前言

Policy gradient是解决reinforcement learning问题的另一种思路，在上一个lecture中，我们采用的是value based的方法，我们优化的也是一个action-value function，实际上有了function approximation的技术，我们可以直接优化一个带参数的策略π_θ，基于对标量性能衡量$J(\theta)$的梯度计算。

介绍

考虑model-free的条件下，来优化：

$$\pi_\theta(s,a) = P[a \mid s, \theta]$$

这是**policy based**的方法；如果还是需要value function，那就是混合的**actor-critic**方法。但是请注意，这两者和value based的本质区别是，在动作选择的时候不会再”**咨询**“value function了。

Advantages of Policy-Based RL

- 更好的收敛性质：参数化带来的随机性使得在一些特殊问题上可以避免振荡
- 在高维或者连续动作空间中很有效：避免了很难处理的$\arg\max_a q_\pi(s,a)$
- 可以学习到随机策略
- 可以引入有关策略形式的先验知识

Disadvantages of Policy-Based RL

- 一般收敛到一个局部最优策略（这里soft-max可以解决嘛？）
- 评估一个策略一般来说很低效且方差大

Policy Objective Functions

- episodic environment：$J(\theta) = E_{S_0}[V^{\pi_\theta}(S_0)]$,在这种情况下一般$\gamma = 1$
- continuing environment:

$$J_{avV}(\theta) = \sum_s d^{\pi_\theta}(s) V^{\pi_\theta}(s)$$

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

其中d^{π_θ}指Markov chain的stationary distribution（又称on-policy distribution），且$J_{avV}(\theta) = \frac{1}{1-\gamma} J_{avR}(\theta)$,所以两个目标式实际上是等价的。

我们的目标是找到最优θ，使得$J(\theta)$最大

Monte-Carlo Policy Gradient

Score Function
出于好计算一些$\nabla_\theta \pi_\theta$的目的，转换了一下形式，定义了一个score function

$$\nabla_\theta log\pi_\theta$$

显然有$\nabla_\theta \pi_\theta = \pi_\theta * \nabla_\theta log\pi_\theta$.对于离散动作空间常用**Softmax Policy**，在连续的动作空间则常用**Gaussian Policy**

Policy Gradient Theorem
有关这个定理的详细推导可以参阅**An Introduction to Reinforcement Learning**，这个定理通过具体的数学推导向我们展示了$\nabla J(\theta)$的形式其实可以很简洁，

$$\nabla J(\theta) \propto \sum_s \mu(s) \sum_a q_{\pi_\theta}(s,a) \nabla \pi_\theta(a \mid s) = \mathbb{E}_{\pi_\theta}[q_{\pi_\theta}(s,a) \nabla_\theta log\pi_\theta(s,a)]$$

其中$\mu(s)$是on-policy distribution，在episodic和continuing环境下二者定义略有区别，除此之外以上关系适用于$J(\theta), J_{avV}(\theta), J_{avR}(\theta)$

Monte-Carlo Policy Gradient(REINFORCE)
采用Stochastic gradient descent，对于梯度采样，也就是说每次更新参数我们只需要$q_{\pi_\theta}(S_t, A_t) \nabla_\theta log\pi_\theta(S_t, A_t)$,这种方法适用于episodic环境，因为我们需要用G_t来替代$q_{\pi_\theta}(S_t, A_t)$，这就是REINFORCE的核心。

REINFORCE with Baseline
Baseline的引入可以减少G_t方差过大的问题，从而使得训练可以更快的收敛。一个很自然的想法是用$\hat{v}(s, \vec{w})$来作为baseline，所以算法的改变体现在如下：

$$\delta \leftarrow G_t - \hat{v}(S_t, \vec{w})$$

$$\vec{w} \leftarrow \vec{w} - \alpha^{\vec{w}} \delta \nabla \hat{v}(S_t, \vec{w})$$

$$\vec{\theta} \leftarrow \vec{\theta} + \alpha^{\vec{\theta}} \gamma^t \delta \nabla \pi(a_t \mid S_t, \vec{\theta})$$

Actor-Critic Policy Gradient

Actor-critic method考虑引入bootstrapping机制，考虑**TD(0)**为例，即对于REINFORCE with Baseline中的G_t以$R_{t+1} + \gamma \hat{v}(S_{t+1}, \vec{w})$ 代替，这样做的好处我们在学习TD方法的时候已经或多或少接触了，首先毫无疑问，这样可以拓展到**continuing environment**；然后就是bootstrapping虽然引入了bias，但是却降低了variance，可以加快算法的收敛。

Policy Gradient for continuing problems

在continuing环境下，我们遇到的on-policy distribution实际上是markov chain中stationary distribution的概念：

$$\sum_s \mu(s) \sum_a \pi_\theta(a \mid s) p_{s,a}^{s'} = \mu(s')$$