

策略梯度.

Monte Carlo 策略梯度.

直接或间接方法

state  $\longrightarrow$  action

不使用 Value function 选择动作

优化  $\pi$  参数.

## 1. Why using Policy-based methods?

①  $\pi$  两种  $\swarrow$

|               |                                   |
|---------------|-----------------------------------|
| deterministic | $\pi(s) = a$                      |
| stochastic    | $\pi(a s) = P(a_t s_t) \geq 70\%$ |

action  $a$  概率.

$\downarrow$  环境不确定

POMDP - partially observable Markov

Decision  
Process

## ② Advantage

### 1) Convergence

Value-based 方法

评估 action value 政策.

2).

action-space 高维且时变高维.

action 是连续的.

PG — 调整是 param — 得到 max-value action  
而不是计算值.

3) 学习随机场.

不带  $\epsilon$ -greed

perceptual aliasing 伪知觉.

两个相同的状态. 不同动作

③ Disadvantage-

local maximum 局部极值.

2. Policy Search

$$\pi_\theta(a|s) = P(a|s)$$

↓  
优化  $J(\theta)$  & max a score function  $J(\theta)$

$$J(\theta) = E_{\pi_\theta} (\sum r)$$

⎧ A.  $J(\theta)$  表示  $\pi_\theta$  的值  
 ⎩ B. Gradient Ascent 对  $\theta$  提升  $\pi$ .

步骤：

Step1:  $J(\theta)$  the expected reward of  $\pi$

3种 optimizing policies 从 env. objective 选择.

A. episodic env

从 Start Value

$$J_1(\theta) = E_{\pi} [G_1 = R_1 + \gamma R_2 + \gamma^2 R_3 + \dots] = E_{\pi} (\underline{V(S_1)})$$

the Value of  $S_1$

调整  $\theta$ , 改变 action 方向  $\rightarrow$  Step2.

B. continuous Env

没有固定的开始状态. the average value.

$$J_{avg}(\theta) = E_{\pi} (V(s)) = \sum \frac{N(s)}{\sum' N(s')} V(s)$$

权重  $\frac{s \text{ 数量}}{\text{所有 } s \text{ 数量}}$   $d(s)$

C. 每步平均奖励。 (每步都想获得最多的奖励)

$$J_{avR}(\theta) = E_{\pi}(\Gamma)$$

$$= \sum_s d(s) \sum_a \pi_{\theta}(s, a) R_s^a$$

状态分布      action概率      reward值

Step2: Policy Gradient ascent 方

$\text{Max } J(\theta)$  即找到最优策略。

Gradient 最陡峭的方向。

Loss function — Min(L-function) GD

the score function  $J(\theta) = \text{Max}(J(\theta))$  GA

Policy:  $\pi_{\theta}$

Objective function:  $J(\theta)$

Gradient:  $\nabla_{\theta} J(\theta)$

Update:  $\theta \leftarrow \theta + \alpha \nabla_{\theta} J(\theta)$

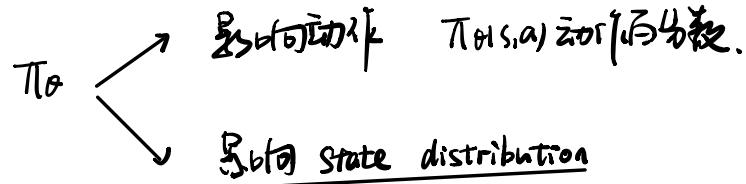
$$\theta^* = \underset{\theta}{\operatorname{argmax}} \underbrace{E_{\pi_{\theta}} \left[ \sum_t R(s_t, a_t) \right]}_{J(\theta)} \quad \begin{matrix} s_0, a_0, r_0 \\ \dots \dots \\ s_t, a_t, r_t \end{matrix}$$

回报的期望。  
Policy 期望

$$J_1(\theta) = V_{\pi_{\theta}}(s_1) = E_{\pi_{\theta}}(V_1)$$

$$= \frac{\sum_{s \in S} d(s) \sum_{a \in A} \pi_\theta(s, a) R_s^a}{\text{State} \uparrow \text{action} \uparrow}$$

如何改变  $\pi_\theta$  确保  $J(\theta)$  提高.



问题：当  $\nabla_\theta J(\theta)$  取决于  $\pi_\theta$  和 State Distribution 未知的情况下。  
如何估计策略参数的梯度。 $\nabla_\theta J(\theta)$

### Policy Gradient Theorem

$\nabla_\theta J(\theta)$  表达不涉及 State distribution

$$J(\theta) = E_{\pi_\theta}[R(z)] \quad z: s_0, a_0, r_0, s_1, a_1, r_1, \dots$$

$$\nabla_\theta J(\theta) = \nabla_\theta \sum_z \pi(z; \theta) R(z)$$

$$= \sum_z \boxed{\nabla_\theta \pi(z; \theta)} R(z)$$

$\pi_\theta$  Stochastic Policy

$\pi(z; \theta)$  概率分布.

$(s_0, a_0, r_0, s_1, a_1, r_1, \dots)$  为样本，参数  $\theta$ .

概率分布对数似然  $\rightarrow$  logarithm 对数.

$$\text{似然函数} \quad \pi(z; \theta) \frac{\nabla_\theta \pi(z; \theta)}{\pi(z; \theta)}$$

$$\nabla \log x = \frac{\nabla x}{x}$$

$$\rightarrow = \sum_z \pi(z; \theta) \nabla_{\theta} (\log \pi(z; \theta)) R(z)$$

$$\nabla_{\theta} J(\theta) = E_z [\nabla_{\theta} (\log \pi(z; \theta)) \underline{R(z)}]$$

Policy Function      Score function

$$\text{Update : } \alpha * \nabla_{\theta} (\log \pi(s, a; \theta)) R(z) = \Delta \theta$$

### 3. Monte Carlo Policy Gradients

从多条路径形成一个 episodes.

Initialize  $\theta$

for each episode  $T = s_0, a_0, r_1, s_1, \dots, s_T$ ;

for  $t$  in  $[1, T-1]$

$$\Delta \theta = \alpha \nabla_{\theta} (\log \pi(s_t, a_t; \theta)) \underline{G_t}$$

$$\theta = \theta + \Delta \theta$$

$\downarrow$   
if  $R-T$  episode.

从多条路径. 那么有环境和(及)不同 actions.

提高方法: A. Actor - Critic

B. Proximal Policy Gradients

#### 4. Cart Pole

不计算 Q-value 如何评价π好坏？

看整个回合的得失情况 → 前得分高方向优化

##### ① 累积回报

$$\text{discount-reward}[i] = \text{reward}[i] + \gamma * \text{discount-reward}[i+1]$$

out - np.mean(out) 中心化

np.std 标准化 标准差  $\sqrt{\frac{\sum(x_i - \bar{x})^2}{n}}$

##### ② loss 权重

rewards 高，action 概率大  $\pi_\theta$ .

越大 loss 值

神经网络朝 rewards 快速梯度方向优化

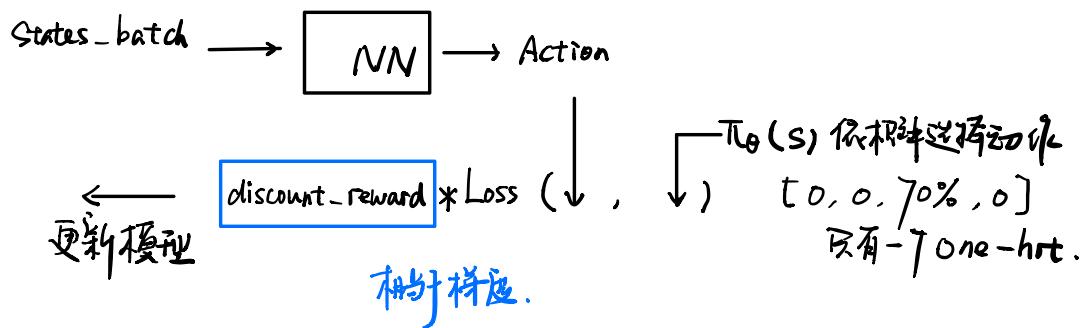
$$\text{loss} = \frac{\text{discount-reward} * \text{loss}}{\text{梯度.}}$$

model.fit (s-batch

p rob-batch

sample\_weight = r-batch

一个完整 episode



model.fit ( sample\_weight = 累积 reward

调整损失函数.

对样本一对一加权. 累积 reward 越高 ↑  
loss 越大.

朝累积期望奖励方向优化.