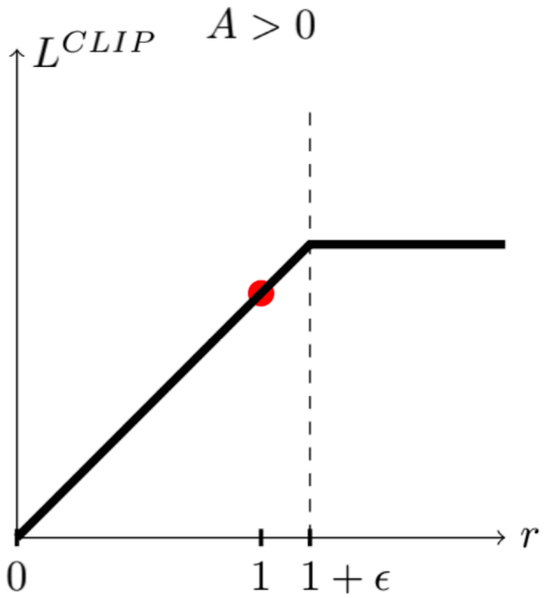
Trust Region Policy Optimization (TRPO): https://arxiv.org/pdf/1502.05477.pdf

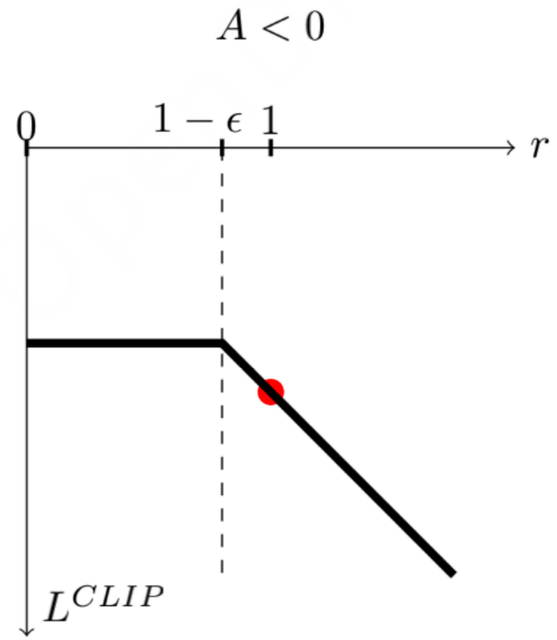
PPO：简洁+高效

问题：TRPO 求解约束优化问题很麻烦，IS 计算的方差很大

解决思路：Proximal Policy Optimization (PPO)

$$\mathbb{E}_t[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)}A^{\theta_k}(s_t,a_t)] \text{ + 约束条件} \Rightarrow \mathbb{E}_t[\min(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)}A^{\theta_k}(s_t,a_t), \text{clip}(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)}, 1-\epsilon, 1+\epsilon)A^{\theta_k}(s_t,a_t))]$$





- 截断式优化目标
- 悲观的约束（pessimistic bound）
- 结合其他改进方法（例如 GAE）

Proximal Policy Optimization (PPO): https://arxiv.org/pdf/1707.06347.pdf

PPO：简洁+高效

Algorithm 1 Proximal Policy Optimization (PPO)

Input: initialize policy (actor) parameters θ and value (critic) parameters ϕ. Training epochs per collect E, trajectory length T, batch size B.
for $k = 0, 1, 2, ...$ **do**
  Collect set of trajectories $\mathcal{D}_k = \{\tau_i\}$by running policy π_{θ_k} interaction with environment.
  Compute trajectory target return estimates $\hat{R}_t$. (e.g. n-step TD or other methods)
  Compute advantage estimates $\hat{A}^{\theta_k}$ with value V_{ϕ_k}. (e.g. GAE or other methods)
  for $e = 0, 1, ..., E-1$ **do**
    for minibatch:$b \in \mathcal{D}_k$ **do**
      Update the policy by maximizing the PPO-Clip objective with SGD:

$$L_\theta = \frac{1}{B \cdot T} \sum_{\tau \in \mathrm{b}} \sum_{t=0}^{T-1} \min(r(\theta)\hat{A}^{\theta_k}(s_t, a_t), \mathrm{clip}(r(\theta), 1-\epsilon, 1+\epsilon)\hat{A}^{\theta_k}(s_t, a_t))$$

$$r(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)}$$

      Fit the value by regression on mean-squared error with SGD:

$$L_\phi = \frac{1}{B \cdot T} \sum_{\tau \in \mathrm{b}} \sum_{t=0}^{T-1} (V_\phi(s_t) - \hat{R}_t)^2$$

    end for
  end for
end for

定性：一种 on-policy 强化学习算法

数据：收集序列轨迹，预计算所需值

两重训练循环：提高数据利用效率

策略更新：截断式优化目标+悲观约束

价值函数更新：结合多步时序差分方法

Proximal Policy Optimization (PPO): https://arxiv.org/pdf/1707.06347.pdf

下节预告

（二）离散与连续之辨：解构复杂动作空间

- 离散动作空间的概述
- 连续动作空间的介绍
- 混合动作空间的探索