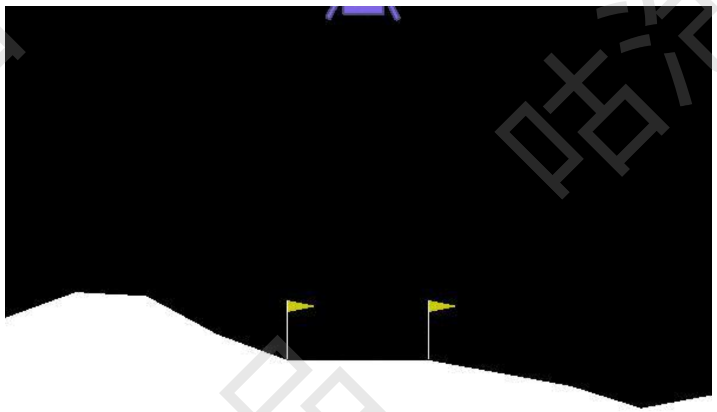
Proximal Policy Optimization

- 获得奖励



- 先来玩一个小游戏，虽然短，但是经历了好多过程：
- 飞船每一步行动都会获得不同的结果（奖励）
- 一个完整的过程，通常叫做**episod**，整个生命周期的奖励：$R = \sum_{t=1}^{T} r_t$

Proximal Policy Optimization

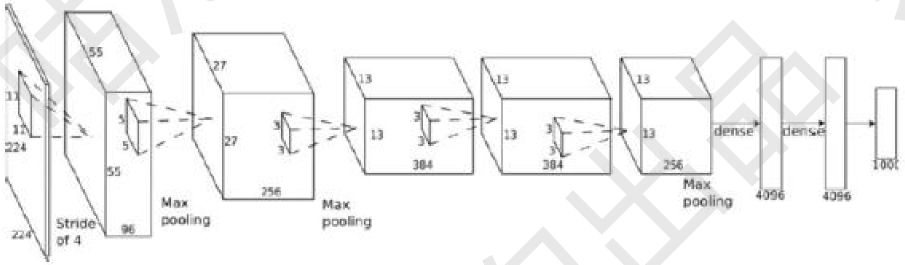
✓ 网络的输入与输出

一次游戏的记录结果：

包括了每一步的状态与行动（trajectory）：$\tau = \{s_1, a_1, s_2, a_2, \cdots, s_T, a_T\}$

每一步如何走才能得到更多的奖励呢？这就需要训练好神经网络了！

$\mathbf{s}_t$

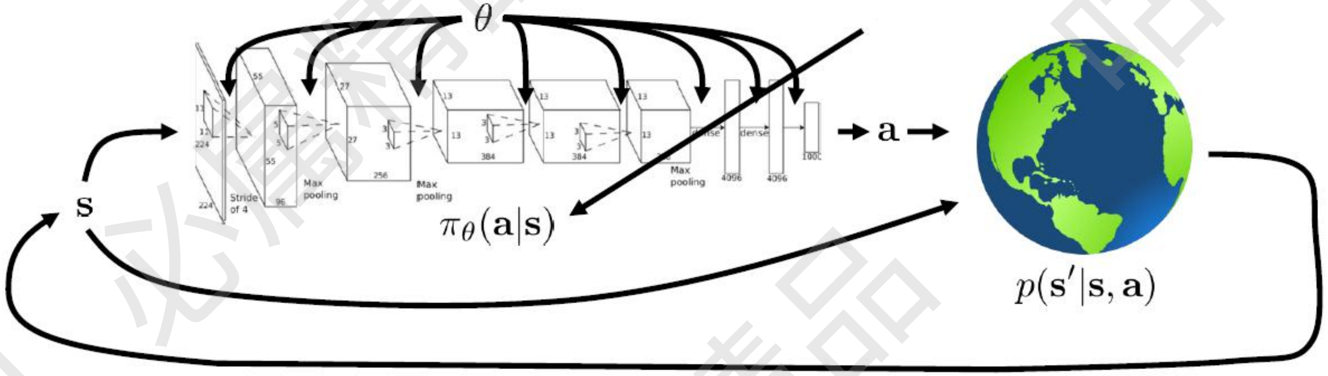


$\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)$

$\mathbf{a}_t$

Proximal Policy Optimization

- 游戏的行为是如何产生的呢?



- 其中 $p(s_{t+1}|s_t, a_t)$ 是游戏自带的

  $p_\theta(a_t|s_t)$ 是模型输出的结果

  其中奖励是由当前第一步的action与state共同决定的（规则，游戏提供）

- 当前模型恰好得到了如下的游戏记录:

$$\underbrace{p_\theta(\mathbf{s}_1, \mathbf{a}_1, \ldots, \mathbf{s}_T, \mathbf{a}_T)}_{p_\theta(\tau)} = p(\mathbf{s}_1) \prod_{t=1}^{T} \pi_\theta(\mathbf{a}_t|\mathbf{s}_t) p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$$

Proximal Policy Optimization

- 希望的目标

- 玩游戏不是目的，先有一个小目标：$\theta^\star = \arg\max_\theta E_{\tau \sim p_\theta(\tau)} \left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t) \right]$

- 其实就是要训练好模型，得到最多的奖励，但为什么是期望呢？

- 这一系列过程，带有太多随机性了，即便相同的θ得到的action也可能不同

Proximal Policy Optimization

希望的目标

目标函数如何来进行求解呢?

$$\theta^\star = \arg\max_\theta \underbrace{E_{\tau\sim p_\theta(\tau)}\left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t)\right]}_{J(\theta)}$$

按照大数定律的思想，直接穷举所有的可能性就好了:

$$J(\theta) = E_{\tau\sim p_\theta(\tau)}\left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t)\right] \approx \frac{1}{N}\sum_i\sum_t r(\mathbf{s}_{i,t}, \mathbf{a}_{i,t})$$

Proximal Policy Optimization

✓ 希望的目标

$\pi_\theta(\tau)$表示当前序列可能性，$r(\tau)$ 为奖励：$J(\theta)=E_{\tau\sim\pi_\theta(\tau)}[\underbrace{r(\tau)}_{\sum_{t=1}^{T} r(\mathbf{s}_t,\mathbf{a}_t)}] = \int \pi_\theta(\tau) r(\tau) d\tau$

计算梯度：

$$\nabla_\theta J(\theta) = \int \nabla_\theta \pi_\theta(\tau) r(\tau) d\tau = \int \pi_\theta(\tau) \nabla_\theta \log \pi_\theta(\tau) r(\tau) d\tau = E_{\tau\sim\pi_\theta(\tau)}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)]$$

等会！这是扯啥子东西呢。。。继续来看一下它的推导过程

Proximal Policy Optimization

- 唠十块钱数学

- 公式：$$\frac{d\log(f(x))}{dx} = \frac{1}{f(x)}\frac{df(x)}{dx}$$

$$\nabla_\theta J(\theta) = \int \nabla_\theta \pi_\theta(\tau) r(\tau) d\tau = \int \pi_\theta(\tau) \nabla_\theta \log \pi_\theta(\tau) r(\tau) d\tau = E_{\tau\sim\pi_\theta(\tau)}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)]$$

- 稍微转换一下就好求解了：

$$\pi_\theta(\tau)\nabla_\theta \log \pi_\theta(\tau) = \pi_\theta(\tau)\frac{\nabla_\theta \pi_\theta(\tau)}{\pi_\theta(\tau)} = \nabla_\theta \pi_\theta(\tau)$$

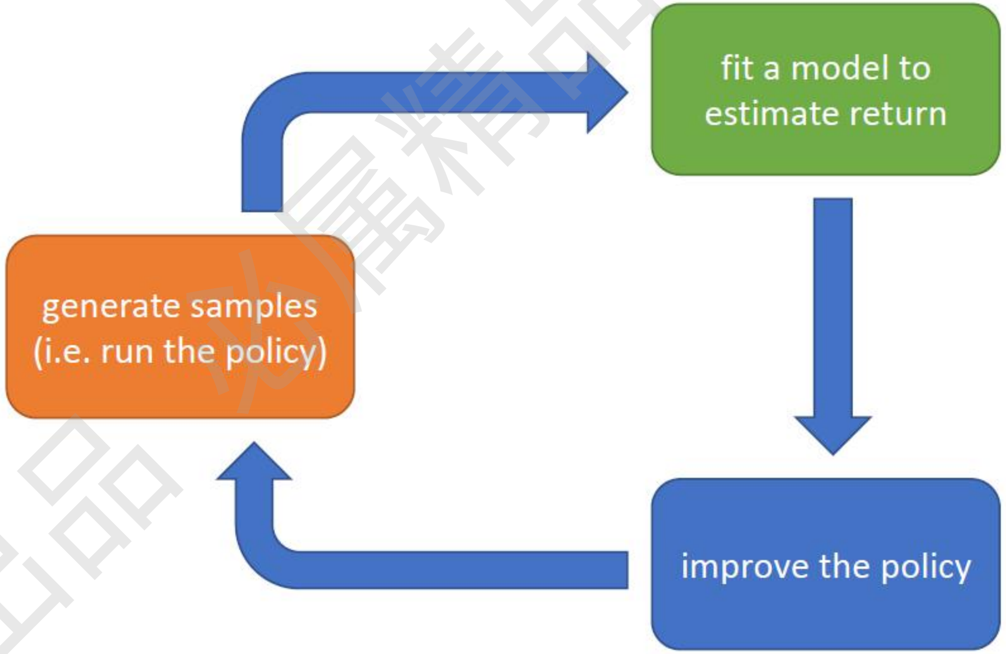
Proximal Policy Optimization

- 最终要求解的梯度

终极版： $\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{t=1}^{T}\nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t}|\mathbf{s}_{i,t})\right)\left(\sum_{t=1}^{T} r(\mathbf{s}_{i,t},\mathbf{a}_{i,t})\right)$

更新参数： $\theta \leftarrow \theta + \alpha\nabla_\theta J(\theta)$ （梯度上升）

就是这个铁三角关系，来训练网络模型：



Proximal Policy Optimization

- 如何获得这么多游戏记录呢?

  - 一直玩就好了！第一场，第二场，第三场。。。

  - 每场游戏记录好各自情况：
    τ^1: (s_1^1, a_1^1) (s_2^1, a_2^1)
    τ^2: (s_1^2, a_1^2) (s_2^2, a_2^2)
    及奖励：$R(\tau^1)$ $R(\tau^2)$

  - 把数据全部带入求解即可：

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \left(\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t} | \mathbf{s}_{i,t}) \right) \left(\sum_{t=1}^{T} r(\mathbf{s}_{i,t}, \mathbf{a}_{i,t}) \right)$$

Proximal Policy Optimization

- baseline

总的奖励（一场游戏总共获得）看起来就像一个权重项：

$$\nabla_\theta J(\theta) \approx \underline{\frac{1}{N}\sum_{i=1}^{N}}\left(\sum_{t=1}^{T}\nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t}|\mathbf{s}_{i,t})\right)\underline{\left(\sum_{t=1}^{T} r(\mathbf{s}_{i,t},\mathbf{a}_{i,t})\right)}$$

我们希望通过奖励和惩罚来完成训练，但是有些游戏可能只有奖励，这回可以对总的奖励来一个去均值操作！

$$R(\tau^n) - b \qquad\qquad b = \frac{1}{N}\sum_{i=1}^{N} r(\tau)$$

Proximal Policy Optimization

- On policy 与 Off policy

  - On policy： 就是训练数据由当前agent与不断环境交互得到的(勤工俭学)

  - Off policy：就是自己可以歇着了，找个打工的帮我跟环境交互得到结果

  - 刚才算出来那兄弟是哪一个呢？为什么需要Off policy?

$$\nabla_\theta J(\theta) = \underbrace{E_{\tau\sim\pi_\theta(\tau)}}_{\text{这好像有麻烦了}}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)]$$

Proximal Policy Optimization

- On policy 与 Off policy

  - 如果使用On policy策略训练起来就太慢了，做一批数据，迭代一次。。。

  - 会看到大部分时间都在等着这个agent在玩，它倒是玩开心了，网络没咋变。。

  - 能不能给我找个打工的，让他去玩，把得到的数据给我就好了呢?
    这其实就是Off policy的思想了，先找一个 $\bar{\pi}(\tau)$ 去替代 $\pi_\theta(\tau)$

Proximal Policy Optimization

Importance Sampling

准备玩一个狸猫换太子，梯度策略中需要我们不断的产生样本数据：

$$E_{x\sim P}[f(x)] = \int_x P(x)f(x)d_x \approx \frac{1}{N}\sum_{x_i\sim P, i=1}^{N} f(x_i)$$

从P这个分布中不断采样X，在把X带入到f(x)中，再求f(x)的期望值
狸猫P '准备上场，来把太子P换下来：

利用公式：$\int f(x)p(x)dx = \int f(x)\frac{p(x)}{q(x)}q(x)dx$ ，其中 $\frac{p(x)}{q(x)}$ 可当做权重项

$$E_{x\sim P}[f(x)] = \int_x P(x)f(x)d_x = E_{x\sim P'}[\frac{P}{P'}f(x)] \approx \frac{1}{N}\sum_{x_i\sim P', i=1}^{N} \frac{P(x)}{P(x)'}f(x)$$

Proximal Policy Optimization

- Importance Sampling

  - 第一个事，咱这个狸猫得长得差不多点（P与P '要尽可能接近）

  - 从P '中sample出数据供θ来进行训练（这一批数据可以训练好多次）

  - 用狸猫上场： $\nabla_{\theta'} J(\theta') = E_{\tau \sim \pi_\theta(\tau)} \left[\frac{\pi_{\theta'}(\tau)}{\pi_\theta(\tau)} \nabla_{\theta'} \log \pi_{\theta'}(\tau) r(\tau) \right]$

Proximal Policy Optimization

- 条件限制

- 狸猫和太子要相近（KL-divergence）： $D_{\mathrm{KL}}(\pi_{\theta'}\|\pi_\theta) = E_{\pi_{\theta'}}[\log \pi_\theta - \log \pi_{\theta'}]$

- 直观解释：$\|\theta' - \theta\|^2 \leq \epsilon$ ，实际中是它俩经过网络的预测结果尽可能差不多

- 这个狸猫去哪找呢？大象的左耳朵最像什么？直接拿要训练模型的前一次迭代时的参数不就可以了嘛！

$$clip\left(\frac{p_\theta(a_t|s_t)}{p_{\theta^k}(a_t|s_t)}, 1-\varepsilon, 1+\varepsilon\right)$$ （PPO2版本中的限制条件）

Proximal Policy Optimization

- 案例解读：ActorCritic组合

Critic的作用：（对捐款这件事，好比当年的葫芦娃，海尔兄弟，哪吒）

通俗解释就是让模型知道现在这个水平该干啥
（级别越高也得打越高级别的怪，这样才能收益更大！）

还记得最开始咱们的约定嘛：$\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{t=1}^{T}\nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t}|\mathbf{s}_{i,t})\right)\left(\sum_{t=1}^{T} r(\mathbf{s}_{i,t},\mathbf{a}_{i,t})\right)$

其中的b就是Critic网络要学习的结果：$R(\tau^n) - b$

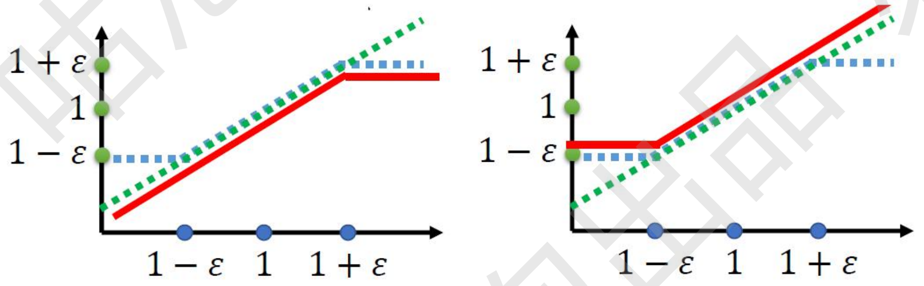
Proximal Policy Optimization

案例解读：PPO2版本

- advantages：$\underline{\left(\sum_{t=1}^{T} r(\mathbf{s}_{i,t}, \mathbf{a}_{i,t})\right)}$ 也就是现在的 $R(\tau^n) - b$ 即 $A^{\theta^k}(s_t, a_t)$

- 限制好范围，例如ε取0.2；A>0是好事，此时 $p_\theta(a_t|s_t)$要越大越好，但有上界！

$$J_{PPO2}^{\theta^k}(\theta) \approx \sum_{(s_t,a_t)} min\left(\frac{p_\theta(a_t|s_t)}{p_{\theta^k}(a_t|s_t)} A^{\theta^k}(s_t,a_t),\ \ clip\left(\frac{p_\theta(a_t|s_t)}{p_{\theta^k}(a_t|s_t)}, 1-\varepsilon, 1+\varepsilon\right) A^{\theta^k}(s_t,a_t)\right)$$



当A>0时　　当A<0时