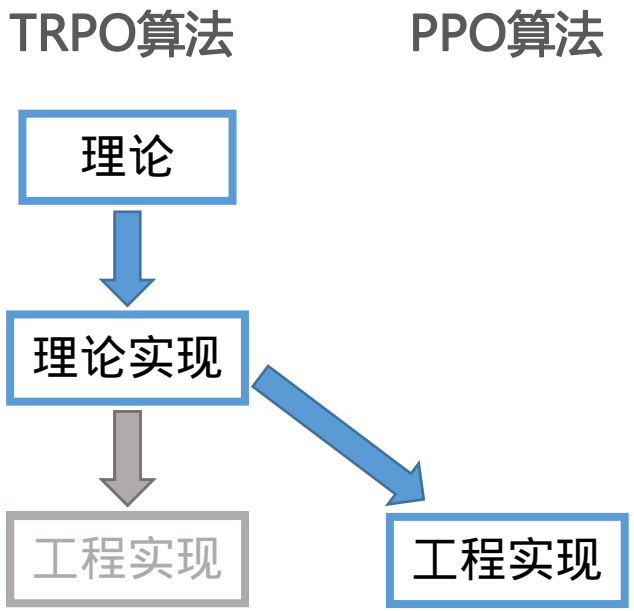
彪哥带你学强化学习

17.深入理解PPO算法

DEEPLY UNDERSTAND REINFORCEMENT LEARNING

讲师：韩路彪

PPO算法：理论基础同TRPO算法，训练过程上的优化



TRPO算法的回顾

TRPO优化目标

$$\begin{aligned} \underset{\theta}{\text{maximize}} \quad & \hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}\hat{A}_t\right] \\ \text{subject to} \quad & \hat{\mathbb{E}}_t[\mathrm{KL}[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)]] \leq \delta. \end{aligned}$$

无约束化

$$\underset{\theta}{\text{maximize}}\, \hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}\hat{A}_t - \beta\,\mathrm{KL}[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)]\right]$$

共轭梯度法求解

问题

- 计算复杂（依赖二阶优化）
- 无法用于噪声模型（如dropout）
- 无法用于参数共享模型

PPO算法

核心思想：限制新旧策略变化不要太大

思路一：自适应KL惩罚系数

使用梯度下降法优化目标函数

$$\underset{\theta}{\text{maximize}}\, \hat{\mathbb{E}}_t \left[\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)} \hat{A}_t - \beta \mathrm{KL}[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)] \right]$$

自适应调整惩罚系数β

$$\text{Compute } d = \hat{\mathbb{E}}_t[\mathrm{KL}[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)]]$$

- If $d < d_{\text{targ}}/1.5$, $\beta \leftarrow \beta/2$
- If $d > d_{\text{targ}} \times 1.5$, $\beta \leftarrow \beta \times 2$

d_{targ}: 超参数

效果不佳

PPO算法

思路二：裁切代理目标

定义为概率比 $r_t(\theta)$

原TRPO优化目标

$$\underset{\theta}{\text{maximize}} \quad \hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}\hat{A}_t\right]$$

$$\text{subject to} \quad \hat{\mathbb{E}}_t[\mathrm{KL}[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)]] \leq \delta.$$

PPO优化目标函数

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

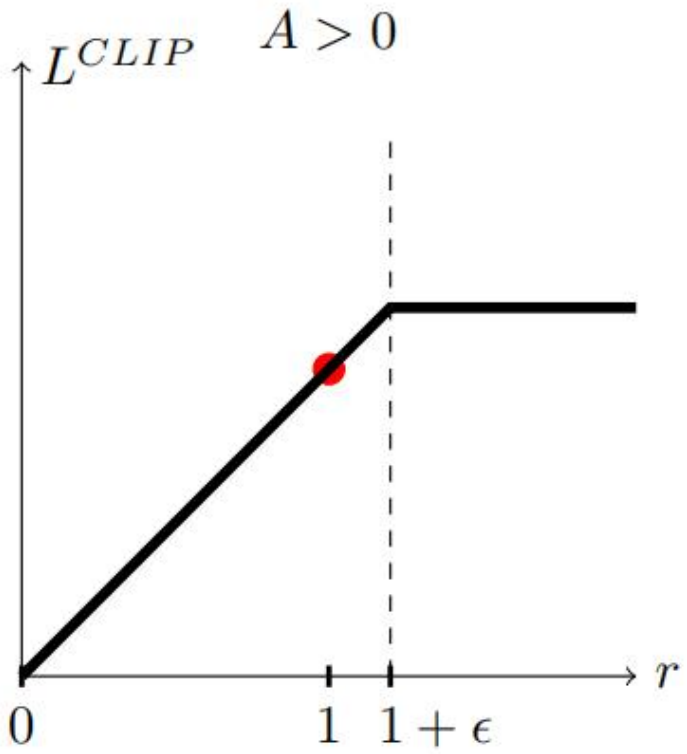
PPO算法

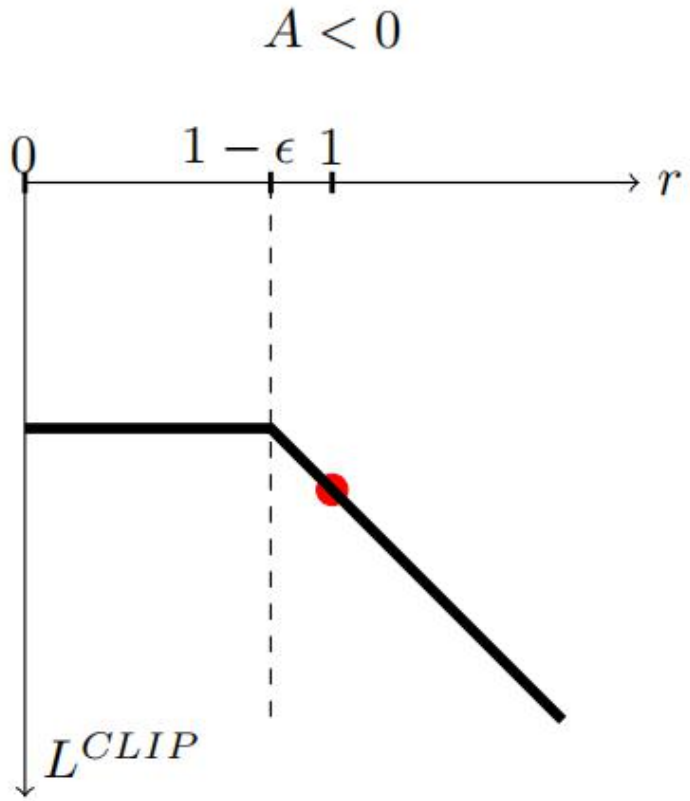
论文测试结果

algorithm	avg. normalized score
No clipping or penalty	-0.39
Clipping, $\epsilon = 0.1$	0.76
Clipping, $\epsilon = 0.2$	**0.82**
Clipping, $\epsilon = 0.3$	0.70
Adaptive KL $d_{\text{targ}} = 0.003$	0.68
Adaptive KL $d_{\text{targ}} = 0.01$	0.74
Adaptive KL $d_{\text{targ}} = 0.03$	0.71
Fixed KL, $\beta = 0.3$	0.62
Fixed KL, $\beta = 1.$	0.71
Fixed KL, $\beta = 3.$	0.72
Fixed KL, $\beta = 10.$	0.69

PPO算法

clip方法的裁切图





PPO算法

clip方法举例

π(a|s)	0.6
ε	0.2

				A	2			A	-2	
πnew(a|s)	r	r_clip		rA	r_clipA	Lclip		rA	r_clipA	Lclip
0.1	0.17	0.8		0.33	1.6	**0.33**		-0.33	-1.6	**-1.6**
0.2	0.33	0.8		0.67	1.6	**0.67**		-0.67	-1.6	**-1.6**
0.3	0.50	0.8		1	1.6	**1**		-1	-1.6	**-1.6**
0.4	0.67	0.8		1.33	1.6	**1.33**		-1.33	-1.6	**-1.6**
0.5	0.83	0.833		1.67	1.6667	**1.67**		-1.67	-1.67	**-1.67**
0.6	1.00	1		2	2	**2**		-2	-2	**-2**
0.7	1.17	1.167		2.33	2.3333	**2.33**		-2.33	-2.33	**-2.33**
0.8	1.33	1.2		2.67	2.4	**2.4**		-2.67	-2.4	**-2.67**
0.9	1.50	1.2		3	2.4	**2.4**		-3	-2.4	**-3**

PPO算法

clip方法举例

π(a | s)	0.6
ε	0.2

				A	2			A	-2	
πnew(a | s)	r	r_clip		rA	r_clipA	Lclip		rA	r_clipA	Lclip
0.1	0.17	0.8		0.33	1.6	**0.33**		-0.33	-1.6	**-1.6**
0.2	0.33	0.8		0.67	1.6	**0.67**		-0.67	-1.6	**-1.6**
0.3	0.50	0.8		1	1.6	**1**		-1	-1.6	**-1.6**
0.4	0.67	0.8		1.33	1.6	**1.33**		-1.33	-1.6	**-1.6**
0.5	0.83	0.833		1.67	1.6667	**1.67**		-1.67	-1.67	**-1.67**
0.6	1.00	1		2	2	**2**		-2	-2	**-2**
0.7	1.17	1.167		2.33	2.3333	**2.33**		-2.33	-2.33	**-2.33**
0.8	1.33	1.2		2.67	2.4	**2.4**		-2.67	-2.4	**-2.67**
0.9	1.50	1.2		3	2.4	**2.4**		-3	-2.4	**-3**

PPO算法

一个隐藏问题：新策略$\pi_{new}(a|s)$在更新策略前怎么计算出来的？

鸡生蛋？蛋生鸡？

解决方法：一批数据（比如一条轨迹）多轮更新

一批数据更新的过程：

1. 采样一批数据
2. 计算样本里边的所有(s,a)对的概率$\pi_{old}(a|s)$
3. 做多轮更新（一般3到10轮）
   - ① 用现在的参数计算样本里边的所有(s,a)对的概率$\pi_{new}(a|s)$
   - ② 用$\pi_{old}(a|s)$(每次不变)和$\pi_{new}(a|s)$计算比例r，并计算损失函数-L
   - ③ 梯度下降更新θ

实际计算r(θ)时，常使用对数概率 log π(a|s) 避免数值问题：

r(θ) = exp(logπ_new(a|s) - logπ_old(a|s)))