# 多智能体强化学习实训
# Multi-agent Reinforcement Learning
## 腾讯“开悟”平台实践体验分享
## 基于强化学习的迷宫寻宝最优策略实践


**教师：张寅** zhangyin98@zju.edu.cn

**助教：邓悦** devindeng@zju.edu.cn

**王子瑞** ziseoiwong@zju.edu.cn

**李奕澄** yichengli@zju.edu.cn

浙江大学计算机学院

# 目录

# 题目内容概要

腾讯开悟

- 环境场景：2D迷宫（64*64网格），可配置起点、终点、宝箱点位
- 任务目标：训练智能体，让其在探索地图过程中学习路线规划，减少碰撞障碍物，以最少的步骤从起点走到终点，并收集宝箱
- 实现路径：
  - 经典强化学习（四种）：提供代码模版，填充算法关键细节
  - 深度强化学习（一种）：提供分布式RL框架，需要完成算法调研与接入
- 开发流程：

# 题目内容概要

腾讯开悟

- 经典强化学习算法
  - Dynamic Programming（固定起始点，无宝箱）
  - Monte-Carlo（固定起始点，无宝箱）
  - Q-learning（固定启始点，2个固定宝箱）
  - SARSA（固定启始点，2个固定宝箱）
- 深度强化学习算法&天梯排位
  - 有固定启始点，5个随机宝箱，10个固定点位

# 精彩回放

腾讯开悟

# 精彩回放

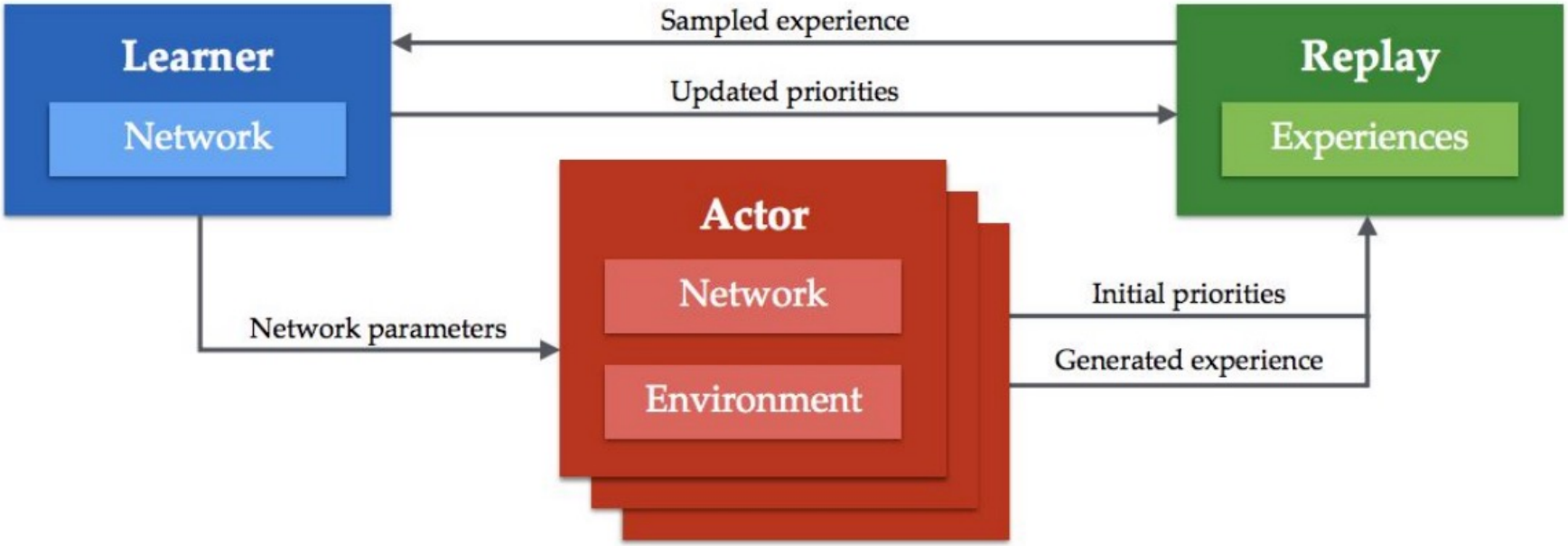
腾讯开悟

# 目录

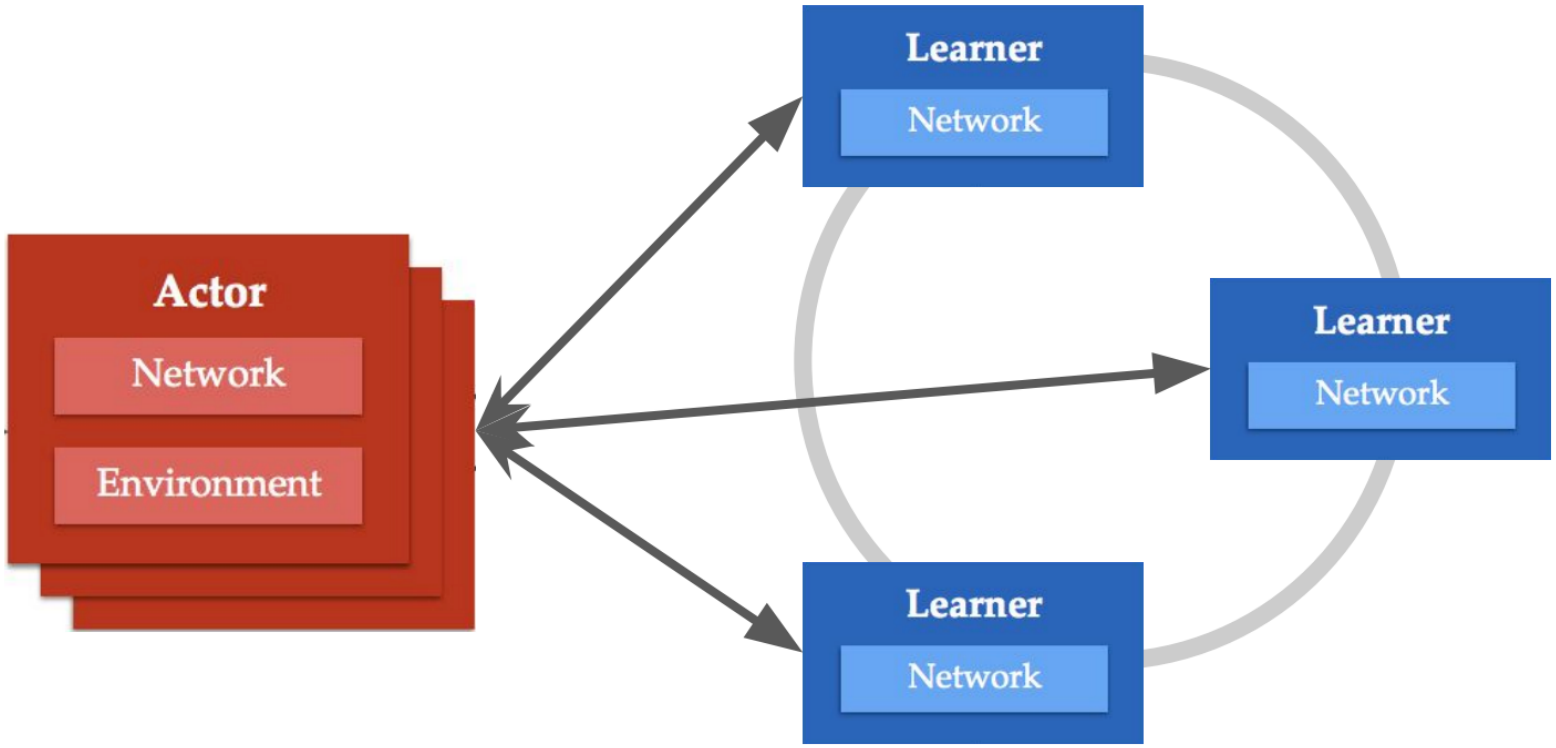
腾讯开悟

# 题目思路分析

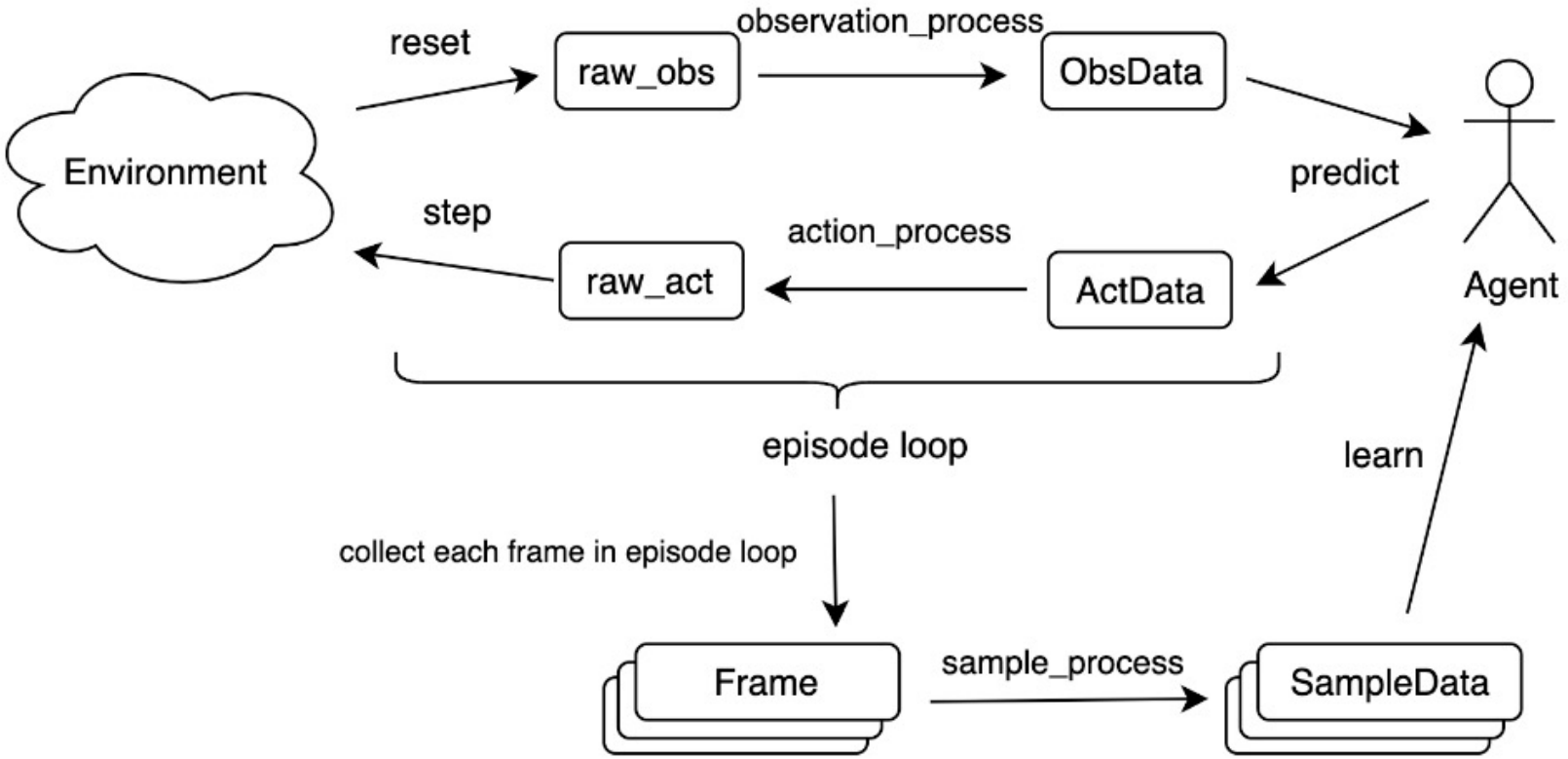
腾讯开悟


Learner
Network
Sampled experience
Updated priorities
Replay
Experiences
Actor
Network
Environment
Network parameters
Initial priorities
Generated experience

# 题目思路分析


Learner
Network
Actor
Network
Environment
Learner
Network
Learner
Network

# 题目思路分析


reset
observation_process
raw_obs
ObsData
Environment
predict
step
action_process
raw_act
ActData
Agent
episode loop
learn
collect each frame in episode loop
Frame
sample_process
SampleData

# 题目思路分析

腾讯开悟

| 数据名 | 数据类型 | 数据维度 | 数据描述 |
| --- | --- | --- | --- |
| state | int | [0] | 智能体当前位置状态，state = x * 64 + z |
| pos_row | list of int | [1: 65] | 智能体当前位置横坐标的 one-hot 编码 |
| pos_col | list of int | [65: 129] | 智能体当前位置纵坐标的 one-hot 编码 |
| end_dist | int | [129: 130] | 智能体当前位置相对于终点的离散化距离，0-6, 数字越大表示越远 |
| treasure_dist | list of int | [130: 140] | 智能体当前位置相对于宝箱的离散化距离，0-6, 数字越大表示越远 |
| obstacle_flat | list of int | [140: 165] | 智能体局部视野中障碍物的信息 (一维化)，1 表示障碍物, 0 表示可通行 |
| treasure_flat | list of int | [165: 190] | 智能体局部视野中宝箱的信息 (一维化)，1 表示宝箱, 0 表示没有宝箱 |
| end_flat | list of int | [190: 215] | 智能体局部视野中终点的信息 (一维化)，1 表示终点, 0 表示非终点 |
| memory_flat | list of int | [215: 240] | 智能体局部视野中的记忆信息 (一维化)，取值范围[0,1], 一个格子每走过一次, 记忆值 +0.1 |
| treasure_status | list of int | [240: 250] | 宝箱的状态，1 表示可以被收集, 0 表示不可被收集(未生成或者已经收集过), 长度为 10 |

# On-policy: Monte Carlo & SARSA

腾讯开悟

**视野域信息**

上面的所有信息都是**向量特征**，而视野域信息是**图特征**，包含障碍物图，终点图，和记忆图。

**视野域**是指以智能体所在位置为中心，分别向上下左右四个方向拓宽 VIEW (default = 2) 格数的一个正方形的**局部观察域** (defualt = 5 x 5)。

视野域中会标注出障碍物、宝箱、终点的位置：有则标注为 1，无则标注为 0，我们分别以障碍物和宝箱为例子

```
obstacle_map = [[1, 1, 1, 1, 1],
                [0, 0, 1, 1, 1],
                [0, 0, 0, 1, 1],
                [0, 0, 0, 0, 1],
                [1, 1, 1, 1, 1]]
```

obstacle_map矩阵的中心位置为智能体所在位置，1 代表有障碍物，0 代表无障碍物可以通行。

```
treasure_map = [[0, 0, 0, 1, 0],
                [0, 0, 0, 0, 0],
                [0, 0, 0, 0, 0],
                [0, 0, 0, 0, 0],
                [1, 0, 0, 0, 0]]
```

treasure_map矩阵的中心位置为智能体所在位置，1 代表有宝箱，0 代表无宝箱。

**记忆图**同样是一个 5 x 5 的局部观察域，值限制在 [0, 1]，初始化为 0。智能体每走到一个位置，该位置对应的值 +0.1，最大为 1，以此来表征智能体对历史探索过的信息的记忆。

# On-policy: Monte Carlo & SARSA

腾讯开悟

- Monte Carlo 奖励设置

```
# The reward for being close to the finish line
# 奖励3. 靠近终点的奖励:
reward -= 0.5 * obs[129]
```

| | Step | Score |
|---|---|---|
| MC (Ours) | 74 | 335 |
| MC-baseline | 84 | 333 |

| | | | |
|---|---|---|---|
| end_dist | int | [129: 130] | 智能体当前位置相对于终点的离散化距离，0-6, 数字越大表示越远 |

**Monte Carlo - Baseline**

**Monte Carlo (Ours)**

# On-policy: Monte Carlo & SARSA

腾讯开悟

- SARSA奖励设置

```
# The reward for being close to the finish line
# 奖励2. 靠近终点的奖励:
reward -=  obs[129] / 6
```

✓ 靠近终点奖励

```
# The reward for being close to the treasure chest
# 奖励4. 靠近宝箱的奖励(只考虑最近的那个宝箱)
min_dist = min(obs[130:140])
if min_dist > 6:
    reward=0
else:
    reward -= 0.5* min_dist / 6
```

✓ 靠近宝箱奖励

| | | | |
|---|---|---|---|
| end_dist | int | [129: 130] | 智能体当前位置相对于终点的离散化距离，0-6, 数字越大表示越远 |
| treasure_dist | list of int | [130: 140] | 智能体当前位置相对于宝箱的离散化距离，0-6, 数字越大表示越远 |

```
#奖励5. 重复路径的奖励
if not terminated:
    reward -= sum(obs[215:240])
```

✓ 重复路径惩罚

| | | | |
|---|---|---|---|
| obstacle_flat | list of int | [140: 165] | 智能体局部视野中障碍物的信息 (一维化)，1 表示障碍物, 0 表示可通行 |

# On-policy: Monte Carlo & SARSA

腾讯开悟

- SARSA奖励设置

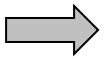
| | Step | Treasure | Score |
|---|---|---|---|
| SARSA (Ours) | **252** | **5/5** | **799** |
| SARSA-baseline | 543 | 4/5 | 640 |

**SARSA- Baseline**

**SARSA (Ours)**

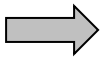
# Off-policy : Q-learning

```
# The reward for being close to the finish line
# 奖励2. 靠近终点的奖励:
if not terminated:
   reward += -1
reward +=  1.2*(6 - obs[129])/6 #obs[129]终点距离
```



- ✓ 步数限制奖励
- ✓ 靠近终点奖励

```
# The reward for being close to the treasure chest
# 奖励4. 靠近宝箱的奖励(只考虑最近的那个宝箱)
    min_dist = min(obs[130:140])
    reward  += 0.5*(6 - min_dist) / 6
```



定位最近宝箱，

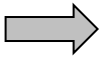
**根据与最近宝箱距离进行奖励**

与Monte-Carlo & SARSA方法的奖励设置有何不同？

# Off-policy : Q-learning

```
# ε-贪心算法用于动作选择
if np.random.rand() <= epsilon:
    action = np.random.randint(0, self.action_size)

 else:
    max_q = np.amax(self.Q[state, :], axis=-1)
    actions = np.where(self.Q[state, :] == max_q)[0]
    action = np.random.choice(actions)
```



- ✓ ε概率进行探索
- ✓ 1-ε概率进行利用

两种动作选择的方式有何不同？

```
# ε-贪心算法用于动作选择
if np.random.rand() <= epsilon:
    action = np.random.randint(0, self.action_size)

 else:
    action = np.argmax(self.Q[state, :], axis=-1)
```

# Off-policy : Q-learning

- **评测结果**

**Q-learning - Baseline**

**Q-learning（Ours）**

# 奖励调参心得

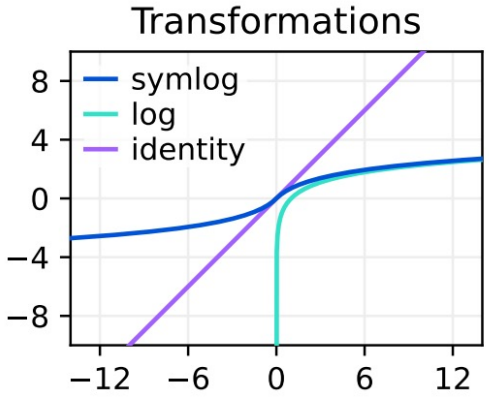
- 平衡RL算法的探索与利用，一般关键都在于激励探索
- On-policy RL会带有“自我模仿”（self-imitation）的倾向，即一旦探索到高于初始值的动作价值便会重复采用该动作。因此负奖励可以使其动作价值低于初始值从而激励探索
- Off-policy RL方法，尤其Q-learning只考虑最大化Q值的动作，可通过设置正奖励将智能体导向高价值区域
- 每项奖励最好都尽可能归一化，方便调整权重系数
- 相近任务需求的奖励需要保持单调性一致，避免混淆

# 目录

# 网络和观测设计

腾讯开悟

- Actor-Critic网络设计
  - 三层全连接结构 + ReLU激活函数
  - Actor采用Softmax计算动作概率分布
- 损失函数设计
  - **对称对数转换 (Symlog Transformation):**




$$\mathrm{symlog}(x) = \mathrm{sign}(x) \cdot \ln(|x| + 1) \tag{1-1}$$

$$\mathrm{symexp}(x) = \mathrm{sign}(x) \cdot (\exp(|x|) - 1) \tag{1-2}$$

  - **双热编码 (Twohot Encoding):**

$$\mathrm{twohot}(x)_i \doteq \begin{cases} |b_{k+1} - x|/|b_{k+1} - b_k| & \text{if } i = k \\ |b_k - x|/|b_{k+1} - b_k| & \text{if } i = k+1 \\ 0 & \text{else} \end{cases} k \doteq \sum_{j=1}^{B} \delta(b_j < x) \tag{2}$$

  - **优势**：更适用于奖励/价值值域范围未知的环境

# 网络和观测设计

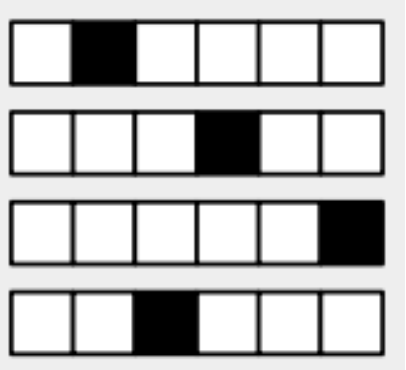
腾讯开悟

- 观测设计
  - 非线性关系处理

    **离散化**和One-hot编码将离散值转换为高维空间向量，提升模型对复杂特征的理解能力。

  - 数值稳定性

    One-hot编码将**输入特征**限制为离散的0和1之间，**避免由于输入数量级的差异带来的影响。**



视野域中会标注出障碍物、宝箱、终点的位置：有则标注为 1，无则标注为 0，我们分别以障碍物和宝箱为例子

```
obstacle_map = [[1, 1, 1, 1, 1],
                [0, 0, 1, 1, 1],
                [0, 0, 0, 1, 1],
                [0, 0, 0, 0, 1],
                [1, 1, 1, 1, 1]]
```

- 优势
  - One-hot编码处理带来了更加稀疏的表征与更丰富的语义。
  - 离散化和One-hot编码使模型在噪声和异常值面前更加鲁棒。

# PPO & Off-policy Actor Critic

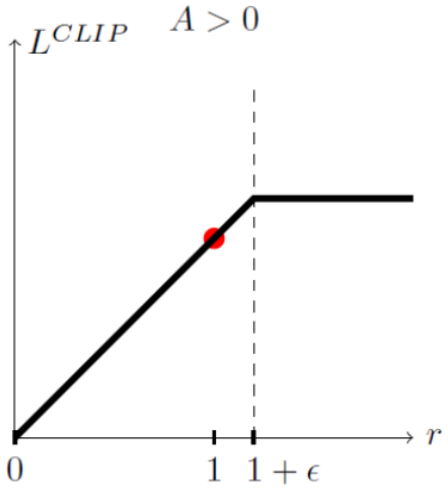
腾讯开悟

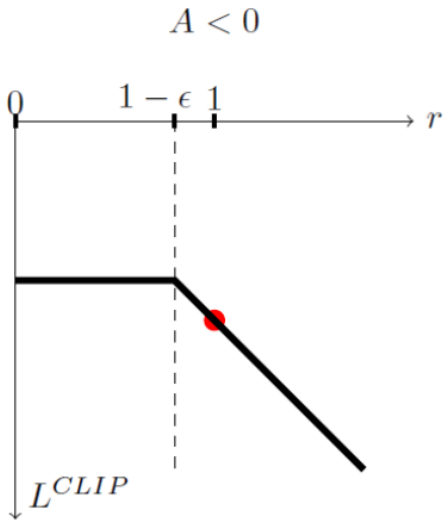
- 截断式优化目标

conservative policy iteration

$$L^{CPI}(\theta) = \hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}\hat{A}_t\right] = \hat{\mathbb{E}}_t\left[r_t(\theta)\hat{A}_t\right]$$

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t\left[\min\left(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\right)\right]$$




构建下界

$$L^{CLIP}(\theta) \leq L^{CPI}(\theta)$$

在$r = 1$附近相等

$$L^{CLIP}(\theta) = L^{CPI}(\theta)$$

# PPO & Off-policy Actor Critic

腾讯开悟

- 第一版方案
  - PPO + GAE + Reward shaping + 单线程采集

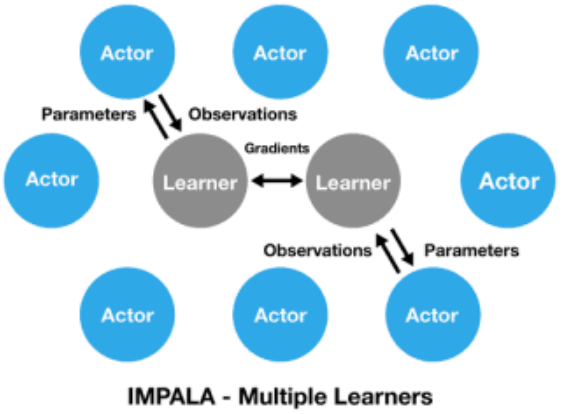
$$\text{GAE}(\gamma, 0): \quad \hat{A}_t := \delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$$

$$\text{GAE}(\gamma, 1): \quad \hat{A}_t := \sum_{l=0}^{\infty} \gamma^l \delta_{t+l} = \sum_{l=0}^{\infty} \gamma^l r_{t+l} - V(s_t)$$

- 第二版方案
  - Off-policy Actor Critic + Retrace + Reward shaping + 分布式训练

| | Definition of $c_s$ | Estimation variance | Guaranteed convergence† | Use full returns (near on-policy) |
|---|---|---|---|---|
| Importance sampling | $\frac{\pi(a_s\|x_s)}{\mu(a_s\|x_s)}$ | High | for any $\pi$, $\mu$ | yes |
| Q($\lambda$) | $\lambda$ | Low | only for $\pi$ close to $\mu$ | yes |
| TB($\lambda$) | $\lambda\pi(a_s\|x_s)$ | Low | for any $\pi$, $\mu$ | no |
| Retrace($\lambda$) | $\lambda \min(1, \frac{\pi(a_s\|x_s)}{\mu(a_s\|x_s)})$ | Low | for any $\pi$, $\mu$ | yes |

- 观测空间设计
  - 位置信息
  - 宝箱状态
- 奖励函数
  - 靠近宝箱的奖励
  - 重复进入记忆区域惩罚



IMPALA - Multiple Learners

# PPO & Off-policy Actor Critic

腾讯开悟

- 第一版方案
  - PPO + GAE + Reward shaping + 单线程采集

$$\text{GAE}(\gamma, 0): \quad \hat{A}_t := \delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$$

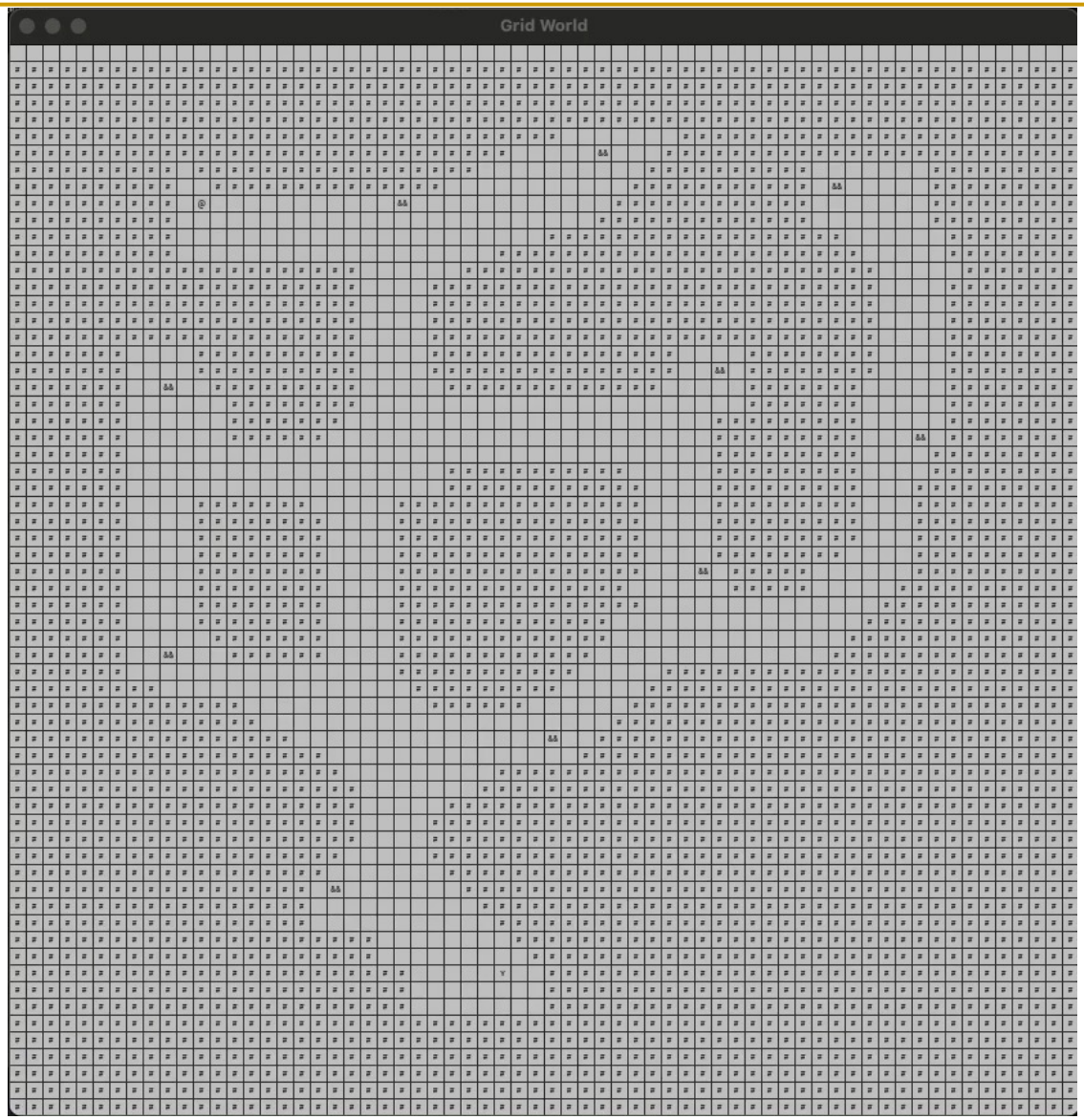
$$\text{GAE}(\gamma, 1): \quad \hat{A}_t := \sum_{l=0}^{\infty} \gamma^l \delta_{t+l} = \sum_{l=0}^{\infty} \gamma^l r_{t+l} - V(s_t)$$

- 第二版方案
  - Off-policy Actor Critic + Retrace + Reward shaping + 分布式训练

| | Definition of $c_s$ | Estimation variance | Guaranteed convergence† | Use full returns (near on-policy) |
|---|---|---|---|---|
| Importance sampling | $\frac{\pi(a_s\|x_s)}{\mu(a_s\|x_s)}$ | High | for any $\pi$, $\mu$ | yes |
| Q($\lambda$) | $\lambda$ | Low | only for $\pi$ close to $\mu$ | yes |
| TB($\lambda$) | $\lambda\pi(a_s\|x_s)$ | Low | for any $\pi$, $\mu$ | no |
| Retrace($\lambda$) | $\lambda \min(1, \frac{\pi(a_s\|x_s)}{\mu(a_s\|x_s)})$ | Low | for any $\pi$, $\mu$ | yes |

- 为何采用第二版方案？
  - Learner 与 Actor 不是存在更新版本差异，导致收集到的数据不是on-policy的，因此需要（1）使用Q函数代替V函数，以兼容一部分“off-policiness”（2）使用重要性采样进行分布修正

# 一些其他尝试



腾讯开悟

# 谢 谢